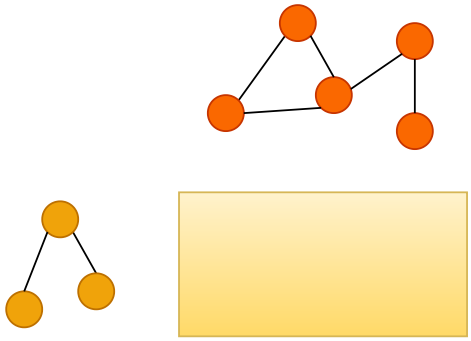
Making Multi-Axis Methods Feasible for Multi-Omics Data

Ethan 'Bailey' Andrew



Submitted in accordance with the requirements for the degree of:
Doctor of Philosophy in Artificial Intelligence for Medical Diagnosis and Care

UNIVERSITY OF LEEDS

School of Computer Science
March 2026

Intellectual Property and Publication Statements

I confirm that the work submitted is my own, except where work which has formed part of jointly authored publications has been included. My contribution and the other authors to this work has been explicitly indicated below. I confirm that appropriate credit has been given within the thesis where reference has been made to the work of others.

This thesis is based on the following publications:

- The main content of Chapter 2 and its corresponding Appendix A is published in the Proceedings of the 27th International Conference on Artificial Intelligence and Statistics (E. B. Andrew et al., 2024). I developed the mathematical theory, the implementation, and performed the experiments. Luisa Cutillo advised on mathematical theory and helped design the synthetic experiments; David Westhead helped design the real world experiments.

- The main content of Chapter 3 and its corresponding Appendix B is published in the Proceedings of the AAAI Conference on Artificial Intelligence (B. Andrew et al., 2025a). I developed the mathematical theory, the implementation, and performed the experiments. Luisa Cutillo advised on mathematical theory and helped design the synthetic experiments; David Westhead helped design the real world experiments.

- The main content of Chapter 4 and its corresponding Appendix C is published in Proceedings of the 28th International Conference on Artificial Intelligence and Statistics (B. Andrew et al., 2025b). I developed the mathematical theory, the implementation, and performed the experiments. Luisa Cutillo advised on mathematical theory and helped design the synthetic experiments; David Westhead helped design the real world experiments.

- The main content of Chapter 5 and its corresponding Appendix D is currently accepted for publication in BMC Medical Research Methodololgy. I developed the mathematical theory, the implementation, and performed the experiments. Luisa Cutillo advised on mathematical theory and helped design the synthetic experiments; David Westhead helped design the real world experiments.

- The main content of Chapter 6 and its corresponding Appendix E is currently submitted for publication in Bioinformatics. I developed the mathematical theory, the implementation, and performed the experiments. Luisa Cutillo advised on mathematical theory and helped design the synthetic experiments; David Westhead helped design the real world experiments. James Poulter and Erica Harris generated the lncRNA dataset we used to validate our results, and advised on interpreting such results.

- A draft paper of work intended to be submitted to NeurIPS 2026 is given in Appendix F. The draft presented is wholly my own, although the version that will ultimately be submitted will be expanded and improved based on feedback from Luisa Cutillo and David Westhead.

This thesis is submitted as an *alternative style of doctoral thesis*. We chose to submit under this style due to the significant amount of work already published or under consideration for publication. Chapters 1 and 7 comprise the introduction and conclusion to the thesis, in which we will lay out the overarching context and motivation for our work. The remainder of the chapters (2-6) are built around individual publications. Each paper chapter starts with an introduction to the paper and its place within the thesis, followed by the paper itself. Papers are included as-is from their publication, except for formatting changes. Additional work pertaining to the papers, such as proofs and additional experiments, are given in the paper's corresponding appendix section. In addition to the papers outlined above, I am also the lead author of a book chapter on recent developments in transcriptomics (B. Andrew et al., 2024); this is omitted from the thesis, but was written during the PhD.

This copy has been supplied on the understanding that it is copyright material and that no quotation from the thesis may be published without proper acknowledgment. The right of Ethan Bailey Andrew to be identified as Author of this work has been asserted by Ethan Bailey Andrew in accordance with the Copyright, Designs and Patents Act 1988.

Acknowledgments

Supervisory Team

Firstly, I would like to thank my supervisors Luisa Cutillo and David Westhead for their support, advice, and insights, as well as giving me the opportunity to travel to Italy and China. You both are the best supervisors anyone could ask for; I am lucky and grateful that you were mine. I would also like to thank the CDT executive committee (David Hogg, Vania Dimitrova, and Owen Johnson) for their support throughout the programme. Without them, this programme would not have existed, and I would have ended up trying my luck in Silicon Valley (yuck) rather than pursuing academia.

Peers and Colleagues[1]

I would also like to acknowledge many of my peers who made doing this PhD one of the best times in my life. I would like to thank Mary Paterson for sharing their family on Christmas, and Siân Carey ~~for not being very tall~~ for their bottomless compassion and moral fiber. I would like to thank Sam Llanwarne for helping me to map the interior of several buildings and for the many games of late-night chummus, and Victoria 'Tazamantha' Moglia for being the best step-fourth-cousin anyone could ask for. I would also like to thank Morgan Thomas and Rachael Harkness for our many ill-fated collaboration attempts, and Allan Pang for a plethora of for loops. I would like to thank Alex Coles for offering to teach me Warhammer 40k. Thank you to Yichao Liang for eating so many pizzas with me during COVID.

I would like to thank Benjamin Wilson and Benjamin Keel for being the two best Benjamins in my life (sorry Ben Affleck!). I would like to thank Jack Breen for his giant brain[2]. I would also like to thank Anna Grace-Linton for the conker you found on the ground, it's still somewhere in my desk drawer. I would like to thank Lucy Godson and Peter Hristov for being the friendliest power couple this side of the Urals, and Sara Jones for introducing me to 'pass the pigs'. I would also like to thank Sam Charlton for being so knowledgeable about local history/trivia, and for giving me an Irn Bru that one time.

I would like to thank Emma Briggs for being patient with me in returning their tupperware, and Zoe Hancox for their leading role in keeping Journal Club and AI^2 going. I would like to thank Lewis Howell for the rock crystal kit he once gave me, and Gerardo Loza for the many blessings he bestowed upon me. I would like to thank Aron Syverson for bringing that balance thingy into the office, and Christopher Winder for our many conversations during the Master's portion of this programme. I would also like to thank Oli Umney and Oli Mills for being the best Olis in my life, but I already said that for the Benjamins; instead, I will thank Oli Umney for his work in organizing the PPIE paper (sorry that I did not end up contributing!) and Oli Mills for having the blondest hair known to man. I would like to thank Zuzanna Wojcik for their advice in a time of crisis, and Mateusz Rychlicki for our many exchanges of tea.

I would like to thank Yushi Guo for reviving the dying office with their decision to start working here, and Raneem Toman for climbing to the top of Arthur's Seat with me. I would also like to thank Xin Ci Wong for helping me run journal club, in addition to our philosophical conversations. I would like to thank Auguste Rumbutyte for being a great friend, neighbor, and supervisor-sibling; I hope you have a great time in Lithuania. I would like to thank Sashi Kiran for their positivity, and most of all for introducing me to my wife Jahnavi! I would like to thank David Marples for always participating in Journal Club; without you, it would have died much sooner. I would also like to thank Manisha Waterston for introducing me to the world of surgical robotics, and Eleanor Bolton and James Battye for being excellent torchholders next generation[3].

[1] Many of whom are dear friends!

[2] And for submitting on-time with a job in hand, paving the way for the rest of us.

[3] You have big shoes to fill... but don't worry, I'm sure your feet'll be adequately sized.

Friends and Family

Dad, you're the one who got me into programming - look where that's led me! Mom, thank you for always being there for me, whether by getting me those the last-minute sugar cubes for that middle school project so long ago, or just last week when you helped research the family tree. Puck, I'm so jealous of the fact that you learned to crochet and play songs to audiences; those are both tasks I once-upon-a-time aimed to do, but never found the time or drive...

Thank you to Krishna and Kalpana for raising a wonderful daughter and welcoming me into your family. Sorry for stealing your daughter away to the far-flung land of Britain. Thank you to Babhita, Yuthika, and Kunal to being great friends to Janu and I, and thank you to Minisha, Adhya, and Tejaswiny for being excellent sisters-and-nieces-in-law.

Thank you to Nana Pam for taking care of me when I was young; I was once told you fed me so much chocolate I became morbidly obese, but I'm sure I was just obese with love. Thank you to Grandad Stan for the constant jokes and amazing, if incomprehensible, accent. You are both dearly missed. Thank you to Gram for their compassion and wisdom, as well as for helping pay for my undergrad. Thank you to Grandad Lewis for keeping me athletic and for their constant support throughout the years. I am glad that my parents are able to live close to you both after all this time.

Thank you to my closest childhood friends Henry Jacobsen, Alex Bednar, Jackson Terry, and Jennifer Leff; it was all of you who made me who I am. I'm sorry that we all lost touch. Thank you to my childhood friend Bradley Goldenberg, who inspired me with his book! Thank you to my childhood friend Alek Halem, who was taken too soon; you are often in my thoughts.

Miscellaneous

Thank you to George Lucas for making Star Wars; I'm not a fan, but it was probably a net positive for the world. Thank you to Lady Luck for not insisting on my demise that time I cracked my head open, that time I fell off a cliff into shallow, rocky water, or that time with the well-meaning but over-equipped police (you know what I mean). I was not the most careful child.

Janu



Figure 1: A woman so lovely I married her twice.

Janu, I couldn't have done this without you. Thank you for all your support and love - and, most of all, thank you for the tasty, tasty lemon rice.

Abstract

Multi-axis models are a promising technique for simultaneously producing information about all 'axes' in a dataset under modest assumptions. For single cell RNA-sequencing data, they learn gene and cell 'conditional dependency' networks. However, the applicability of such models is limited due to their inability to handle multi-modal datasets and poor scalability: they can only run on datasets with thousands of cells/genes, whereas modern datasets have 20,000-30,000 genes, *millions* of cells, and potentially additional modalities such as ATAC-seq. In this thesis, we increase the scalability of such models to millions of cells, and verify this by applying our methodology to one such million-cell dataset. We also generalize the model to multi-modal datasets. Finally, we investigate the assumptions underpinning these models and apply our techniques to learn causal networks. In particular, we demonstrate how to relax the zero mean assumption to a 'Kronecker-sum-structured mean' assumption, how to relax the Cartesian product assumption common in multi-axis models, and finally how to relax the Gaussian assumption, by generalizing multi-axis methodologies to the family of elliptical distributions. These relaxations are built on top of GmGM, and in particular rely on its scalability; they would not be feasible without the scalability work we undertook.

Contents

List of Figures **11**

List of Tables **14**

List of Abbreviations **15**

1 Introduction **17**
1.1 An introduction to 'omics datasets . . . 18
1.2 An introduction to graphical models . . . 20
1.3 An introduction to multi-axis models . . . 22
1.4 A review of multi-axis models . . . 24
1.5 My work in context . . . 25

Bibliography for the introduction **27**

2 GmGM: a fast multi-axis Gaussian graphical model **31**
2.1 Introduction . . . 33
2.1.1 Notation . . . 33
2.2 Prior work . . . 34
2.2.1 Unmet need . . . 35
2.3 Our contributions . . . 35
2.3.1 The model . . . 35
2.3.2 The algorithm . . . 36
2.3.3 Additional theorems . . . 37
2.3.4 Convexity and uniqueness . . . 38
2.4 Results . . . 38
2.4.1 Synthetic data . . . 38
2.4.2 Real data . . . 39
2.5 Limitations and future work . . . 42
2.6 Conclusion . . . 42

Bibliography for Paper 1 **43**

3 Gaussian graphical modelling without independence assumptions for uncentered data **45**
3.1 Introduction . . . 47
3.1.1 Our contributions . . . 48
3.1.2 Notation . . . 49
3.2 Estimation method . . . 50
3.3 Global optimality and complexity . . . 52
3.4 Results . . . 54
3.5 Conclusion . . . 57

Bibliography for Paper 2 **58**

4 The strong product model for network inference without independence assumptions 61
4.1 Introduction . . . 62
4.2 Background . . . 63
4.3 Our contributions . . . 64
4.4 Methodology . . . 64
4.4.1 Notation . . . 64
4.4.2 The strong product model . . . 65
4.4.3 Derivation of the method . . . 65
4.4.4 Geodesic convexity . . . 66
4.4.5 Riemannian gradient descent . . . 67
4.4.6 Identifiability . . . 67
4.4.7 Existence and uniqueness . . . 69
4.5 Experimental results . . . 69
4.5.1 Synthetic data . . . 71
4.5.2 COIL dataset . . . 72
4.5.3 scRNA-seq dataset . . . 72
4.6 Conclusion . . . 74

Bibliography for Paper 3 75

5 The Cartesian Gaussian additive noise model for directed network inference in omics data 79
5.1 Introduction . . . 80
5.2 Background: Kronecker-structured covariance estimation . . . 81
5.3 Methodology: the Cartesian additive noise model . . . 82
5.3.1 Fitting the model . . . 83
5.3.2 Tensor-variate datasets . . . 83
5.4 Theory: establishing the efficacy of the method . . . 84
5.4.1 Inferring the ordering . . . 85
5.5 Results . . . 86
5.5.1 Synthetic data . . . 86
5.5.2 Effect of ordering . . . 87
5.5.3 Simulated scRNA-seq data . . . 88
5.5.4 Establishing causal ordering . . . 88
5.5.5 Choosing the parameters . . . 89
5.6 Discussion . . . 90
5.7 Declarations . . . 90
5.7.1 Ethis approval and consent to participate . . . 90
5.7.2 Consent for publication . . . 90
5.7.3 Availability of data and materials . . . 90
5.7.4 Competing interests . . . 90
5.7.5 Funding . . . 90
5.7.6 Author contributions statement . . . 90
5.7.7 Acknowledgements . . . 90

Bibliography for Paper 4 91

6 Making multi-axis Gaussian graphical models scalable to millions of samples and features 93
6.1 Introduction . . . 95
6.2 Materials and methods . . . 96
6.2.1 Notation . . . 96
6.2.2 Assumptions . . . 97
6.2.3 The model . . . 98
6.2.4 Theory . . . 99
6.3 Algorithm . . . 99
6.4 Implementation . . . 100

6.5 Results . . . 100
6.5.1 Scalability . . . 101
6.5.2 Application to single cell neuronal differentiation data . . . 101
6.6 Discussion . . . 104
6.7 Acknowledgements . . . 104
6.8 Author Contributions . . . 104

Bibliography for Paper 5 **105**

7 Discussion and conclusions **107**
7.1 Implications . . . 107
7.2 Limitations . . . 107
7.3 Future Directions . . . 108
7.3.1 Applications . . . 108
7.3.2 Beyond tensor-variate data . . . 108
7.3.3 Beyond the Kronecker sum . . . 110
7.3.4 Becoming robust to noise . . . 111
7.3.5 Multi-axis mixture models . . . 112
7.4 Conclusion . . . 113

Bibliography for the discussion **113**

Appendix **115**

A GmGM: a fast multi-axis Gaussian graphical model **115**
A.1 Notation . . . 117
A.2 Proofs . . . 118
A.2.1 Permutations . . . 118
A.2.2 Derivation of the probability density function . . . 120
A.2.3 Gradient . . . 121
A.2.4 Maximum likelihood estimate for the eigenvectors . . . 122
A.2.5 Maximum likelihood estimate for the eigenvalues . . . 124
A.2.6 Incorporation of priors . . . 124
A.2.7 Covariance thresholding . . . 125
A.3 Dependencies . . . 127
A.4 Experiments . . . 127
A.4.1 Synthetic data . . . 127
A.4.2 COIL video . . . 129
A.4.3 EchoNet-Dynamic ECGs . . . 130
A.4.4 Mouse embryo stem cell transcriptomics . . . 132
A.4.5 LifeLines-DEEP metagenomics + metabolomics . . . 140
A.4.6 10x Genomics flash frozen lymph nodes . . . 141
A.5 Regularization . . . 142
A.6 Asymptotic complexity . . . 147
A.7 Centering the data . . . 147

Bibliography for Paper 1 (appendix) **149**

B Gaussian graphical modelling without independence assumptions for uncentered data **151**
B.1 Deferred proofs . . . 151
B.1.1 Derivation of gradients . . . 152
B.1.2 Derivation of the coordinate-wise minimum of m . . . 152
B.1.3 Derivation of the coordinate-wise minima of $\widetilde{\boldsymbol{\mu}}_\ell$ . . . 153
B.1.4 Derivation of $\mathbf{A}_\ell$ and $\mathbf{b}_\ell$ . . . 154
B.1.5 Invertibility of $\mathbf{A}_\ell$ . . . 156
B.1.6 Proof of global optimality . . . 157
B.2 Erdos-Renyi synthetic data . . . 160

B.3 Number of iterations needed . . . 160

Bibliography for Paper 2 (appendix) 161

C The Cartesian Gaussian additive noise model for directed network inference in omics data 165
C.1 Additional experiments . . . 166
C.2 Derivation of gradients . . . 166
C.2.1 Efficient gradients . . . 168
C.2.2 Addition of frobenius norm of kogarithm . . . 171
C.2.3 Riemannian gradients . . . 171
C.3 Efficient evaluation of log-likelihood . . . 172

Bibliography for Paper 3 (appendix) 173

D The Cartesian Gaussian additive noise model for directed network inference in omics data 175
D.1 Model derivation . . . 176
D.2 Gradient derivation . . . 177
D.3 Proofs . . . 180

Bibliography for Paper 4 (appendix) 189

E Making multi-axis Gaussian graphical models scalable to millions of samples and features 191
E.1 Evidence for the low-rank approximability assumption . . . 191
E.2 Derivation of the probability distribution . . . 192
E.3 Identifiability . . . 193
E.4 The algorithm . . . 196
E.5 Computational complexity . . . 196
E.6 Kronecker product and stridewise-blockwise trace lemmas . . . 197
E.7 Finding the maximum likelihood estimator . . . 198
E.8 Existence and uniqueness . . . 200
E.9 Derivation of the Fisher information . . . 201
E.10 Preserving input sparsity . . . 207
E.11 Additional experiments . . . 209
E.11.1 Synthetic data . . . 209
E.11.2 Sanity check of million-cell results . . . 209
E.11.3 COIL Dataset . . . 211
E.12 Neuronal scRNA-seq dataset details . . . 212
E.12.1 Culture of induced pluripotent stem cells (iPSCs) . . . 212
E.12.2 iPSC-derived neural progenitor cell (NPC) differentiation . . . 212
E.12.3 Cell fixation . . . 213
E.12.4 Barcoding, cDNA library preparation and sequencing . . . 213
E.13 Neuronal dataset cell marker genes . . . 213
E.14 GO terms for network metrics . . . 213
E.15 Neuronal scRNA-seq dataset full results . . . 213

Bibliography for Paper 5 (appendix) 225

F Making multi-axis models robust to multiplicative noise: how, and why? 227
F.1 Introduction . . . 228
F.2 Background . . . 229
F.3 Methodology . . . 230
F.3.1 Handling non-identifiability . . . 231
F.3.2 Removing the effect of the noise . . . 231
F.3.3 Performing expectation maximization . . . 232
F.3.4 Evaluating $\mathbf{S}^{\text{rows}}$ and $\mathbf{S}^{\text{cols}}$ with Laplace's approximation . . . 233

F.3.5 Finding $\mathbf{z}^*$ with quadratic programming . . . 233
F.4 Results . . . 233
F.4.1 Synthetic validation . . . 234
F.4.2 Systematic validation . . . 234
F.4.3 Specific evaluation . . . 234
F.5 Discussion and limitations . . . 235
F.6 Proof that f removes only the corrupted directions . . . 236
F.7 A version of f that handles zero values . . . 237
F.8 Proof that optimizing Q is equivalent to GmGM . . . 238
F.9 Efficient evaluation of $\frac{1}{2}\mathrm{tr}\left[\left(\mathbf{P}\boldsymbol{\Omega}_{(t)}\mathbf{P}^T\right)^{-1}\mathbf{P}\left(\nabla^2 g(\mathbf{z}^*)\right)\mathbf{P}^T\right]$ . . . 239
F.10 Description of the systematic approach to preprocessing and evaluating datasets . . . 240
F.11 Description of datasets used . . . 241

Bibliography for Paper 6 **242**

G Addendum to Chapter 2 ('GmGM: a fast multi-axis Gaussian graphical model') **247**

H Addendum to Chapter 3 ('Gaussian graphical modelling without independence assumptions for uncentered data') **251**

I Addendum to Chapter 4 ('The strong product model for network inference without independence assumptions') **255**

J Addendum to Chapter 5 ('The Cartesian Gaussian additive noise model for directed network inference in omics data') **257**

K Addendum to Chapter 6 ('Making multi-axis Gaussian graphical models scalable for millions of samples and features') **259**

List of Figures

1 A woman so lovely I married her twice. . . . 4

1.1 The difference between correlations and conditional dependencies. . . . 21
1.2 All possible interations in a dataset with two cells and two genes. . . . 24

2.1 Graphical structure of the LifeLines-DEEP dataset. . . . 35
2.2 A graphical overview of how the GmGM algorithm works. . . . 37
2.3 Comparison of GmGM runtimes to TeraLasso and EiGLasso. . . . 39
2.4 Precision-recall curves for GmGM, EiGLasso, and TeraLasso over synthetic 50x50 data. 39
2.5 Assortativity of cell types in learned graphs, duck recovery, and echocardiogram frames. 40
2.6 Assortativity of GmGM and ZiLN on Lifelines-DEEP data. . . . 40
2.7 PR curves for GmGM and GmGM+covariance thresholding. . . . 40
2.8 UMAP plots of GmGM on real scRNA-seq data. . . . 41

3.1 PR curves of GmGM and Noncentral GmGM on synthetic data. . . . 49
3.2 A graphical description of the Noncentral GmGM algorithm. . . . 51
3.3 PR curves when the underlying data is a KS-structured Gaussian. . . . 55
3.4 PR curves for unstructured-mean datasets. . . . 56
3.5 Assortativity on scRNA-seq data with GmGM and Noncentral GmGM. . . . 56
3.6 GmGM, TeraLasso, and their noncentral variants on COIL duck data. . . . 57

4.1 An empirical demonstration of linear convergence. . . . 67
4.2 PR curves for various 500-node graphs. . . . 70
4.3 How AUPR varies as between-row and within-row variances change. . . . 71
4.4 The strong product model's results on the COIL dataset. . . . 72
4.5 Assortativity as we vary thresholding with the strong product method. . . . 73

5.1 A description of Krumsiek data and our assumed ordering. . . . 88
5.2 Two gene graphs arising from ⊞-LGAM's grid search. . . . 88
5.3 The best baseline LGAM graph, and clustering metrics for LGAM and ⊞-LGAM. . . . 89

6.1 An example two-modality dataset. . . . 97
6.2 The runtime of various multi-axis algorithms as the problem size increases. . . . 100
6.3 Clustering metrics on scRNA-seq data, comparing GmGM and hdWGCNA. . . . 101
6.4 GmGM's cell type classification, and the emergent connection of one learned module and cortical neurons. . . . 103

A.1 Graphical representation of the benefits of Theorem 3. . . . 125
A.2 Various PR curves on synthetic data. . . . 130
A.3 A demonstration of the benefits of using shared axes. . . . 131
A.4 PR curves on tensor-variate data. . . . 131
A.5 Runtimes as the number of axes increase. . . . 132
A.6 A reconstruction of the COIL-20 duck video after shuffling, using GmGM. . . . 133
A.7 A validation of GmGM's recovery of heartbeats. . . . 133
A.8 Assortativity when using various thresholding methods. . . . 137
A.9 Assortativity when using various thresholding methods after centering the data. . . . 137
A.10 Increasing regularization strength with GmGM. . . . 138

A.11 Effect of regularization on assortativity. . . . 138
A.12 A comparison of GmGM and EiGLasso using assortativity on scRNA-seq data. . . . 139
A.13 A 'UMAP consistency plot'. . . . 141
A.14 Comparison of GmGM and ZiLN's assortativity on LifeLines-DEEP. . . . 142
A.15 Assortativity when incorporating prior knowledge. . . . 143
A.16 'UMAP and tSNE consistency plots' on LifeLines-DEEP data. . . . 144
A.17 'UMAP consistency' on 10x Genomics scRNA-seq data. . . . 145
A.18 More UMAP consistency. . . . 145
A.19 More tSNE consistency. . . . 146
A.20 Expression of top differentially-expressed genes by cluster. . . . 146
A.21 Runtime comparison of our method with and without regularization. . . . 146
A.22 Self-connectedness as regularization varies. . . . 147
A.23 Assortativity as the number of kept edges per cell varies. . . . 147

B.1 Precision and recall for a synthetic dataset. . . . 161
B.2 Precision and recall on another synthetic dataset. . . . 162
B.3 Precision and recall as the number of iterations increase. . . . 163

C.1 An enlarged version of the COIL results plot. . . . 166
C.2 AUPR as ρ varies. . . . 167
C.3 Performance of algorithms when data is generated from the Kronecker sum or Kronecker product structures. . . . 168
C.4 The effect of different graph sizes on performance. . . . 169
C.5 AUPR as y changes. . . . 170

D.1 Reported MCCs during a grid search. . . . 176

E.1 Real datasets are often low rank. . . . 191
E.2 PR curves for large datasets. . . . 210
E.3 A comparison of several methods to find the frame graph. . . . 211
E.4 A neuronal scRNA-seq UMAP. . . . 212

F.1 Estimated tail dependence across 1000 randomly selected pairs of genes (left) and cells (right) from the E-MTAB-7249 scRNA-seq dataset discussed in Section F.4.3, compared to random data generated from a Gaussian copula. . . . 229
F.2 MED-MAGMA PR curves. . . . 234
F.3 MED-MAGMA on real data. . . . 235
F.4 MED-MAGMA captures cell-type-specific gene modules. . . . 235
F.5 MED-MAGMA AUPR as sparsity and noise vary. . . . 238

G.1 A revised Figure 2.6a, splitting each level into its own plot for easy comparison between GmGM (solid lines) and ZiLN (dot-dashed). . . . 248
G.2 A revised Figure 2.6b, splitting each level into its own plot for easy comparison between GmGM (solid lines) and ZiLN (dot-dashed). . . . 248
G.3 A revised Figure 2.6c, splitting each level into its own plot for easy comparison between GmGM (solid lines) and ZiLN (dot-dashed). Note that, because the phylum is given as a prior, GmGM recovers it very well. . . . 249
G.4 GmGM (Leiden resolution=1.2) clusters compared to cell-surface-protein-based annotations from a 10X Genomics human PGMC dataset. . . . 250
G.5 Each row represents the composition of a single GmGM cluster in terms of the 10X cell type labels. Only when a 10X label comproses at least 20% of a cluster is the percentage displayed on the plot. . . . 250

H.1 A histogram of the mean log-expression across cells for all genes (orange) compared to just the mitosis-related genes that we used (blue). The spread of expression values of the two categories is roughly equal, but the mitosis-related genes tend to utilize more of that spread, whereas the expression values of all genes are much more concentrated on the low end. . . . 252

H.2 (Figure 3.1) Precision and recall for a synthetic dataset in which data was generated from a Kronecker-sum normal distribution with a constant value of 1 added to every entry of the matrix. Error bars are the best/worst performance over 10 trials; the center line is average performance. 253
H.3 (Figure 3.3) Precision and recall for on a synthetic dataset in which data was generated from a noncentral Kronecker-sum normal distribution. Noncentral TeraLasso's high susceptibility to changes in the regularization parameter led to our grid search being too coarse-grained during some trials. 254
H.4 (Figure 3.4) Precision and recall for on a synthetic dataset in which data was generated from a zero-mean Kronecker-sum normal distribution, with independent Gaussian (left) and Poisson (right) noise added to every element. Error bars are the best/worst performance over 10 trials; the center line is average performance. 254

K.1 (Left) The percentage of gene expression taken by the upper-quartile-ranked gene in m13 (i.e. the gene in m13 whose expression is greater than 75% of other m13 genes but less than the remaining 25%). (Right) The same as the left plot, but with a cutoff of 0.01% to make the differences clearer. 260

List of Tables

1.1 Precision matrix decompositions in the literature. 23

2.1 A comparison of the asymptotic complexity of TeraLasso and GmGM. 36

3.1 The distributions of means in synthetic data experiments. 54

4.1 A comparison of different Kronecker-structured decomposition methods. 63
4.2 Metrics for the COIL dataset experiment with the strong product. 73

5.1 Results on synthetic data for LGAM, ⊞-LGAM, and GmGM. 87
5.2 Results on synthetic data when the assumed ordering is increasingly incorrect. 87

6.1 Overexpressed modules for each GmGM cell cluster. 103

A.1 The point at which a K-axis tensor of double precision floats surpasses a gigabyte of data. 129
A.2 A comparison of the runtimes our algorithm and TeraLasso. 134
A.3 Runtimes of our algorithm and TeraLasso, with various preprocessing methods, on the EchoNet-Dynamic dataset. 134
A.4 Mitral valve labelings and precision matrices for the EchoNet-Dynamic dataset (without nonparanormal skeptic). 135
A.5 Mitral valve labelings and precision matrices for the EchoNet-Dynamic dataset (with nonparanormal skeptic). 136
A.6 Runtime comparison of GmGM and EiGLasso on scRNA-seq data. 140
A.7 Runtimes of various algorithms on the LifeLines-DEEP dataset. 140
A.8 Runtimes of our algorithm and EiGLasso, with various preprocessing techniques, on the 10x Genomics data. 141
A.9 Computational complexities of various algorithms. 148

E.1 Identifiable parameterizations for several datasets. 195
E.2 Allowed interactions between parameters in $\mathbf{F}$. 206
E.3 Significant GO terms for GmGM gene clusters. 209
E.4 The performance of several methods on reconstructing the frame graph of COIL-20 videos. 211
E.5 Precision/recall values for various algorithms on COIL data. 212
E.6 Marker genes for various neuronal cell types. 213
E.7 GO term choices for metric comparison in Figure 6.3. 214
E.8 Every hdWGCNA module, in order of size. 215

F.1 All datasets analyzed during Section F.4.2. 241

List of Abbreviations

AAAI Association for the Advancement of Artificial Intelligence
AISTATS Artificial Intelligence and Statistics Conference
AMI Adjusted mutual information
ATAC Assay for transposase-accessible chromatin
ATAC-seq ATAC sequencing
AUPR Area under the precision-recall curve

BiGLasso Bi-Graphical Lasso
BMC BioMed Central

CLGAM Cartesian linear Gaussian additive noise model
COCA Copula Component Analysis

DAG Directed acyclic graph
DNA Deoxyribonucleic acid
DNNlasso Diagonally Non-Negative Graphical Lasso
DPT Diffusion Pseudotime

EiGLasso Eigen Graphical Lasso
EM Expectation Maximization

g-convex Geodesically convex
GGM Gaussian graphical model
glasso Graphical Least Absolute Shrinkage and Selection Operator
GmGM Gaussian multi-Graphical Model
GO Gene ontology
GRN Gene regulatory networks

hdWGCNA High-Dimensional Weighted Correlation Network Analysis
HGSOC High-grade serous ovarian carcinoma

iPSC Induced pluripotent stem cell

KKT Karush-Kuhn-Tucker conditions
kNN k-nearest neighbors

lasso Least Absolute Shrinkage and Selection Operator
LGAM Linear Gaussian additive noise model
lncRNA Long non-coding RNA

MCC Matthew's correlation coefficient
MED-MAGMA The Matrix Elliptical Distribution's Multi-Axis Graphical Modelling Algorithm
MLE Maximum likelihood estimation

NeurIPS Neural Information Processing Systems
NLL Negative log-likelihood
NP-hard Class of problems at least as hard as those that can be solved in nondeterministic polynomial time

PAGA Partition-Based Graph Abstraction
PBMC Peripheral blood mononuclear cells
PCA Principal Component Analysis
PR Precision-recall

QP Quadratic programming

RNA Ribonucleic acid
RNA-seq RNA sequencing

scATAC-seq single-cell ATAC sequencing
scBiGLasso Scalable Bi-Graphical Lasso
scRNA-seq single-cell RNA sequencing
snRNA-seq single-nucleus RNA sequencing

TeraLasso Tensor Graphical Lasso

UMAP Uniform Manifold Approximation and Projection

WGCNA Weighted Correlation Network Analysis

ZiLN Zero-inflated log normal

List of Symbols

$a, b, c, ...$	*Italic lowercase letters.*	Scalars.
$A, B, C, ...$	*Italic uppercase letters.*	Scalars, especially when used as a count of things that are iterated over.
$\mathbf{a}, \mathbf{b}, \mathbf{c}, ...$	*Bold lowercase letters.*	Vectors.
$\mathbf{A}, \mathbf{B}, \mathbf{C}, ...$	*Bold uppercase letters.*	Matrices.
$\mathcal{A}, \mathcal{B}, \mathcal{C}, ...$	*Calligraphic uppercase letters.*	Tensors, metric spaces, and distributions.
$\mathfrak{A}, \mathfrak{B}, \mathfrak{C}, ...$	*Fraktur uppercase letters.*	Graphs and optimization problems.
ℓ	*L.*	An arbitrary axis.
L	*L.*	The number of axes in a dataset.
γ	*Lowercase gamma.*	An arbitrary modality.
Γ	*Uppercase gamma.*	The number of modalities in a dataset.
d_ℓ^γ and similar	*d with various sub/superscripts.*	Quantities pertaining to axis size and sample counts.
$\mathbf{I}_{a\times a}$	*Bold I.*	The $a \times a$ identity matrix.
$\mathbf{0}_{a\times a}$	*Bold zero.*	The $a \times a$ zero matrix.
$\mathbf{J}_{a\times a}^{(ij)}$	*Bold J with script.*	The matrix of zeros except with a 1 at (i, j).
$\boldsymbol{\mu}$	*Mu.*	A mean.
$\boldsymbol{\Omega}$	*Omega.*	The precision matrix between matrix entries.
A_ℓ^γ	*Value with scripts.*	A value for a specific axis ℓ and modality γ.
$\boldsymbol{\Psi}_\ell$	*Psi with subscript.*	The precision matrix of axis ℓ.
$\boldsymbol{\Theta}_\ell$	*Theta with subscript.*	Same as $\boldsymbol{\Psi}_\ell$ (rare).
$\mathbf{S}_\ell$	*S with subscript.*	The empirical Gram matrix of axis ℓ.
$\boldsymbol{\Sigma}_\ell$	*Sigma with subscript.*	The true covariance matrix of axis ℓ.
$\mathcal{K}_{++}^{\oplus}$	*Calligraphic K with $\oplus$/++.*	The space of all Kronecker-sum-decomposable positive-definite matrices.
$\mathbb{E}[\cdot]$	*Expected value.*	Expected value.
$\mathbb{P}[\cdot]$ or $\text{pdf}(\cdot)$	*Probability density.*	Probability density.
$\|\cdot\|_a$	*Norm.*	Some norm, indicated by subscript.
$\underset{X}{\text{argmax}}[\cdot]$	*Argument of maximum.*	The value X that maximizes the formula.
$\text{vec}[\cdot]$	*Vectorization.*	Flattens a matrix by stacking its columns.
$\text{diag}[\cdot]$	*Diagonal.*	Diagonal matrix with given diagonals.
$\det[\cdot]$ or $\|\cdot\|$	*Determinant.*	The determinant of a matrix.
$\det^\dagger[\cdot]$	*Pseudo-determinant.*	The pseudo-determinant of a matrix.
$\text{tr}[\cdot]$	*Trace.*	Sum of the diagonals of a matrix.
$\text{tr}_b^a[\cdot]$	*Stridewise-blockwise trace.*	Defined in Equation 2.1.
$\oplus$	*O-plus.*	The Kronecker sum.
$\otimes$	*O-times.*	The Kronecker product.
$\tilde{\oplus}$	*Tilde-O-plus.*	The Kronecker sum for vectors.
$\bigoplus_\ell^L$	*Big O-plus.*	Analogous to $\sum_\ell^L$ for $\oplus$.
$\bigotimes_\ell^L$	*Big O-times.*	Analogous to $\sum_\ell^L$ for $\otimes$.
$\mathbf{A} \succ \mathbf{B}$	*Lowner ordering.*	The smallest eigenvalue of $\mathbf{A}$ is larger than the largest of $\mathbf{B}$.
$A \sim \mathcal{D}$	*Is drawn from.*	A is drawn from some distribution $\mathcal{D}$.
$a \sim_{\mathfrak{G}} b$ or $a \sim b$	*Is connected to.*	a is connected to b in some graph $\mathfrak{G}$.
$a \perp b \mid c$ or $a \perp\!\!\!\perp b \mid c$	*Conditionally independent.*	a is conditionally independent of b given c.

Chapter 1

Introduction

It is often the case that one wants to learn a network (a 'graph') describing a dataset; not only can they be a useful pre-processing step in a pipeline (e.g. for clustering), but also because they can contain valuable information in their own right. This is especially true in the field of bioinformatics, in which learning 'gene regulatory networks', 'gene co-expression networks', and 'cell association networks' (among others) are key goals in the field. The goal of this thesis is to develop **scalable methods** that make only **weak assumptions about the data** while being applicable to a **large variety of datasets**.

We will focus primarily on bioinformatics datasets (often called *'omics datasets*) and especially single cell RNA-sequencing (scRNA-seq) datasets, but the methods we develop are applicable more broadly to datasets that can be in expressed in the form of a set of matrices or tensors. Historically, bioinformatics has used 'bulk' datasets - subject-level, rather than cell-level, aggregates of data. For this type of data, it can be reasonable to make an 'independence assumption' (i.e. that the subjects are statistically independent). Such an assumption is not always warranted (a longitudinal bulk-RNA dataset would have dependencies between timepoints), but it is often 'good enough' and comes with significant advantages: methods that make such an assumption are typically more tractable than those that do not, both in terms of scalability and statistical power.

Example Dataset

Example 1 (Bulk RNA-seq)**.** *A dataset of multiple cancer patients; each patient has had a biopsy, and the bulk RNA from their biopsy has been sequenced.*

Motivation. Knowledge of gene expression (RNA), especially when paired with patient metrics, can lead to the discovery of new targets for cancer treatments. Can also be used for diagnostic and prognostic purposes due to its relative cost effectiveness compared to scRNA-seq.

Example uses. This type of data is exceedingly common; it is impossible to fully summarize its uses. As one example, it has been used in ovarian cancer to identify candidate biomarkers for early detection (Mosig et al., 2012) and identify low grade serous ovarian cancer's cellular origins (Qiu et al., 2017), among other uses (see the review of J. Wang et al. (2019)).

Form. A matrix $\mathbf{X}$; $\mathbf{X}_{pg}$ represents the expression of gene g in patient p, typically in terms of a count of measured RNA transcripts corresponding to the gene.

More recently, the use of scRNA-seq data has exploded, with studies, authors, and cell counts all growing exponentially (Svensson et al., 2020). Whereas last decade such datasets were rarely more than 10,000 cells, several datasets have now passed the million-cell mark. The largest dataset we are aware of contains seven million cells (Bolognesi et al., 2025); it, among many other million-cell datasets, is available on the Human Cell Atlas (Regev et al., 2017), an international effort to create a comprehensive reference map of all human cells. This explosion of data and interest requires the development of new types of algorithms - while the independence assumption may have been suitable for bulk data, the samples of scRNA-seq are individual cells which developed and interacted within the same micro-environment.

There is good reason to focus on scRNA-seq data; our body contains many different cell types, some with exceedingly different RNA expression patterns and some with rather similar patterns. In a bulk RNA dataset, we can only see the aggregate gene expression across all cell types in the sample. While deconvolution methods can be used to estimate the constituent cell types from bulk data, such estimations are necessarily limited by the quality of the reference atlas used for the deconvolution - a reference atlas that must, by its nature, be established on the basis of domain knowledge of the gene expression of individual cell types. This is inherently limited, especially in the context of cancer, in which there is considerable heterogeneity from one patient to the next and thus would require bespoke reference atlases.

Example Dataset

Example 2 (scRNA-seq)**.** *A dataset from a single biopsy of a cancer patient; from each cell, the RNA has been sequenced separately.*

Motivation. Same benefits as Example 1, except that it is unlikely to be able to be used routinely for diagnosis/prognosis due to the expense of collecting the data. Can distinguish between the gene expression profiles of different cell types due to the cellular resolution, and can be used to investigate cell-cell relationships as well. Overall, gives much more biological information than bulk RNA-seq despite having less immediate medical utility.

Example uses. scRNA-seq datasets have been used to show the effect of the cellular composition of a tumor on the prognosis of lung adenocarcenoma (Guo et al., 2018). They have also been used to study which cells are likely to expand in CAR-T therapy[a] (Sheih et al., 2020).

Form. A matrix $\mathbf{X}$; $\mathbf{X}_{cg}$ represents the expression of gene g in cell c, typically in terms of a count of measured RNA transcripts corresponding to the gene.

[a] **CAR-T cell**: A type of T cell that has been taken from and then returned to a patient, having been modified to target an antigen that is specific to the cancer. CAR stands for 'chimeric antigen receptor'.

In this thesis, we address the following problems: how to learn gene networks on modern-sized datasets without the independence assumption (Chapters 2 and 6), how to further relax assumptions to better fit the data (Chapters 3 and 4), and finally how to learn *directed* gene networks without the independence assumption (Chapter 5). Each chapter corresponds to a paper (either published or currently under review); among these, the most significant contribution of this thesis is an algorithm, GmGM, which does not make the independence assumption, can be used to analyze million-cell datasets *on a personal computer in only a couple minutes*, and has been validated on novel biological data.

The rest of this chapter will give a more thorough introduction to 'omics data (Section 1.1), multi-axis models (Sections 1.2 and 1.3), and place my work in the context of pre-existing research gaps (Section 1.5).

1.1 An introduction to 'omics datasets

The **Central Dogma of Biology** states that our DNA encodes our genes, which is then *transcribed* into RNA before being *translated* into proteins; datasets capturing this information can be denoted genomics, transcriptomics, and proteomics, respectively. At a very high level, genomics captures 'what our body can do', transcriptomics captures 'what our body wants to do', and proteomics captures 'what our body is doing'. Reality, of course, is more complicated: for example, not every RNA transcript codes for a protein (some have functional roles).

However, the high-level intuition is useful for understanding at a glance why one may be interested in transcriptomics data (RNA-sequencing, 'RNA-seq'): RNA, like DNA, is much more amenable than protein to untargeted high-throughput sequencing. In other words, it is possible to capture a majority of the transcriptome in a cell in one go. Yet unlike DNA, RNA tells us which genes are actually expressed, i.e. which genes are active in the processes governing the cell.

Beyond the central dogma, there are a variety of biological processes ('modalities') that fall under the domain of 'omics. DNA can be modified in various ways that affect gene expression, such as via methylation or through changes to the chromatin rendering certain genes inaccessible. 'Chromatin'

refers to the material that makes up chromosomes; which comprises DNA and also 'nucleosomes' - the proteins around which DNA wraps, giving the chromosome its density and structure. Depending on how densely packed the chromatin is, regions of DNA may be inaccessible to various enzymes, such as RNA polymerase (responsible for transcribing DNA into RNA). The accessibility of chromatin to such enzymes can be probed using 'Assay for Transposase-Accessible Chromatin using sequencing' (ATAC-seq) methodology, which measures the ability of Tn5 transposase to bind to the chromatin[1]. ATAC-seq is often paired with transcriptomics data, as it naturally complements scRNA-seq's answer to 'which genes are expressed' with an answer to the question 'which genes are *expressible*'. Beyond sub-cellular processes, metagenomics datasets can capture the contents of microbiomes, such as the gut microbiome.

'Omics datasets can often be taken at either a 'bulk' or 'single-cell' level. Bulk RNA-seq measures the total RNA expression across a set of cells, whereas single-cell RNA-seq (scRNA-seq) measures RNA expression separately for each individual cell in the dataset. In this thesis, we are primarily interested in single-cell datasets.

Below are a few examples of different kinds of transcriptomics datasets. Each example has same theme (cancer) to emphasize that they provide different options for investigating the same thing; however, each example could easily be adapted into other (non-cancer) contexts.

Example Dataset

Example 3 (Paired scRNA-seq/scATAC-seq)**.** *A dataset from a single biopsy of a cancer patient; from each cell, the RNA has been sequenced and ATAC-seq has been performed.*

Motivation. Same benefits as Example 2, while complementing it with information on gene accessibility (measured as 'peaks'). Could also be 'unpaired', i.e. the cells measured by each modality are from the same sample but different - in the unpaired case, no cell has both RNA and ATAC measured.

Example uses. This technology has been used to identify regulators of breast cancer metastasis (Xu et al., 2021) as well as map the developmental trajectory of cell types (J. Wang et al., 2023).

Form. Two matrices $\mathbf{X}^{(\text{scRNA-seq})}$ and $\mathbf{X}^{(\text{scATAC-seq})}$; $\mathbf{X}^{(\text{scRNA-seq})}_{cg}$ represents the expression of gene g in cell c, and $\mathbf{X}^{(\text{scATAC-seq})}_{cp}$ represents the strength of peak p in the same cell c.

Example Dataset

Example 4 (Longitudinal bulk RNA-seq)**.** *A dataset of multiple blood cancer patients in complete or partial remission; each patient has had their bulk RNA taken from a blood sample at three different time points: diagnosis, after the first round of chemotherapy, and after determination of being in remission.*

Motivation. Same benefits as in Example 1, but with a temporal nature allowing one measure the effects of treatment on gene expression.

Example uses. Such studies have been used to investigate the process of aging in vertebrates (Baumgart et al., 2016) and the disease mechanisms of psoriasis (L. Wang et al., 2018).

Form. A tensor $\mathcal{X}$; $\mathcal{X}_{tpg}$ denotes the expression of gene g in patient p at time t.

Example Dataset

Example 5 (Spatial RNA-seq)**.** *A histopathology slide[a], paired with gene expression values at many points on the image.*

Motivation. Can put the findings of RNA-seq in a spatial context. Histopathology slides are routinely collected and studied; findings in the slide can help inform findings from the RNA and vice-versa.

[1] Tn5 transposase is chosen for the task as it can cut and tag accessible regions of the DNA ('tagmentation'), making it easy to measure via sequencing.

If the resolution at which RNA-seq is measured is high enough, can provide the same benefits as scRNA-seq as well. Useful for the study of the tumor microenvironment.

Example uses. This is the newest technology discussed, and yet it has already been applied in the study of almost all solid organ malignancies (Cilento et al., 2024).

Form. An image $\mathbf{M}$ along with a tensor $\mathcal{X}$; $\mathcal{X}_{rcg}$ containing the expression of gene g in row r column c in grid.

[a] **Histopathology slide**: A microscopic image of a 2D slice of tissue.

In this thesis, we will develop methodologies to produce networks describing datasets in each of these examples. Each example given can be expressed in the form of a set of matrices/tensors; the methodology we will develop will work for any dataset that can be expressed in such a manner. While this does include spatial data (Example 5), it is not a focus of our work. Adding spatial information can allow deeper insights into one's data; for example, one can use the spatial context to identify neighbors of cells and even structures of cell agglomerations (such as the shape of a tumor boundary or the presence of tertiary lymphoid structures). Our methods are still applicable on this type of data; one could first identify cells present in the spatial data, then organize the data into a scRNA-seq-esque data matrix on which we run our methods. However, in doing so we would loose all spatial context; spatial data would benefit from more bespoke technologies that can take advantage of this context.

1.2 An introduction to graphical models

The goal of this thesis is to efficiently learn graphs describing an 'omics dataset; this is an ambiguous goal. There are multiple ways to describe a dataset. For example, if one's vertices are 'people', should edges represent friendship? Or should they instead represent physical proximity in the last week? The first may be useful for playing 'seven degrees of Kevin Bacon', but the latter is useful for contact tracing. Two common types of graphs are those based on spatial nearness and correlations.

Example 6 (Spatial nearness)**.** *Two vertices are connected if they are near each other in feature space. A k-nearest neighbors (kNN) graph is an example.*

Example 7 (Correlation)**.** *Two vertices are connected if they are correlated. A gene co-expression graph is an example.*

Both are frequently used on 'omics data; a kNN graph is usually generated as a precursor to popular visualization techniques such as UMAP (McInnes et al., 2020), as well as for cell clusterings in preparation for cell type identification. There are many methods to create gene co-expression graphs, such as WGCNA (Langfelder & Horvath, 2008), which are used to generate candidate gene regulatory modules (van Dam et al., 2017).

However, spatial measures are highly dependent on the scaling of the features, as well as the metric chosen. Correlation networks tend to be dense, with many of the edges capturing only 'indirect' relations. Consider the classic example of ice cream sales being correlated with shark attacks; this is true, but only because the confounding factor of 'it being hot outside' encourages both the eating of ice cream and the eating by sharks (via the mediator of beach visits). The relationship between ice cream and shark attacks is only 'indirect', knowledge of this link is of limited value. Much more useful would be knowledge of the direct effects.

Example 8 (Causal)**.** *Two vertices are connected if one of them causes the other; unlike the other graphs considered, the edges in this graph are directed. A gene regulatory network is an example of this.*

Knowledge of a causal graph describing how the genes regulate each other is an important end goal of transcriptomics. However, the burden of proof for causal declarations is quite high, and can only truly be satisfied with interventional experiments. Causal networks learned from observational data are useful for the generation of plausible hypotheses to be confirmed via a later interventional experiment, and we consider the generation of such networks in Chapter 5. Other than this excursion, the remainder of the thesis is devoted to learning *conditional dependency graphs*.

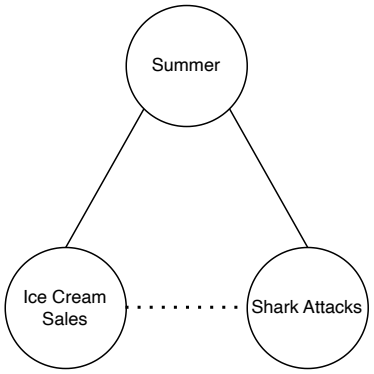


Figure 1.1: Solid lines represent correlations and conditional dependencies; dotted lines represent correlations. There is no direct link between ice cream sales and shark attacks, but they are correlated due to their mutual direct links with the heat of summer.

Example 9 (Conditional dependency)**.** *Two vertices are connected if they are conditionally dependent on one another.*

We say that two genes are conditionally dependent on one another if, after one conditions out the rest of the dataset, they are statistically independent from one another. This notion captures only the 'direct effects' between two genes. Shark attacks and ice cream sales are conditionally **independent** of each other, because once one conditions out the confounder of 'it being hot outside' the link between them disappears. This example is summarized graphically in Figure 1.1.

$$i \perp j|\mathbf{k} \iff \mathbb{P}\left(i|\mathbf{k}\right)\mathbb{P}\left(j|\mathbf{k}\right) = \mathbb{P}\left(i \cap j|\mathbf{k}\right)$$
(Conditional independence of i and j given the rest of the dataset $\mathbf{k}$)

Other than Chapter 5, this thesis focuses on the efficient learning of conditional dependency networks (a type of 'probabilistic graphical model'). Not only are they more meaningful than correlation networks, but their relative sparsity allows the development of very large-scale models for which storing a dense graph would be prohibitive. The amount of memory needed to store a dense graph with d vertices is $O(d^2)$, but sparse graphs need much less; one can make very reasonable assumptions about the data to arrive at an $O(d)$ memory requirement (for example, by assuming that there is a constant upper bound on how many direct effects any typical gene may have). When one considers that there are around 20,000-30,000 genes in a typical scRNA-seq dataset, this can be the difference between needing to use a high-performance computing cluster and being able to use a personal computer to fit the graph in RAM during an analysis.

Conditional dependency networks can be much harder to learn than correlation networks, but if the data is Gaussian then they are encoded by the 'inverse covariance matrix' ($\mathbf{\Sigma}^{-1}$), also called the 'precision matrix' ($\mathbf{\Psi}$). In particular, the nonzero elements of $\mathbf{\Psi}$ define which vertices are conditionally dependent on each other.

$$\begin{bmatrix}\mathbf{x}_i\\ \mathbf{x}_j\\ \mathbf{x}_\mathbf{k}\end{bmatrix} \sim \mathcal{N}\left(\boldsymbol{\mu}, \mathbf{\Psi}^{-1}\right)$$
$$\implies \mathbf{\Psi}_{ij} \neq 0 \iff i \perp j|\mathbf{k}$$

There are many methods which take advantage of this property, such as neighborhood selection (Meinshausen & Bühlmann, 2006) and the graphical lasso (Friedman et al., 2008). The graphical lasso, or 'glasso', is perhaps the most famous; to learn $\mathbf{\Psi}$ (and hence the graph of conditional dependencies), Friedman et al. set the problem up as a regularized maximum likelihood estimation (MLE) problem.

$$\hat{\mathbf{\Psi}} = \underset{\text{symmetric } \mathbf{\Psi}}{\operatorname{argmin}} \operatorname{tr}[\mathbf{\Psi}\mathbf{S}] - \log\det\mathbf{\Psi} - \rho\|\mathbf{\Psi}\|_{1,od} \qquad \text{(Graphical Lasso)}$$

$$\|\mathbf{X}\|_{1,od} = \sum_{i \neq j} |\mathbf{X}_{ij}|$$

In the above equation, $\mathbf{S}$ is the empirical covariance matrix, ρ is a regularization hyperparameter, and $\|\cdot\|_{1,od}$ is an L1 penalty on the off-diagonal elements of $\mathbf{\Psi}$. This is to ensure it learns a sparse graph, in line with our expectation of conditional dependency graphs being sparse. Most of the methods we develop in this thesis are based in some way on the graphical lasso.

Of course, most datasets are not Gaussian; in particular, scRNA-seq datasets are often comprised of integer values (counts of gene expression)! Instead of making a Gaussian assumption, the nice correspondence of $\mathbf{\Psi}$ and the graph of conditional dependencies can be recovered by making a *Gaussian copula* (or 'nonparanormal') assumption (Liu et al., 2012). On a practical level, this entails using a rank-based estimator for $\mathbf{S}$ (the 'nonparanormal skeptic'), rather than using the empirical covariance matrix; all other steps of the analysis proceed the same.

Assumption 1 (Gaussian copula assumption)**.** *Intuitively, this assumption states that our data can have* ***any marginal distribution****, as long as the genes 'interact Gaussian-ly'. Formally, we assume that our dataset is a monotonic transformation f of Gaussian data.*

$$f^{-1}(\mathbf{x}) \sim \mathcal{N}\left(\boldsymbol{\mu}, \mathbf{\Psi}^{-1}\right)$$

In particular, count distributions such as Poisson and negative binomial can adhere to the Gaussian copula assumption (depending on how the variables interact).

The graphical lasso, equipped with the nonparanormal skeptic, is very scalable and quite general. However, it **also makes the independence assumption**. For scRNA-seq datasets, if one wants to learn a graph of genes, one has to assume that there is no graph of cells, and vice versa. Both of these graphs are useful in their own right - the first can teach us which genes may regulate each other, and the second can teach us about the cellular microenvironment. Circumventing this deficiency is the subject of the next section and this thesis as a whole.

1.3 An introduction to multi-axis models

Multi-axis models (also called 'Kronecker-separable models') have two goals: to avoid the independence assumption, and to learn graphs for each 'axis' of the data (matrix-variate datasets have two axes, rows and columns; in the context of scRNA-seq, these axes are cells and genes). They form a tractable midpoint between two extremes. At one extreme, the standard independence assumption is that there are no row (cell) dependencies. At the other extreme, one could assume 'full dependence' - every matrix element (i_1, i_2) could potentially be dependent on any other matrix element (j_1, j_2): any gene in any cell can affect the expression of any other gene in any other cell. This is intractible; for $d \times d$ input data it would require $O(d^4)$ space to compute. Not only that, but one can hardly have any statistical certainty in the results when one only has a single sample for each matrix element!

Multi-axis models sit between these two extremes; rather than avoiding making any independence assumption, they make an assumption about how the dependencies of the two axes *interact*. They factor the full dependency matrix $\mathbf{\Omega}$ into some function ζ of factor dependency matrices, one for each axis: $\mathbf{\Omega} = \zeta\left(\mathbf{\Psi}^{(rows)}, \mathbf{\Psi}^{(cols)}\right)$. By doing so, they reduce the parameters needed from $O(d^4)$ to $O(d^2)$, the same as for methods making the independence assumption; the choice of ζ governs how these dependencies may interact.

[2]This is commonly cited as being the tensor product, i.e. 'a gene only affects genes in different cells'. This mistake has been made in all prior work to our knowledge, including our own. A Kronecker product only represents the tensor product if the diagonals of $\mathbf{\Psi}_i$ are zero - something that will not traditionally happen in our context as the resulting precision matrix is typically positive definite.

Decomposition method	Definition	Graph Product	Intuition
Kronecker sum	$\mathbf{\Psi}_1 \oplus \mathbf{\Psi}_2$	Cartesian product	A gene can only affect the expression of other genes within the same cell, or the same gene in differetn cells
Kronecker product	$\mathbf{\Psi}_1 \otimes \mathbf{\Psi}_2$	Strong product[2]	A gene has the exact same effect on the expression of genes within and between cells
The strong product model	$\mathbf{\Psi}_1 \oplus \mathbf{\Psi}_{2,w}$ $+\mathbf{\Psi}_1 \otimes \mathbf{\Psi}_{2,b}$	Strong product	A gene can affect other genes within the same cell, and can also have a different affect on genes in different cells.
Squared Kronecker sum	$(\mathbf{\Psi}_1 \oplus \mathbf{\Psi}_2)^2$	N/A	Arises when one constructs a stochastic model for the data based on partial differential equations of a simple form (Y. Wang & Hero, 2021).

Table 1.1: Precision matrix decompositions based on graph products in use in the literature; 'the strong product model' was introduced by us (B. Andrew et al., 2025). $\mathbf{\Psi}_{2,w}$ and $\mathbf{\Psi}_{2,b}$ are two different graphs describing the second axis.

In practice, ζ is a function that can be expressed in terms of Kronecker products $\otimes$, which are constructed out of repeated block matrices. The Kronecker product of a $d_1 \times d_2$ matrix and a $d_3 \times d_4$ matrix is a $d_1 d_3 \times d_2 d_4$ matrix; the definition is given below.

$$\begin{bmatrix} a_{11} & a_{12} & \cdots \\ a_{21} & \ddots & \\ \vdots & & \end{bmatrix} \otimes \mathbf{B} = \begin{bmatrix} a_{11}\mathbf{B} & a_{12}\mathbf{B} & \cdots \\ a_{21}\mathbf{B} & \ddots & \\ \vdots & & \end{bmatrix}$$

The choice of ζ corresponds to either a natural operation on the graph of conditional dependencies or a generative process. Table 1.1 gives a brief description of the main options, which we elaborate on in Section 1.4; for now we will focus on the Cartesian product, $\zeta\left(\mathbf{\Psi}^{(rows)}, \mathbf{\Psi}^{(cols)}\right) = \mathbf{\Psi}^{(rows)} \otimes \mathbf{I} + \mathbf{I} \otimes \mathbf{\Psi}^{(cols)}$. This operation is known as the 'Kronecker sum', $\mathbf{\Psi}^{(rows)} \oplus \mathbf{\Psi}^{(cols)}$; the adjacency matrix of the Cartesian product of two graphs is the Kronecker sum of the adjacency matrices of said graphs. This was first considered by Kalaitzis et al. 2013. Below, we show how to express the probability distributions resulting from various choices of ζ (including those that correspond to standard independence-assuming models).

$$\text{vec}\left[\mathbf{X}_{d_1 \times d_2}\right] \sim \mathcal{N}\left(\mathbf{0}_{d_1 d_2}, \left(\mathbf{\Psi}^{(genes)}_{d_2 \times d_2} \otimes \mathbf{I}_{d_1 \times d_1}\right)^{-1}\right) \quad \text{(Independence of cells)}$$

$$\text{vec}\left[\mathbf{X}_{d_1 \times d_2}\right] \sim \mathcal{N}\left(\mathbf{0}_{d_1 d_2}, \left(\mathbf{I}_{d_2 \times d_2} \otimes \mathbf{\Psi}^{(cells)}_{d_1 \times d_1}\right)^{-1}\right) \quad \text{(Independence of genes)}$$

$$\text{vec}\left[\mathbf{X}_{d_1 \times d_2}\right] \sim \mathcal{N}\left(\mathbf{0}_{d_1 d_2}, \left(\mathbf{\Omega}^{(matrix\ elements)}_{d_1 d_2 \times d_1 d_2}\right)^{-1}\right) \quad \text{(Full dependence)}$$

$$\text{vec}\left[\mathbf{X}_{d_1 \times d_2}\right] \sim \mathcal{N}\left(\mathbf{0}_{d_1 d_2}, \left(\mathbf{\Psi}^{(genes)}_{d_2 \times d_2} \otimes \mathbf{I}_{d_1 \times d_1} + \mathbf{I}_{d_2 \times d_2} \otimes \mathbf{\Psi}^{(cells)}_{d_1 \times d_1}\right)^{-1}\right) \quad \text{(Cartesian product)}$$

By using the Cartesian product, we are making an assumption. We describe this assumption graphically in Figure 1.2, as well as both conceptually and formally below:

Assumption 2 (Cartesian product assumption for scRNA-seq data)**.** *A gene g may only be conditionally dependent ($\sim$) on other genes g' in the same cell c, or the same gene in other cells c': $(c, g) \sim (c, g')$, $(c, g) \sim (c', g)$, and $(c, g) \nsim (c', g')$.*

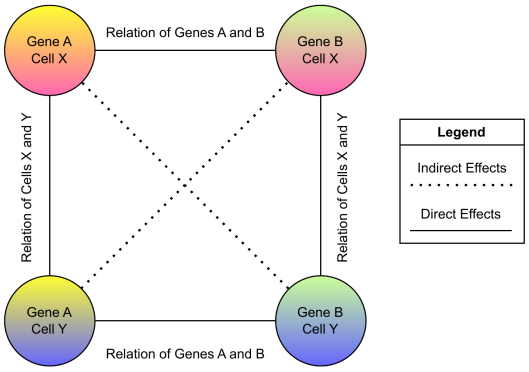


Figure 1.2: All possible interactions in a dataset with two cells and two genes. When using the Cartesian product, there are only two free parameters: the strength of the relationship between the two genes, and between the two cells. The relationship between (cell, gene) pairs is determined by these two values; the cross-term relations are indirect (not conditionally dependent, but likely correlated).

Assumption 3 (Formal Cartesian product assumption)**.** *Let $\mathfrak{C}$ be the true conditional dependency matrix of our data. Then we can decompose it as follows: $\mathfrak{C} = \mathfrak{F}^{(rows)} \boxplus \mathfrak{F}^{(cols)}$, where $\boxplus$ represents the Cartesian product and $\mathfrak{F}$ represent row- and column-level dependencies. In other words, letting $a \sim_{\mathfrak{X}} b$ mean that a and b are connected in $\mathfrak{X}$, we have that:*

$$(i_1, i_2) \sim_{\mathfrak{C}} (j_1, j_2) \iff i_1 \sim_{\mathfrak{F}^{(rows)}} j_1 \text{ and } i_2 \sim_{\mathfrak{F}^{(cols)}} j_2$$

This is a biologically plausible assumption; the strongest regulatory effects of genes are those occurring within a cell. That is not to say there are no regulatory effects crossing cells - there are: genes coding for signalling pathways likely have cross-cell effects, and we investigate such a scenario in Chapter 4. However, we found that accounting for them did not yield a better model. In the next section, we review the many options for multi-axis models.

1.4 A review of multi-axis models

The simplest multi-axis models are those which use the Kronecker product; this corresponds to the strong product of graphs, although it is commonly misreported as corresponding to the tensor product. It is in common use - see for example (Zhou, 2014) and (Cai et al., 2016) - and has a long history (DAWID, 1981). More recently, Kalaitzis et al. (2013) proposed the use of the Kronecker sum, representing the Cartesian product. This has a few key advantages over the Kronecker product: it arises from a maximum entropy construction and is jointly convex in the factor matrices.

Despite this, Kalaitzis et al.'s original algorithm ('BiGLasso') was far too slow to be used at appreciable scale; for $d \times d$ data, the runtime scales as $O(d^4)$, and in our experience could not handle much more than $d = 100$ in reasonable time. Their work kicked off a series of papers introducing faster algorithms: TeraLasso (Greenewald et al., 2019), EiGLasso (Yoon & Kim, 2020), scBiGLasso (Li et al., 2022), and DNNLasso (Lin & Zhang, 2024), as well as our own work on GmGM (B. Andrew et al., 2024; E. B. Andrew et al., 2024). However, until our work on GmGM none of the methodologies were scalable beyond even a thousand cells/genes. We will defer discussion of GmGM until it is introduced in Chapters 2 and 6, and consider here only the state of affairs before its introduction.

Of the prior work, TeraLasso was by far the most consistently scalable. It also generalized the methodology to work for tensor-variate data (data with more than two axes) as well. This type of data is rare in bioinformatics, but does occur (see Example Datasets 4 and 5). Such a generalization is conceptually simple, and remains jointly convex:

$$\operatorname{vec}\left[\mathbf{X}_{d_1\times\cdots\times d_L}\right] \sim \mathcal{N}\left(\mathbf{0}_{d_1\cdots d_L}, \left(\bigoplus_{1\le\ell\le L} \mathbf{\Psi}^{(axis\ \ell)}_{d_\ell\times d_\ell}\right)^{-1}\right)$$

More commonly needed in bioinformatics is the ability to work with multiple modalities simultaneously; however, this is not possible with TeraLasso.

An advantage of the Kronecker product approach not shared by the Kronecker sum approach is its ability to be naturally framed as a generative model. For example, we have that:

$$\begin{aligned} & \operatorname{vec}\left[\mathbf{E}\right] \sim \mathcal{N}(\mathbf{0}, \mathbf{I}) \\ \iff \quad & \operatorname{vec}\left[\mathbf{A}\mathbf{E}\mathbf{B}^T\right] \sim \mathcal{N}(\mathbf{0}, \mathbf{B}\mathbf{B}^T \otimes \mathbf{A}\mathbf{A}^T) \qquad \text{(Matrix normal distribution)} \end{aligned}$$

Thus the Kronecker product approach lends itself to a natural description in terms of the Cholesky factors or matrix square roots of the axis precision matrices; it is the corruption of Gaussian white noise when left- and right-multiplied by such factors. Y. Wang et al. (2020) introduced the SyGLasso algorithm built on a generative process involving the Kronecker sum. The final precision matrix is not the Kronecker sum, but rather the square of the Kroneker sum:

$$\begin{aligned} & \operatorname{vec}\left[\mathbf{E}\right] \sim \mathcal{N}(\mathbf{0}, \mathbf{I}) \\ \iff \quad & \operatorname{vec}\left[\mathbf{A}\mathbf{E} + \mathbf{E}\mathbf{B}^T\right] \sim \mathcal{N}(\mathbf{0}, (\mathbf{B}\oplus\mathbf{A})(\mathbf{B}\oplus\mathbf{A})^T) \quad \text{(Sylvester-structured normal distribution)} \end{aligned}$$

If the data is generated by a physical process with the form $\mathcal{D}\mathbf{x} = \mathbf{e}$, where $\mathbf{x}$ is the vectorized dataset, $\mathbf{e}$ is Gaussian white noise, and $\mathcal{D}$ is a differential operator with no mixed derivatives, then the data follows the Sylvester-structured normal distribution (Y. Wang & Hero, 2021). However, it does not have an easy interpretation as a graph product; the only methodology interpretable as both a generative process and a graph product is the Kronecker product.

The Kronecker product approach is applied to both covariance and precision matrices, whereas the other two decompositions are primarily applied to the precision matrix. There are reasons for this: we would not expect correlations to obey Cartesian product structure. The Cartesian product structure would enforce zero correlation on many edges, even though they would likely be correlated in reality due to indirect effects (if A is correlated with B, and B is correlated with C, then A will be correlated with C unless the correlations perfectly cancel each other out!).

There are also possibilities beyond covariance and precision matrices; one could apply these decompositions to the *Laplacian* of the graph of conditional dependencies (Shi & Mishne, 2024) or even a causal graph. Causal graph estimation is especially hard, as directed acyclic graph (DAG) optimization is NP-hard and it is sometimes only possible to learn a DAG up to an equivalence class. However, it is also quite promising: gene regulation is a fundamentally directed, causal process, whereas techniques based on conditional dependency matrices are undirected and non-causal. Prior to our work in Chapter 5, no Kronecker-structured causal method had been developed.

1.5 My work in context

Despite the work already done in multi-axis models, a few critical gaps remained at the start of this PhD. They were as follows:

- These methods were not scalable beyond about 1000 to 2000 cells/genes.
- These methods were not suitable for multi-modal datasets, such as multi-omics.
- These methods assumed that the data was centered (i.e. zero-mean)[3]

[3] Allen and Tibshirani (2010) did address this problem for the Kronecker product methodologies, but not for the Kronecker sum.

- There was no multi-axis causal method
- It was not clear if there existed a better structure than the Kronecker sum/Cartesian product.

The first gap was by far the most critical; modern scRNA-seq datasets can be millions of cells large. Furthermore, multi-axis methods perform better as the size of the dataset (in both samples and features) increases (Y. Wang et al., 2022). With the increasing use of paired datasets, such as scRNA-seq+scATAC-seq (Example Dataset 3), the need for multi-modal methods was also apparent. Chapters 2 and 6 address this gap.

The third gap was also significant; we could show that the centered assumption can lead to serious errors, even when applying the ad-hoc mitigation techniques that were being used in practice. This gap is addressed in Chapter 3 in the form of a wrapper that can be used around any of the existing Kronecker-sum-structured methods.

The lack of a causal multi-axis method is partially addressed in Chapter 5 through our introduction of the 'Cartesian-structured linear Gaussian additive noise model'. The method we develop is able to learn causal networks, but only under the strong assumption of *already knowing the causal order of events*. Thus, we have only partially closed this research gap.

In pursuit of addressing the last gap, we explored an alternative structure meant to capture the differences between within-cell and between-cell gene regulation (Chapter 4). We found that our novel structure did not yield substantial improvements beyond the Kronecker sum structure, leaving this gap still mostly open.

Bibliography for the introduction

Allen, G. I., & Tibshirani, R. (2010). Transposable regularized covariance models with an application to missing data imputation. *The Annals of Applied Statistics*, *4*(2), 764–790. https://doi.org/10.1214/09-AOAS314

Andrew, B., Westhead, D. R., & Cutillo, L. (2024, July). Making Multi-Axis Gaussian Graphical Models Scalable to Millions of Samples and Features [arXiv:2407.19892 [stat]]. https://doi.org/10.48550/arXiv.2407.19892

Andrew, B., Westhead, D. R., & Cutillo, L. (2025). The Strong Product Model for Network Inference without Independence Assumptions. *Proceedings of The 28th International Conference on Artificial Intelligence and Statistics*, 5230–5238. Retrieved January 14, 2026, from https://proceedings.mlr.press/v258/andrew25a.html

Andrew, E. B., Westhead, D., & Cutillo, L. (2024). GmGM: A fast multi-axis Gaussian graphical model. *Proceedings of The 27th International Conference on Artificial Intelligence and Statistics*, 2053–2061. Retrieved January 14, 2026, from https://proceedings.mlr.press/v238/andrew24a.html

Baumgart, M., Priebe, S., Groth, M., Hartmann, N., Menzel, U., Pandolfini, L., Koch, P., Felder, M., Ristow, M., Englert, C., Guthke, R., Platzer, M., & Cellerino, A. (2016). Longitudinal RNA-Seq Analysis of Vertebrate Aging Identifies Mitochondrial Complex I as a Small-Molecule-Sensitive Modifier of Lifespan. *Cell Systems*, *2*(2), 122–132. https://doi.org/10.1016/j.cels.2016.01.014

Bolognesi, M. M., Dall'Olio, L., Mandelli, G. E., Lorenzi, L., Bosisio, F. M., Haberman, A. M., Bhagat, G., Borghesi, S., Faretta, M., Castellani, G., & Cattoretti, G. (2025, October). The normal human lymph node cell classification and landscape defined by high-dimensional spatial proteomic [ISSN: 2692-8205 Pages: 2025.10.19.683341 Section: New Results]. https://doi.org/10.1101/2025.10.19.683341

Cai, T. T., Li, H., Liu, W., & Xie, J. (2016). Joint Estimation of Multiple High-Dimensional Precision Matrices. *Statistica Sinica*, *26*(2), 445–464. Retrieved February 26, 2026, from https://www.jstor.org/stable/24721282

Cilento, M. A., Sweeney, C. J., & Butler, L. M. (2024). Spatial transcriptomics in cancer research and potential clinical impact: A narrative review. *Journal of Cancer Research and Clinical Oncology*, *150*(6), 296. https://doi.org/10.1007/s00432-024-05816-0

DAWID, A. P. (1981). Some matrix-variate distribution theory: Notational considerations and a Bayesian application. *Biometrika*, *68*(1), 265–274. https://doi.org/10.1093/biomet/68.1.265

Friedman, J., Hastie, T., & Tibshirani, R. (2008). Sparse inverse covariance estimation with the graphical lasso [Place: Oxford, England]. *Biostatistics*, *9*(3), 432–441. https://doi.org/10.1093/biostatistics/kxm045

Greenewald, K., Zhou, S., & Hero, A., III. (2019). Tensor Graphical Lasso (TeraLasso). *Journal of the Royal Statistical Society Series B: Statistical Methodology*, *81*(5), 901–931. https://doi.org/10.1111/rssb.12339

Guo, X., Zhang, Y., Zheng, L., Zheng, C., Song, J., Zhang, Q., Kang, B., Liu, Z., Jin, L., Xing, R., Gao, R., Zhang, L., Dong, M., Hu, X., Ren, X., Kirchhoff, D., Roider, H. G., Yan, T., & Zhang, Z. (2018). Global characterization of T cells in non-small-cell lung cancer by single-cell sequencing. *Nature Medicine*, *24*(7), 978–985. https://doi.org/10.1038/s41591-018-0045-3

Kalaitzis, A., Lafferty, J., Lawrence, N. D., & Zhou, S. (2013). The Bigraphical Lasso. *Proceedings of the 30th International Conference on Machine Learning*, 1229–1237. Retrieved February 25, 2026, from https://proceedings.mlr.press/v28/kalaitzis13.html

Langfelder, P., & Horvath, S. (2008). WGCNA: An R package for weighted correlation network analysis. *BMC Bioinformatics*, *9*(1), 559. https://doi.org/10.1186/1471-2105-9-559

Li, S., López-García, M., Lawrence, N. D., & Cutillo, L. (2022). Two-way Sparse Network Inference for Count Data. *Proceedings of The 25th International Conference on Artificial Intelligence and Statistics*, 10924–10938. Retrieved February 26, 2026, from https://proceedings.mlr.press/v151/li22g.html

Lin, M., & Zhang, Y. (2024). DNNLasso: Scalable Graph Learning for Matrix-Variate Data. *Proceedings of The 27th International Conference on Artificial Intelligence and Statistics*, 316–324. Retrieved February 26, 2026, from https://proceedings.mlr.press/v238/lin24b.html

Liu, H., Han, F., Yuan, M., Lafferty, J., & Wasserman, L. (2012). The nonparanormal skeptic. *Proceedings of the 29th International Coference on International Conference on Machine Learning*, 1731–1738.

McInnes, L., Healy, J., & Melville, J. (2020, September). UMAP: Uniform Manifold Approximation and Projection for Dimension Reduction [arXiv:1802.03426 [stat]]. https://doi.org/10.48550/arXiv.1802.03426

Meinshausen, N., & Bühlmann, P. (2006). High-dimensional graphs and variable selection with the Lasso. *Annals of Statistics*, *34*(3). Retrieved January 26, 2026, from https://ora.ox.ac.uk/objects/uuid:b13c6f22-805b-4a80-b71d-ca59d4bcf815

Mosig, R. A., Lobl, M., Senturk, E., Shah, H., Cohen, S., Chudin, E., Fruscio, R., Marchini, S., D'Incalci, M., Sachidanandam, R., Dottino, P., & Martignetti, J. A. (2012). IGFBP-4 tumor and serum levels are increased across all stages of epithelial ovarian cancer. *Journal of Ovarian Research*, *5*(1), 3. https://doi.org/10.1186/1757-2215-5-3

Qiu, C., Lu, N., Wang, X., Zhang, Q., Yuan, C., Yan, S., Dongol, S., Li, Y., Sun, X., Sun, C., Zhang, Z., Zheng, W., & Kong, B. (2017). Gene expression profiles of ovarian low-grade serous carcinoma resemble those of fallopian tube epithelium. *Gynecologic Oncology*, *147*(3), 634–641. https://doi.org/10.1016/j.ygyno.2017.09.029

Regev, A., Teichmann, S. A., Lander, E. S., Amit, I., Benoist, C., Birney, E., Bodenmiller, B., Campbell, P., Carninci, P., Clatworthy, M., Clevers, H., Deplancke, B., Dunham, I., Eberwine, J., Eils, R., Enard, W., Farmer, A., Fugger, L., Göttgens, B., ... Human Cell Atlas Meeting Participants. (2017). The Human Cell Atlas. *eLife*, *6*, e27041. https://doi.org/10.7554/eLife.27041

Sheih, A., Voillet, V., Hanafi, L.-A., DeBerg, H. A., Yajima, M., Hawkins, R., Gersuk, V., Riddell, S. R., Maloney, D. G., Wohlfahrt, M. E., Pande, D., Enstrom, M. R., Kiem, H.-P., Adair, J. E., Gottardo, R., Linsley, P. S., & Turtle, C. J. (2020). Clonal kinetics and single-cell transcriptional profiling of CAR-T cells in patients undergoing CD19 CAR-T immunotherapy. *Nature Communications*, *11*(1), 219. https://doi.org/10.1038/s41467-019-13880-1

Shi, C., & Mishne, G. (2024). Learning Cartesian Product Graphs with Laplacian Constraints. *Proceedings of The 27th International Conference on Artificial Intelligence and Statistics*, 2521–2529. Retrieved February 26, 2026, from https://proceedings.mlr.press/v238/shi24a.html

Svensson, V., da Veiga Beltrame, E., & Pachter, L. (2020). A curated database reveals trends in single-cell transcriptomics. *Database*, *2020*, baaa073. https://doi.org/10.1093/database/baaa073

van Dam, S., Võsa, U., van der Graaf, A., Franke, L., & de Magalhães, J. P. (2017). Gene co-expression analysis for functional classification and gene–disease predictions. *Briefings in Bioinformatics*, *19*(4), 575–592. https://doi.org/10.1093/bib/bbw139

Wang, J., Adrianto, I., Subedi, K., Liu, T., Wu, X., Yi, Q., Loveless, I., Yin, C., Datta, I., Sant'Angelo, D. B., Kronenberg, M., Zhou, L., & Mi, Q.-S. (2023). Integrative scATAC-seq and scRNA-seq analyses map thymic iNKT cell development and identify Cbfβ for its commitment. *Cell Discovery*, *9*(1), 61. https://doi.org/10.1038/s41421-023-00547-x

Wang, J., Dean, D. C., Hornicek, F. J., Shi, H., & Duan, Z. (2019). RNA sequencing (RNA-Seq) and its application in ovarian cancer. *Gynecologic Oncology*, *152*(1), 194–201. https://doi.org/10.1016/j.ygyno.2018.10.002

Wang, L., Yu, X., Wu, C., Zhu, T., Wang, W., Zheng, X., & Jin, H. (2018). RNA sequencing-based longitudinal transcriptomic profiling gives novel insights into the disease mechanism of generalized pustular psoriasis. *BMC Medical Genomics*, *11*(1), 52. https://doi.org/10.1186/s12920-018-0369-3

Wang, Y., & Hero, A. (2021). SG-PALM: A Fast Physically Interpretable Tensor Graphical Model. *Proceedings of the 38th International Conference on Machine Learning*, 10783–10793. Retrieved February 26, 2026, from https://proceedings.mlr.press/v139/wang21k.html

Wang, Y., Jang, B., & Hero, A. (2020). The Sylvester Graphical Lasso (SyGlasso). *Proceedings of the Twenty Third International Conference on Artificial Intelligence and Statistics*, 1943–1953. Retrieved February 26, 2026, from https://proceedings.mlr.press/v108/wang20d.html

Wang, Y., Sun, Z., Song, D., & Hero, A. (2022). Kronecker-structured covariance models for multiway data. *Statistics Surveys*, *16*(none), 238–270. https://doi.org/10.1214/22-SS139

Xu, K., Zhang, W., Wang, C., Hu, L., Wang, R., Wang, C., Tang, L., Zhou, G., Zou, B., Xie, H., Tang, J., & Guan, X. (2021). Integrative analyses of scRNA-seq and scATAC-seq reveal CXCL14 as a key regulator of lymph node metastasis in breast cancer. *Human Molecular Genetics*, *30*(5), 370–380. https://doi.org/10.1093/hmg/ddab042

Yoon, J. H., & Kim, S. (2020). EiGLasso: Scalable Estimation of Cartesian Product of Sparse Inverse Covariance Matrices. *Proceedings of the 36th Conference on Uncertainty in Artificial Intelligence (UAI)*, 1248–1257. Retrieved February 26, 2026, from https://proceedings.mlr.press/v124/ho-yoon20a.html

Zhou, S. (2014). Gemini: Graph Estimation with Matrix Variate Normal Instances. *The Annals of Statistics*, *42*(2), 532–562. Retrieved February 26, 2026, from https://www.jstor.org/stable/43556295

Chapter 2

GmGM: a fast multi-axis Gaussian graphical model

The following paper was the first of two papers we wrote on improving the scalability of Kronecker-sum-structured multi-axis models. The state-of-the-art before this paper was the TeraLasso algorithm by Greenewald et al. (2019), which was only scalable to matrices whose dimensions were in the low thousands. Also state-of-the-art was the EiGLasso algorithm of Yoon and Kim (2020), which narrowly eked out a performance advantage on matrix-variate data but was not applicable to tensor-variate data. Both were built on similar ideas of using eigendecompositions in a clever way to speed up the optimization of an L1-penalized maximum likelihood estimation routine.

In the paper, we showed that one could achieve massive scalability improvements by doing away with the L1 penalty at only a modest cost to performance. Whereas prior work's scalability was limited by runtime, ultimately our model's scalability would primarily be limited by memory usage, which we later address in Chapter 6. We also generalized the concept of multi-axis models to work on multi-tensor datasets, allowing the approach to be used on multi-omics datasets.

On a personal note, this was my first paper, and as such I am fond of it. However, there are several deficiencies in the work. Firstly, it does not go far enough in demonstrating a real use case, rather relying on a handful of shallow examples to validate performance. Secondly, it does not address the problem of whether a solution to the optimization problem considered exists - in practice it does, but the paper stops short of guaranteeing it. There was also a clear future direction to work towards: while scalable to tens of thousands of cells, our model at the time was not yet scalable to millions - the size of the largest modern datasets. These issues would later be addressed in our second paper on scalability (Chapter 6).

The paper was published in the Proceedings of the International Conference on Artificial Intelligence and Statistics in 2024. It is presented unchanged from the published version, except for formatting changes to fit the thesis style. The supplementary material for the paper is available in Appendix A, which contains further details and experiments. For further details and experiments that came up during the viva, see Appendix G.

GmGM: a Fast Multi-Axis Gaussian Graphical Model

Bailey Andrew, David R. Westhead, and Luisa Cutillo

Published in the Proceedings of the 27th International Conference on Artificial Intelligence and Statistics

Abstract. This paper introduces the Gaussian multi-Graphical Model, a model to construct sparse graph representations of matrix- and tensor-variate data. We generalize prior work in this area by simultaneously learning this representation across several tensors that share axes, which is necessary to allow the analysis of multimodal datasets such as those encountered in multi-omics. Our algorithm uses only a single eigendecomposition per axis, achieving an order of magnitude speedup over prior work in the ungeneralized case. This allows the use of our methodology on large multi-modal datasets such as single-cell multi-omics data, which was challenging with previous approaches. We validate our model on synthetic data and five real-world datasets.

2.1 Introduction

A number of modern applications require the estimation of networks (graphs) exploring the dependency structures underlying the data. In this paper, we propose a new approach for estimating conditional dependency graphs. 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: $\mathbb{P}(x|y, \mathcal{D}_{\setminus xy}) = \mathbb{P}(x|\mathcal{D}_{\setminus xy})$. 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 not zero. If our dataset was in the form of a vector $\mathbf{d}$, we could then model it as $\mathbf{d} \sim \mathcal{N}(\mathbf{0}, \mathbf{\Psi}^{-1})$ for precision matrix $\mathbf{\Psi}$. This is a Gaussian Graphical Model (GGM); $\mathbf{\Psi}$ encodes the adjacency matrix of the graph.

However, datasets are often more structured than vectors. For example, single-cell RNA sequencing datasets (scRNA-seq) come 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. Furthermore, multi-omics datasets, such as those including both scRNA-seq and scATAC-seq, may require two or more matrices to be properly represented; one for each modality.

We could assume that each row of our matrix is an i.i.d. sample drawn from our model. However, independence is a strong and often incorrect assumption. If we wanted to make no independence assumptions, we could vectorize the dataset $\mathcal{D}$ and estimate $\mathbf{\Psi}$ in $\text{vec}[\mathcal{D}] \sim \mathcal{N}(\mathbf{0}, \mathbf{\Psi}^{-1})$. Unfortunately, this produces intractably large $\mathbf{\Psi}$, whose number of elements is quadratic in the product of the lengths of our dataset's axes: the size of $\mathbf{\Psi}$ is $O\left(\prod_\ell d_\ell^2\right)$ (throughout this paper, d_ℓ will denote the size of the ℓth axis of a tensor).

Thankfully, tensors are highly structured, and we are often interested in the dependency structure of each axis individually - i.e. the dependencies between cells and the dependencies between genes, in the case of scRNA-seq - rather than the dependencies between the elements of the tensor themselves. To model this, we can represent $\mathbf{\Psi}$ as some deterministic combination of the axis-wise dependencies: $\text{vec}[\mathbf{D}] \sim \mathcal{N}(\mathbf{0}, \zeta(\mathbf{\Psi}_{\text{row}}, \mathbf{\Psi}_{\text{col}})^{-1})$, for some function ζ. The strategy is to estimate $\mathbf{\Psi}_{\text{row}}$ and $\mathbf{\Psi}_{\text{col}}$ directly, without computing the intractable $\zeta(\mathbf{\Psi}_{\text{row}}, \mathbf{\Psi}_{\text{col}})^{-1}$. While there are multiple choices for ζ, this paper considers only the Kronecker sum. Choices of ζ, and their definitions, will be given in Section 2.2.

2.1.1 Notation

When possible, this paper follows the notation of prior work in multi-axis methods, such as Kalaitzis et al., 2013 and Greenewald et al., 2019. For tensor-specific notation, we typically adhere to Kolda and Bader, 2009. Some changes have been made due to inconsistencies or conflicts.

The subscript ℓ will always be used to denote an arbitrary axis. d_ℓ and $\mathbf{\Psi}_\ell$ represent the size and precision matrix for the axis ℓ, respectively - as in prior work. It will be helpful to define the symbols $d_{<\ell}$, $d_{>\ell}$, $d_{\setminus\ell}$, and $d_\forall$ to represent the product of the lengths of all axes before ℓ, after ℓ, excluding ℓ, and overall, respectively.

Lowercase bold represents vectors ($\mathbf{d}$), uppercase bold represents matrices ($\mathbf{D}$), and uppercase calligraphic represents tensors ($\mathcal{D}$). We will also be dealing with sets of tensors, $\{\mathcal{D}^\gamma\}_\gamma$; each individual tensor is called a modality, and the superscript γ will always represent indexing a modality. The number of axes of a tensor $\mathcal{D}^\gamma$ is denoted K^γ, and to describe that an axis ℓ is one of the axes of a tensor $\mathcal{D}^\gamma$, we will write $\ell \in \gamma$. Axes can be part of several modalities. An important tensor-variate operation is matricization, which takes a $d_1 \times \ldots \times d_K$ tensor $\mathcal{D}$ and outputs a $d_\ell \times d_{\setminus\ell}$ matrix. It is denoted $\text{mat}_\ell[\mathcal{D}]$, and can be viewed as vectorizing each slice of $\mathcal{D}$ along the ℓth axis.

We will make frequent use of the Kronecker sum, defined in terms of the Kronecker product. The Kronecker product of two matrices $\mathbf{\Psi}_{\ell_1}, \mathbf{\Psi}_{\ell_2}$ is denoted $\mathbf{\Psi}_{\ell_1} \otimes \mathbf{\Psi}_{\ell_2}$, and is best defined by example;

$$\begin{bmatrix} 1 & 2 \\ 3 & 4 \end{bmatrix} \otimes \begin{bmatrix} 5 & 6 \end{bmatrix} = \begin{bmatrix} 1 \times \begin{bmatrix} 5 & 6 \end{bmatrix} & 2 \times \begin{bmatrix} 5 & 6 \end{bmatrix} \\ 3 \times \begin{bmatrix} 5 & 6 \end{bmatrix} & 4 \times \begin{bmatrix} 5 & 6 \end{bmatrix} \end{bmatrix}$$
$$= \begin{bmatrix} 5 & 6 & 10 & 12 \\ 15 & 18 & 20 & 24 \end{bmatrix}$$

The Kronecker sum is $\mathbf{\Psi}_{\ell_1} \oplus \mathbf{\Psi}_{\ell_2} = \mathbf{\Psi}_{\ell_1} \otimes \mathbf{I}_{d_{\ell_2} \times d_{\ell_2}} + \mathbf{I}_{d_{\ell_1} \times d_{\ell_1}} \otimes \mathbf{\Psi}_{\ell_2}$. 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.

The "blockwise trace" operation defined by Kalaitzis et al. (2013) appears when calculating the gradient of the log-likelihood of KS-structured normal distributions (and is strongly related to the proj_{KS} operator in Greenewald et al., 2019). Let $\mathbf{I}_a$ be the $a \times a$ identity matrix, and let $\mathbf{J}^{ij}$ be a matrix of zeros except at (i, j), where it equals 1. Then, we denote the blockwise trace of a matrix tr_b^a and define it according to Line 2.1.

$$\text{tr}_b^a \left[\mathbf{M}\right] = \left[\text{tr}\left[\mathbf{M}\left(\mathbf{I}_a \otimes \mathbf{J}^{ij} \otimes \mathbf{I}_b\right)\right]\right]_{ij} \tag{2.1}$$

The final needed concept is that of the Gram matrix $\mathbf{S}_\ell^\gamma$, defined as $\mathbf{S}_\ell^\gamma = \text{mat}_\ell \left[\mathcal{D}^\gamma\right] \text{mat}_\ell \left[\mathcal{D}^\gamma\right]^T$. In the single-tensor (multi-axis) case, this is a sufficient statistic; all prior work first computes these matrices as the initial step in their algorithm, as do we. When there are multiple modalities, we consider the 'effective Gram matrices' $\mathbf{S}_\ell = \sum_{\gamma | \ell \in \gamma} \mathbf{S}_\ell^\gamma$, as these fulfill the role of sufficient statistic in the multi-tensor case.

2.2 Prior work

The Graphical Lasso (Friedman et al., 2008) is the standard single-axis GGM. The model assumes one has n independent samples of d_1-length vectors, and seeks to estimate the $d_1 \times d_1$ precision matrix $\mathbf{\Psi}$. $\mathbf{\Psi}$ is estimated via Equation 2.2.

$$\mathbf{\Psi} = \text{argmax}_{\mathbf{\Psi} \succ 0} \log\det \mathbf{\Psi} - \text{tr}\left[\mathbf{S}\mathbf{\Psi}\right] + \rho \|\mathbf{\Psi}\|_1 \tag{2.2}$$

Without the regularizing $\rho\|\mathbf{\Psi}\|_1$ term, this represents the maximum likelihood estimate for $\mathbf{\Psi}$; $\mathbf{\Psi} \succ 0$ ensures positive definiteness. The regularization penalty is an L1 penalty over the elements of $\mathbf{\Psi}$ - typically, the diagonal elements are not included in this penalty, as they do not represent edges on the conditional dependency graph.

Our work considers the Kronecker-sum (multi-axis) Graphical Lasso, which was introduced by Kalaitzis et al. (2013). This sum is one choice of ζ to combine the per-axis precision matrices into the precision matrix of the vectorized dataset, $\text{vec}[\mathbf{D}] \sim \mathcal{N}(\mathbf{0}, (\mathbf{\Psi}_{\text{row}} \oplus \mathbf{\Psi}_{\text{col}})^{-1})$. Other choices for ζ have been considered, such as using the Kronecker product (Dahl et al., 2013; Tsiligkaridis et al., 2013), or the square of the Kronecker sum (Wang & Hero, 2021; Wang et al., 2020). Each method has its strengths; the benefits of a Kronecker sum structure are its interpretability as a graph product, stronger sparsity, convexity of the maximum likelihood, and its allowance of inter-task transfer.

Kalaitzis et al. (2013)'s BiGLasso model only worked for matrix-variate data, and was very slow to converge to a solution, in large part due to its non-optimal space complexity of $O(d_1^2 d_2^2)$. This prohibited its use on moderately sized datasets. Numerous modifications have been made to the algorithm to improve its speed and to achieve an optimal space complexity of $O(d_1^2 + d_2^2)$, such as scBiGLasso (Li et al., 2022), TeraLasso (Greenewald et al., 2019), and EiGLasso (Yoon & Kim, 2020). Of these TeraLasso is notable in that it allows an arbitrary number of axes, i.e. $\zeta(\mathbf{\Psi}_1, ..., \mathbf{\Psi}_k) = \mathbf{\Psi}_1 \oplus ... \oplus \mathbf{\Psi}_k$, like our proposed method. TeraLasso and EiGLasso, the fastest prior algorithms, both rely on computing an eigendecomposition every iteration.

All of these models, including our own, rely on a normality assumption. We are most interested in the case of omics data, which is typically non-Gaussian; an overview of the use of GGMs in omics data is given by Altenbuchinger et al. (2020). For single-axis methods, the nonparanormal skeptic (Liu et al., 2012) can replace the normality assumption with a weaker nonparanormality assumption; Li et al. (2022) proposed a generalization of this approach to multi-axis methods, which is compatible with our approach. The nonparanormal skeptic often leads to better results on real datasets, for example see Figure 2.5a. For a thorough comparison of the nonparanormal skeptic, see the supplementary material.

In the single-axis case, several techniques exist for combining datasets with shared axes (Cai et al., 2016; Danaher et al., 2014; McCarter & Kim, 2014). However, none of these have been applied to the multi-axis case.

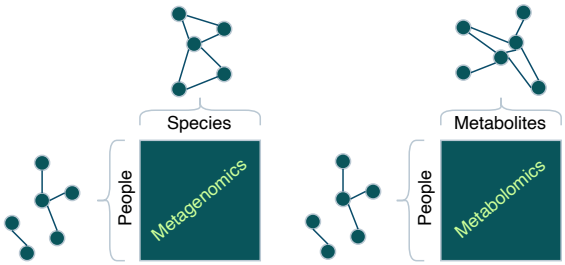


Figure 2.1: The two matrices of the LifeLines-DEEP dataset. As both matrices include data for the same people, the learned graph between people should be the same.

2.2.1 Unmet need

Many datasets, especially those in multi-omics, are representable as a collection of matrices or tensors. As a case study, we consider (a subset of) the Lifelines-DEEP dataset from Tigchelaar et al. (2015), which is summarized graphically in Figure 2.1.

In this dataset, two different modalities of data were gathered from the same people: counts of microbial species found in their stools (metagenomics) and counts of metabolites found in their blood plasma (metabolomics). While the metagenomics and metabolomics are different matrices, each modality shares an axis ('people'). If we were to estimate a graph of people on each modality independently, they would likely yield different graphs. This is not ideal; if our aim is to estimate the true graph of conditional dependencies, there should be only one resultant graph. To estimate it, we should be considering both modalities simultaneously to take advantage of the shared axis.

One way to do this would be to concatenate the modalities, producing a matrix of people by 'species+metabolites'. This could yield interesting results, if one is interested in connections between individual species and a metabolite. However, it would vastly increase the size of the output graph, which grows quadratically in the length of the axis. Furthermore, it is not always possible; some datasets may not be concatenatable.

2.3 Our contributions

We make three contributions to the field of multi-axis graphical models. Firstly, we produce an algorithm to estimate multi-axis graphical models an order of magnitude faster than existing work. This allows us to apply our algorithm on large datasets, such as single-cell RNA-sequencing datasets which include roughly 20,000 genes and tens of thousands of cells. Prior work could not handle these datasets in reasonable time - ours is the only method able to produce multi-axis estimates for data of this size, and runs comparably quickly to single-axis methods.

Secondly, we generalize the multi-axis model from prior work, which was only able to consider unimodal data, into a model that can handle multiple tensors with shared axes. This generalization is essential to the use of multi-axis methods on multi-modal data such as multi-omics; no multi-axis method had been made for this type of data before our work.

Finally, we produce a few helpful theorems to improve accuracy and further improve speed. A covariance-thresholding technique by Mazumder and Hastie (2012) allowed the partitioning of data into guaranteed-independent chunks of data in the single-axis case; we show how to lift this result to the multi-axis case, and demonstrate its efficacy (Figure 2.7). We also show how to efficiently include some natural prior distributions into our model, which allows us to demonstrate an extraordinary increase in performance on real-world data (Figure 2.6c).

2.3.1 The model

To properly handle sets of tensors, we propose modelling each tensor as being drawn independently from a Kronecker-sum normal distribution. If the tensors share an axis ℓ, then they will still be drawn independently - but their distributions will be parameterized by the same $\mathbf{\Psi}_\ell$. For an arbitrary set of tensors, the model is:

$$\mathcal{D}^\gamma \sim \mathcal{N}_{KS}\left(\{\mathbf{\Psi}_\ell\}_{\ell\in\gamma}\right)$$
$$\text{for } \mathcal{D}^\gamma \in \{\mathcal{D}^\gamma\}_\gamma$$

We call this model the "Gaussian multi-Graphical Model" (GmGM) as it extends Gaussian Graphical Models to estimate multiple graphs from a set of tensors. In this paper, we will make the assumption that no tensor in our set contains the same axis twice. Any tensor with a repeated axis would naturally be interpretable as a graph - such datasets are rare, and if one already has a graph then the need for an algorithm such as this is diminished.

As an example, we model the LifeLines-DEEP dataset $\mathbf{D}^{\text{metagenomics}}$ and $\mathbf{D}^{\text{metabolomics}}$ independently as:

$$\mathbf{D}^{\text{metagenomics}} \sim \mathcal{N}_{KS}\left(\mathbf{\Psi}^{\text{people}}, \mathbf{\Psi}^{\text{species}}\right)$$
$$\mathbf{D}^{\text{metabolomics}} \sim \mathcal{N}_{KS}\left(\mathbf{\Psi}^{\text{people}}, \mathbf{\Psi}^{\text{metabolites}}\right)$$

2.3.2 The algorithm

Table 2.1: A comparison of the asymptotic complexity of TeraLasso and GmGM. EiGLasso is omitted because it has the same complexity as TeraLasso. N represents the number of iterations before convergence. "Simplified Time Complexity" represents the case when all axes have the same size d, there is only one modality, and the dataset only has two axes.

Algorithm	Simplified Time Complexity	Full Time Complexity	Memory Complexity
TeraLasso	$O(Nd^3)$	$O(\sum_\ell d_\ell^2 d_{\setminus\ell} + Nd_\ell^3 + N\prod_\ell d_\ell)$	$O(\sum_\ell d_\ell^2)$
GmGM	$O(d^3 + Nd^2)$	$O(\sum_\ell d_\ell^2 d_{\setminus\ell} + d_\ell^3 + \sum_\gamma N\prod_{\ell\in\gamma} d_\ell)$	$O(\sum_\ell d_\ell^2)$

Here, we present an algorithm to compute the maximum likelihood estimate (MLE) jointly for all parameters $\mathbf{\Psi}_\ell$ of the GmGM. The general idea is to produce an analytic estimate for the eigenvectors of $\mathbf{\Psi}_\ell$, and then iterate to solve for the eigenvalues; this is summed up graphically in Figure 2.2.

Theorem 1. *Let* $\mathbf{V}_\ell \text{diag}\left[\mathbf{e}_\ell\right] \mathbf{V}_\ell^T$ *be the eigendecomposition of* $\mathbf{S}_\ell$ *(where* $\mathbf{V}_\ell \in \mathbb{R}^{d_\ell \times d_\ell}$ *and* $\text{diag}\left[\mathbf{e}_\ell\right] \in \mathbb{R}^{d_\ell \times d_\ell}$ *is a diagonal matrix with diagonal* $\mathbf{e}_\ell$*). Then* $\mathbf{V}_\ell$ *are the eigenvectors of the maximum likelihood estimate of* $\mathbf{\Psi}_\ell$*.*

Theorem G is critical to allowing efficient estimation of $\mathbf{\Psi}_\ell$, as it not only allows us to extract the computationally intensive eigendecomposition operation from the iterative portion of the algorithm, but also reduces the remaining number of parameters to be linear in the length of an axis.

It is now necessary to find estimate the eigenvalues $\mathbf{\Lambda}_\ell = \text{diag}\left[\boldsymbol{\lambda}_\ell\right]$. We will do this iteratively, which requires the gradient of the negative log-likelihood with respect to the eigenvalues. A sketch of the derivation of this is given.

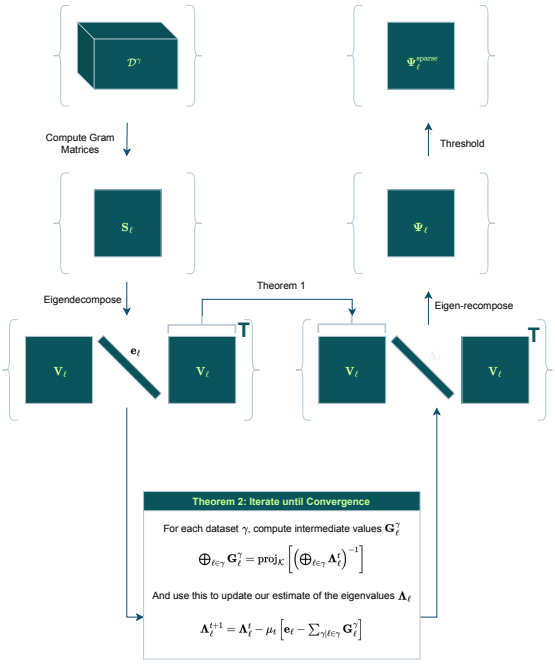


Figure 2.2: A graphical overview of how the GmGM algorithm works. We use γ to represent an arbitrary modality, and ℓ to represent an arbitrary axis. Proofs are given in the supplementary material.

$$\frac{\partial \text{NLL}}{\partial \lambda_\ell^{(i)}} = \frac{1}{2}\frac{\partial}{\partial \lambda_\ell^{(i)}}\left(\text{tr}\left[\mathbf{S}_\ell \mathbf{V}_\ell \mathbf{\Lambda}_\ell \mathbf{V}_\ell^T\right] - \sum_\gamma \log\left|\bigoplus_{\ell' \in \gamma} \mathbf{\Lambda}_{\ell'}\right|\right) \tag{2.3}$$

$$= \frac{1}{2}\frac{\partial}{\partial \lambda_\ell^{(i)}}\mathbf{e}_\ell \boldsymbol{\lambda}_\ell - \frac{1}{2}\frac{\partial}{\partial \lambda_{\ell,(i)}}\sum_{\gamma,j}\log\left[\bigoplus_{\ell' \in \gamma}\mathbf{\Lambda}_{\ell'}\right]_{jj} \tag{2.4}$$

$$= \frac{1}{2}\mathbf{e}_{\ell,(i)} - \frac{1}{2}\sum_{\gamma,j}\frac{\left[\mathbf{I}_{d_{<\ell}} \otimes \mathbf{J}_{ii} \otimes \mathbf{I}_{d_{>\ell}}\right]_{jj}}{\left[\bigoplus_{\ell' \in \gamma}\mathbf{\Lambda}_{\ell'}\right]_{jj}} \tag{2.5}$$

$$\frac{\partial \text{NLL}}{\partial \boldsymbol{\lambda}_\ell} = \frac{1}{2}\mathbf{e}_\ell - \frac{1}{2}\sum_\gamma \text{tr}_{d_{>\ell}}^{d_{<\ell}}\left[\left(\bigoplus_{\ell' \in \gamma}\mathbf{\Lambda}_{\ell'}\right)^{-1}\right] \tag{2.6}$$

Line 2.6 follows from the definition of the blockwise trace; the rest use Theorem G and standard algebra.

An L1 penalty is often included in graphical models to enforce sparsity. To enforce sparsity in our model, there are a few options; one can choose to either keep the top $p\%$ of edges, or keep the top k edges per vertex. In Section 2.4, we see that this technique performs well. Including regularization in our algorithm is nontrivial, as we rely on analytic estimates of the eigenvectors. However, by utilizing a restricted form of the L1 penalty, we can bypass this limitation. For details on how to do so, see Section 5 in the supplementary material - we will refer to GmGM equipped with this penalty as 'GmGM L1'. In short, the restricted L1 penalty is an L1 penalty on the off-diagonals of a matrix (i.e. the Graphical Lasso) that has given (fixed) eigenvectors (since the iterative portion of our algorithm only iterates over the eigenvalues). We incorporate it with subgradient descent.

2.3.3 Additional theorems

We can incorporate priors into our algorithm as well. This is nontrivial, as we want it to be in a manner compatible with Theorem G.

Theorem 2 (GmGM Estimator with Priors)**.** *Suppose we have the same setup as in Theorem G, and that we have a prior of the form:*

$$\prod_{\ell} g_\ell(\mathbf{\Theta}_\ell) e^{\mathrm{tr}\left[\eta_\ell(\mathbf{\Theta}_\ell)^T \mathbf{\Psi}_\ell\right] + h_\ell(\mathbf{\Psi}_\ell)} \tag{2.7}$$

In other words, our prior is an exponential distribution for $\mathbf{\Psi}_\ell$, in which $\mathbf{\Psi}_\ell$ is the sufficient statistic. Then, if h_ℓ depends only on the eigenvalues (i.e. it is 'unitarily invariant'), the eigenvectors of $\frac{1}{2}\mathbf{S}_\ell - \eta_\ell(\mathbf{\Theta}_\ell)$ are the eigenvectors $\mathbf{V}_\ell$ of the MAP estimate for $\mathbf{\Psi}_\ell$. Furthermore, if h_ℓ is convex, then a result by Lewis (1996) allows us perform gradient descent for the eigenvalues in an almost-unaltered way. (see supplementary material).

$\frac{1}{2}\mathbf{S}_\ell - \eta_\ell(\mathbf{\Theta}_\ell)$ can be thought of as our "priorized" effective Gram matrix; its eigenvalues and eigenvectors play the same role as those of $\mathbf{S}_\ell$ in the original algorithm. The required property of unitary invariance is very common - both the Wishart (arguably the most natural prior to use in this context) and matrix gamma distributions satisfy Theorem 2. The benefits of including priors are demonstrated in Figure 2.6c.

We also show that the covariance thresholding trick (Mazumder & Hastie, 2012) works in the multi-axis case. This allows us to partition the dataset into guaranteed-to-be-disconnected subsets before using our algorithm (although we do not make use of this theorem when comparing performance to other work, as it equally benefits all methods). The benefits of using this trick are shown in Figure 2.7.

Theorem 3. *Set all elements of $\mathbf{S}_\ell$ whose absolute value is less than ρ to 0. This encodes a potentially disconnected graph. Likewise, consider $\mathbf{\Psi}_\ell$ to be the estimated precision matrix for our model equipped with an L1 penalty of strength ρ. This also encodes a potentially disconnected graph. If we label the vertices by which disconnected component they are part of, then this labeling is the same in both procedures (the procedure with $\mathbf{S}_\ell$ and the procedure with $\mathbf{\Psi}_\ell$).*

2.3.4 Convexity and uniqueness

As can be observed in Line 2.4, the negative log-likelihood, as a function of the eigenvalues, can be expressed as the sum of a dot product (an affine, and hence convex, function) and negative logarithms, which are also convex. Thus, the objective is convex.

The Kronecker-sum model, without additional restrictions, has non-identifiable diagonals. Greenewald et al. (2019) show that this can be solved by projecting the gradients at each iteration. Letting $\mathbf{A}_\ell = \mathrm{tr}\left[\bigoplus_{\ell \in \gamma} \mathbf{\Lambda}_\ell\right]$, their projection is defined according to Line 2.8. See Sections D and I.3 in their paper for thorough information. We apply this projection to our algorithm as well.

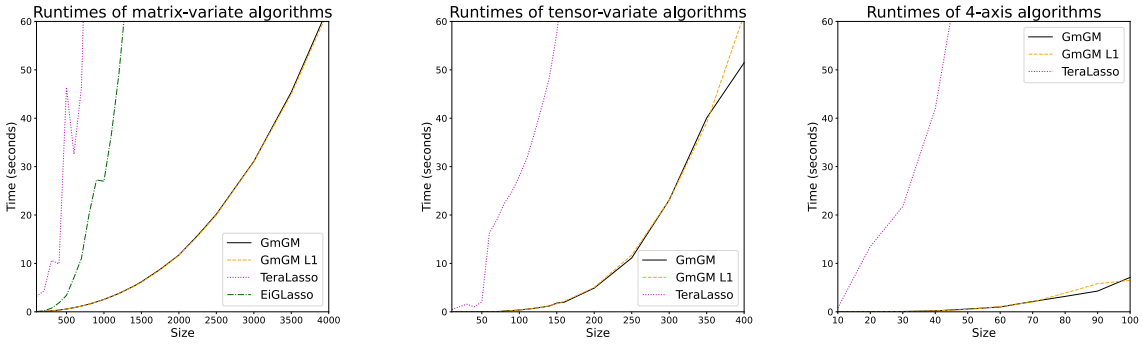
$$\mathrm{proj}_{\mathcal{K}} = \mathbf{A}_\ell - \frac{K_\gamma - 1}{K_\gamma} \frac{\mathrm{tr}\left[\mathbf{A}_\ell\right]}{d_\ell} \mathbf{I}_{d_\ell} \tag{2.8}$$

2.4 Results

We tested our algorithm on synthetic data and five real-world datasets. For synthetic data, we generated Erdos-Renyi graphs as our ground truth precision matrices for each axis (except for Figure 2.4b which uses an AR(1) process). Datasets were then generated from the Kronecker-sum normal distribution using these precision matrices. In all cases, we only generated 1 sample from this distribution. For comparison, we set EiGLasso's K parameter to 1; this parameter corresponds to the fastest version of EiGLasso. We used the same convergence tolerance and max iterations for all algorithms. Time and memory complexity for GmGM, TeraLasso, and EiGLasso are given in Table 2.1. When measuring runtimes on synthetic data, we let the regularization parameters be zero for prior work.

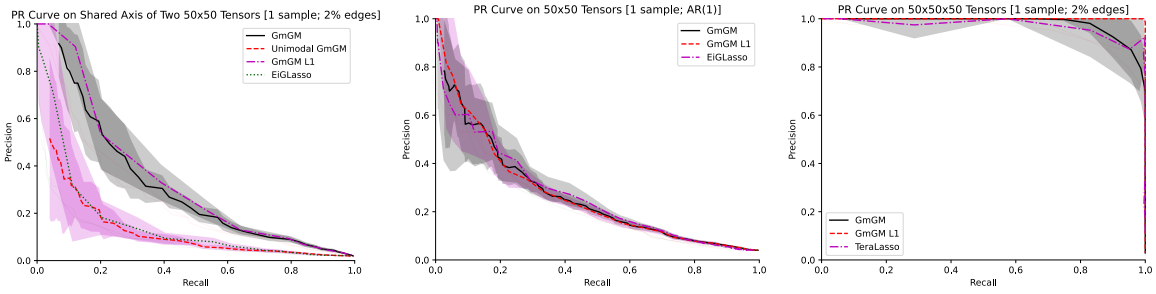
2.4.1 Synthetic data

We verified that our algorithm was indeed faster on matrix-variate data compared to prior work (Figure 2.3) on our computer (2020 MacBook Pro with an M1 chip and 8GB RAM). Our results on matrix



(a) (b) (c)

Figure 2.3: A comparison of the runtimes of our algorithm against (a) bi-graphical, (b) 3-axis, and (c) 4-axis prior work.



(a) (b) (c)

Figure 2.4: Precision-recall curves comparing various algorithms on synthetic 50x50 data, averaged over multiple runs. For (a) and (c), true graphs have each edge independently having a 2% chance of existing. For (b), true graphs are generated from an AR(1) process. Shaded background represents standard deviation. (a) Two 50x50 matrices with a shared axis. EiGLasso and 'Unimodal GmGM' only consider one of the matrices. (b) A single 50x50 matrix. (c) A single 50x50x50 tensor. With the restricted L1 penalty, our algorithm performs nearly perfectly in the tensor-variate case.

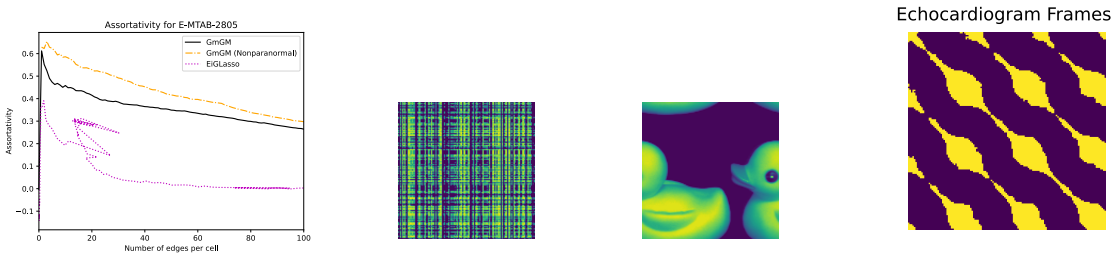
data are encouraging - datasets of size 4,000 by 4,000 could have their graphs estimated in a minute, and, extrapolating the runtimes, datasets up to size 16,000 by 16,000 could have their graphs estimated in about an hour. This is significantly faster than our baselines; neither EiGLasso nor TeraLasso could compute on 1,500 by 1,500 datasets in a minute. Larger datasets would require more than 6GB of memory for our algorithm to run, pushing the limits of RAM. On higher-order tensor data, our algorithm continues to outperform prior work. For 4-axis and higher data, our algorithm could handle every dataset that could fit in RAM in less than a minute (Fig 2.3c and supplementary material).

In addition to these speed improvements, we show that we perform similarly to state-of-the-art (Figure 2.4). We demonstrate that taking into account shared axes does indeed improve performance (Figure 2.4a), and furthermore that our restricted L1 penalty leads to near-perfect performance on tensor data (Figure 2.4c).

2.4.2 Real data

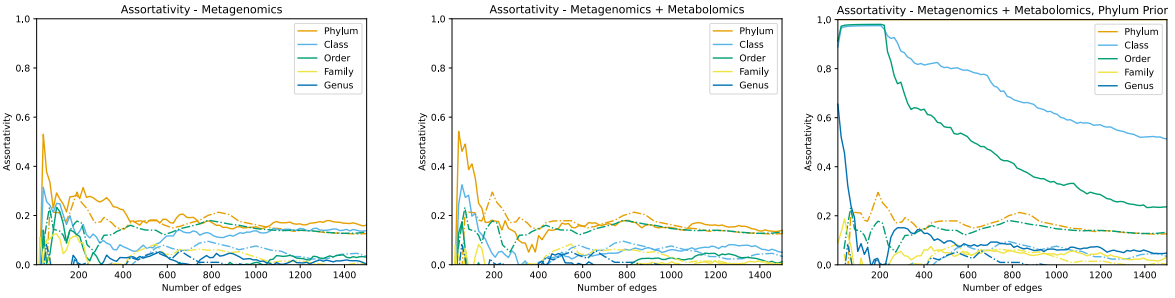
We tested our method on various real datasets. These include two video datasets (COIL-20 (Nene et al., n.d.) and EchoNet-Dynamic (Ouyang et al., 2020)), a transcriptomics dataset (E-MTAB-2805 (Buettner et al., 2015)), and two multi-omics datasets (LifeLines-DEEP (Tigchelaar et al., 2015) and a 10x Genomics dataset (10x Genomics, 2021)). Due to space concerns, we only briefly describe each experiment and its results here. Full details are available in the supplementary material.

For the COIL-20 dataset, we tested whether our algorithm could be used to reconstruct the frames



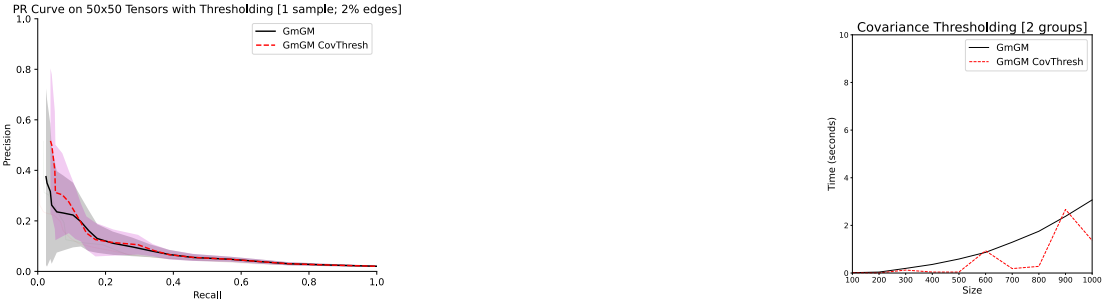
(a) (b) (c) (d)

Figure 2.5: (a) Assortativity as we increase the number of edges in the final graph. EiGLasso data was generated by calculating the assortativity-per-regularization penalty, and calculating the edges-per-cell from this data afterwards; this is the cause for the curve's strange behavior around 20 edges per cell. (b) The COIL-20 duck, in which the rows, columns and frames are shuffled. (c) GmGM does an almost perfect job at recovering the duck; unfortunately, it gets cut in half. (d) Predicted precision matrix for echocardiogram frames (yellow=connected, blue=disconnected). The periodic structure of the heartbeat is evident in the diagonals.



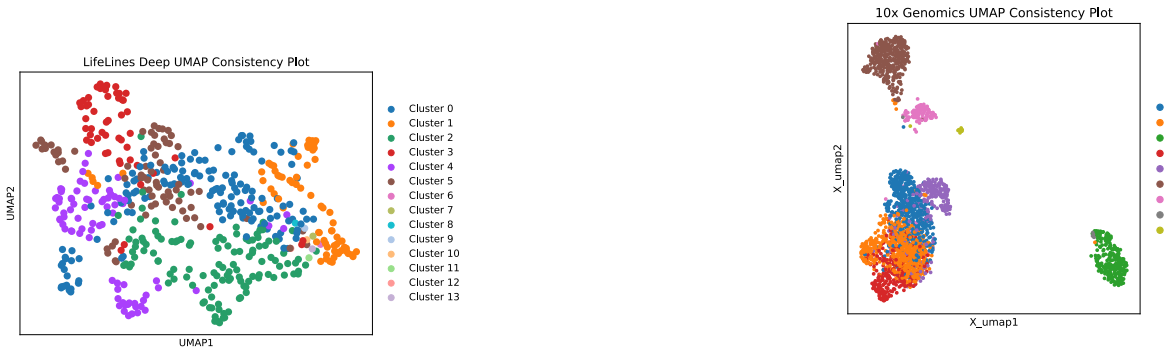
(a) (b) (c)

Figure 2.6: Assortativity in the LifeLines-DEEP dataset, comparing our method with the Zero-inflated Log-Normal (ZiLN) model. Solid lines represent our algorithm, dashed represent ZiLN. In one case we show the performance of our algorithm restricted to the metagenomics dataset (a) and when augmented with the metabolomics dataset (b). In both cases, ZiLN only has access to the metagenomics dataset, as it is a single-axis model. (c) The same test as (b), with the incorporation of phylum information as a prior (Theorem 2). Specifically, we used a Wishart prior whose parameter encoded a graph connecting two species if and only if they are in the same phylum.



(a) (b)

Figure 2.7: (a) The precision-recall curves comparing vanilla GmGM with its performance using Theorem 3's covariance thresholding trick. (b) Runtime comparison of vanilla GmGM and the covariance thresholding trick. Times reported were averaged over 50 runs. Data was drawn from two disjoint versions of the same distribution as from Figure 2.4a, to simulate there being multiple groups.



(a) (b)

Figure 2.8: Both (a) and (b) are UMAP plots, colored by Louvain clustering of GmGM's resultant graphs. (a) Species from the LifeLines-DEEP dataset. (b) Cells from the 10x Genomics Dataset. In both cases, we can see that clusters on our graph correspond to contiguous regions of UMAP-space.

of a shuffled video. We chose this test as it was the test used in the first Kronecker sum multi-axis model, BiGLasso. BiGLasso had to heavily downsample the image (to a 9x9 image with half the frames), and flatten the rows and frames into a single axis. Due to the speed improvements of our algorithm, and its ability to handle tensor-variate data, we were able to run our algorithm on the full-sized dataset and achieve a similar result in negligible time. Specifically, the reconstruction of the rows, columns, and frames all had an accuracy of 99%. In Figure 2.5b we show the shuffled data, with Figure 2.5c being the reconstruction using our algorithm.

Due to the simplicity of COIL-20, we chose to do a similar test on the more complicated EchoNet-Dynamic dataset. We expected there to be structure in our predicted graphs, which would allow us to detect the pattern of heartbeats. This pattern can be seen in Figure 2.5d. We were largely successful in this endeavor, with only one of the five videos we tested having poor results. For full details on the experiment, and how the heartbeat was extracted, see the supplementary material.

To test GmGM on our intended use case, 'omics data, we started with the E-MTAB-2805 transcriptomics dataset, as this dataset was used in the original scBiGLasso paper for exploratory analysis. In this dataset, each cell belongs to one of three classes (G, S, or G2M) depending on its stage in the cell cycle. To evaluate our algorithm's performance, we use assortativity. Assortativity measures the tendency of cells in the same class to cluster together in the estimated graph; an assortativity of 0 represents no tendency for connection, with 1 being the maximum. Negative assortativites are possible, if there is a tendency to connect to different classes. We report the assortativities in Figure 2.5a, comparing our results to EiGLasso and with the nonparanormal skeptic applied to the input.

Our fourth dataset, the LifeLines-DEEP dataset, was chosen because it was multi-modal, and because a single-axis graphical model, ZiLN (Prost et al., 2021), had already been tested on one of its modalities. We use assortativity as a metric, as that is what was used in prior work. Our results are comparable to their work (Figure 2.6). Our algorithm's runtime of 4.7 seconds was similar to ZiLN's 3.6 seconds, showing that our multi-axis method was capable of similar runtimes as single-axis methods for the first time. ZiLN was specifically built for metagenomics data - the compositionality assumption it makes would not be correct for most other datasets considered here, and hence we do not evaluate its performance on other datasets. Likewise, TeraLasso and EiGLasso were too slow to evaluate assortativity curves as in Figure 2.6.

Finally, we tested our approach on a 10x Genomics single-cell (RNA+ATAC) dataset taken from a B cell lymphoma tumour. We chose this dataset because it is multi-modal and quite large - 14,000 cells, 20,000 genes, and 100,000 peaks (although after preprocessing it became 2359 cells, 5350 genes, and 12485 peaks). This dataset is unlabelled, prohibiting more quantitative analyses; our main aim in this test was to prove that the algorithm could produce results in reasonable time. Our algorithm took 590 seconds to consider both modalities jointly and 52 seconds on just the scRNA-seq data. In contrast, EiGLasso took more than 60,000 seconds on the scRNA-seq data - due to its excessive runtime, we did not let EiGLasso finish.

To ensure our algorithm was producing sensible results on the 10x Genomics dataset, we performed

a 'UMAP consistency analysis'. Specifically, we ran Louvain clustering (Blondel et al., 2008) on our graph, and viewed the clusters in UMAP-space (McInnes et al., 2020). We believe this is a good test in this case, as nonlinear dimensionality reduction is used in almost every scRNA-seq study (Svensson et al., 2020). As we can see in Figure 2.8b, the clusters on our graph correspond to sensible regions in UMAP-space as well. In the supplementary material, we provide interpretations for some of these clusters. Figure 2.8a shows another UMAP consistency plot, for the LifeLines-DEEP dataset.

In the supplementary material, we give variants of the UMAP consistency plot for both the LifeLines-DEEP and 10x Genomics datasets. These variants use different clustering methods (Leiden (Traag et al., 2019) and Ensemble Clustering (Poulin & Théberge, 2019)), or tSNE instead of UMAP (Maaten & Hinton, 2008). In all cases the results are broadly similar, except that Leiden on the LifeLines-Deep dataset tends to put everything in one cluster.

All of the code to run the algorithm and recreate the experiments has been made publicly available on GitHub: https://github.com/AIStats-GmGM/GmGM, and we have a package "GmGM" on PyPI.

2.5 Limitations and future work

One limitation of our algorithm is its lack of compatibility with the standard L1 penalty. We do show that our restricted L1 penalty leads to better performance in synthetic data, even decisively outperforming L1-penalized prior work in some cases (Figure 2.4c). In the supplementary material, we also show that our restricted penalty leads to much better performance on the E-MTAB-2805 transcriptomics dataset. However, it would be nice to incorporate the standard L1 penalty, as it is a well-studied and well-accepted technique. This would be a rewarding avenue for future work.

Another limitation of our algorithm is that it does not handle the case in which two axes partially overlap. For example, in the LifeLines-DEEP dataset, not every person is in both the metabolomics and metagenomics datasets. For now, our solution to this is to restrict ourselves to the subset of indices that are available in all modalities sharing that axis (this was about 90% of people in the LifeLines-DEEP dataset). Prior multi-axis methods could only handle a single modality, whereas this partial overlap problem only arises when considering multiple modalities - a case which we are the first to consider. We intend to relax this restriction in future work.

One notable effect of our algorithm's speed improvements is that we are now limited by RAM rather than runtime; this affected our experiments, wherein we were more agressive with our quality control filtering of the 100,000 ATAC peaks in the 10x dataset. Before our work, algorithms were limited by how long it would take to run rather than how much space was needed - our algorithm is the first to be fast enough to hit memory limits. Our algorithm has a theoretically optimal space usage of $O(\sum_\ell d_\ell^2)$, since the output is a set of $d_\ell \times d_\ell$ precision matrices. However, given that we are often interested in *sparse* matrices, an avenue for future work would be to create variants of the algorithm that are capable of leveraging this sparsity. As our algorithm (along with the fastest prior work, EiGLasso and TeraLasso) depends on eigendecompositions, this would be a nontrivial task.

2.6 Conclusion

We have created a novel model, GmGM, which successfully generalizes Gaussian graphical models to the common scenario of multi-modal datasets. Furthermore, we demonstrated that our algorithm is significantly faster than prior work focusing on Gaussian multi-graphical models while still preserving state-of-the-art performance. These improvements allow multi-graphical models to be applied to datasets with axes of length in the tens of thousands. Additionally, we showed how to integrate a couple natural priors into our model, as well as provided a technique to partition datasets into smaller portions before analysis. The latter is particularly useful as runtime scales cubically after the computation of the empirical Gram matrices; partitioning a dataset in half provides an eightfold speedup. Finally, we demonstrated the application of our algorithm on five real-world datasets to prove its efficacy.

Acknowledgements

We thank the participants and the staff of LifeLines-DEEP for their collaboration. Funding for the project was provided by the Top Institute Food and Nutrition Wageningen grant GH001. The sequencing was carried out in collaboration with the Broad Institute.

We also thank the reviewers for their comments - this paper has been much improved thanks to their feedback.

Bibliography for Paper 1

10x Genomics. (2021, May). Flash-Frozen Lymph Node with B Cell Lymphoma (14k sorted nuclei). Retrieved May 10, 2023, from https://www.10xgenomics.com/resources/datasets/fresh-frozen-lymph-node-with-b-cell-lymphoma-14-k-sorted-nuclei-1-standard-2-0-0

Altenbuchinger, M., Weihs, A., Quackenbush, J., Grabe, H. J., & Zacharias, H. U. (2020). Gaussian and Mixed Graphical Models as (multi-)omics data analysis tools. *Biochimica et Biophysica Acta (BBA) - Gene Regulatory Mechanisms*, *1863*(6), 194418. https://doi.org/10.1016/j.bbagrm.2019.194418

Blondel, V. D., Guillaume, J.-L., Lambiotte, R., & Lefebvre, E. (2008). Fast unfolding of communities in large networks [arXiv:0803.0476 [cond-mat, physics:physics]]. *Journal of Statistical Mechanics: Theory and Experiment*, *2008*(10), P10008. https://doi.org/10.1088/1742-5468/2008/10/P10008

Buettner, F., Natarajan, K. N., Casale, F. P., Proserpio, V., Scialdone, A., Theis, F. J., Teichmann, S. A., Marioni, J. C., & Stegle, O. (2015). Computational analysis of cell-to-cell heterogeneity in single-cell RNA-sequencing data reveals hidden subpopulations of cells [Number: 2 Publisher: Nature Publishing Group]. *Nature Biotechnology*, *33*(2), 155–160. https://doi.org/10.1038/nbt.3102

Cai, T. T., Li, H., Liu, W., & Xie, J. (2016). Joint Estimation of Multiple High-dimensional Precision Matrices. *Statistica Sinica*, *26*(2), 445–464. https://doi.org/10.5705/ss.2014.256

Dahl, A., Hore, V., Iotchkova, V., & Marchini, J. (2013, December). Network inference in matrix-variate Gaussian models with non-independent noise [arXiv:1312.1622 [stat]]. https://doi.org/10.48550/arXiv.1312.1622

Danaher, P., Wang, P., & Witten, D. M. (2014). The joint graphical lasso for inverse covariance estimation across multiple classes. *Journal of the Royal Statistical Society. Series B, Statistical methodology*, *76*(2), 373–397. https://doi.org/10.1111/rssb.12033

Friedman, J., Hastie, T., & Tibshirani, R. (2008). Sparse inverse covariance estimation with the graphical lasso. *Biostatistics*, *9*(3), 432–441. https://doi.org/10.1093/biostatistics/kxm045

Greenewald, K., Zhou, S., & Hero III, A. (2019, September). Tensor Graphical Lasso (TeraLasso) [arXiv:1705.03983 [stat]]. Retrieved February 24, 2023, from http://arxiv.org/abs/1705.03983

Kalaitzis, A., Lafferty, J., Lawrence, N. D., & Zhou, S. (2013). The Bigraphical Lasso [ISSN: 1938-7228]. *Proceedings of the 30th International Conference on Machine Learning*, 1229–1237. Retrieved February 24, 2023, from https://proceedings.mlr.press/v28/kalaitzis13.html

Kolda, T. G., & Bader, B. W. (2009). Tensor Decompositions and Applications. *SIAM Review*, *51*(3), 455–500. https://doi.org/10.1137/07070111X

Lewis, A. S. (1996). Derivatives of Spectral Functions [Publisher: INFORMS]. *Mathematics of Operations Research*, *21*(3), 576–588. https://doi.org/10.1287/moor.21.3.576

Li, S., López-García, M., Lawrence, N. D., & Cutillo, L. (2022, March). Scalable Bigraphical Lasso: Two-way Sparse Network Inference for Count Data [arXiv:2203.07912 [cs, stat]]. Retrieved February 24, 2023, from http://arxiv.org/abs/2203.07912

Liu, H., Han, F., Yuan, M., Lafferty, J., & Wasserman, L. (2012). The nonparanormal SKEPTIC. *Proceedings of the 29th International Conference on Machine Learning, ICML 2012*, 1415–1422. Retrieved October 9, 2023, from https://pure.johnshopkins.edu/en/publications/the-nonparanormal-skeptic-4

Maaten, L. v. d., & Hinton, G. (2008). Visualizing Data using t-SNE. *Journal of Machine Learning Research*, *9*(86), 2579–2605. Retrieved February 22, 2024, from http://jmlr.org/papers/v9/vandermaaten08a.html

Mazumder, R., & Hastie, T. (2012). Exact Covariance Thresholding into Connected Components for Large-Scale Graphical Lasso. *Journal of machine learning research : JMLR*, *13*, 781–794. Retrieved October 9, 2023, from https://www.ncbi.nlm.nih.gov/pmc/articles/PMC4225650/

McCarter, C., & Kim, S. (2014). On Sparse Gaussian Chain Graph Models. *Advances in Neural Information Processing Systems*, *27*. Retrieved February 22, 2024, from https://papers.nips.cc/paper_files/paper/2014/hash/81c650caac28cdefce4de5ddc18befa0-Abstract.html

McInnes, L., Healy, J., & Melville, J. (2020, September). UMAP: Uniform Manifold Approximation and Projection for Dimension Reduction [arXiv:1802.03426 [cs, stat]]. https://doi.org/10.48550/arXiv.1802.03426

Nene, S. A., Nayar, S. K., & Murase, H. (n.d.). Columbia Object Image Library (COIL-20).

Ouyang, D., He, B., Ghorbani, A., Yuan, N., Ebinger, J., Langlotz, C. P., Heidenreich, P. A., Harrington, R. A., Liang, D. H., Ashley, E. A., & Zou, J. Y. (2020). Video-based AI for beat-to-beat assessment of cardiac function [Number: 7802 Publisher: Nature Publishing Group]. *Nature*, *580*(7802), 252–256. https://doi.org/10.1038/s41586-020-2145-8

Poulin, V., & Théberge, F. (2019). Ensemble clustering for graphs: Comparisons and applications [Number: 1 Publisher: SpringerOpen]. *Applied Network Science*, *4*(1), 1–13. https://doi.org/10.1007/s41109-019-0162-z

Prost, V., Gazut, S., & Brüls, T. (2021). A zero inflated log-normal model for inference of sparse microbial association networks [Publisher: Public Library of Science]. *PLOS Computational Biology*, *17*(6), e1009089. https://doi.org/10.1371/journal.pcbi.1009089

Svensson, V., da Veiga Beltrame, E., & Pachter, L. (2020). A curated database reveals trends in single-cell transcriptomics. *Database*, *2020*, baaa073. https://doi.org/10.1093/database/baaa073

Tigchelaar, E. F., Zhernakova, A., Dekens, J. A. M., Hermes, G., Baranska, A., Mujagic, Z., Swertz, M. A., Muñoz, A. M., Deelen, P., Cénit, M. C., Franke, L., Scholtens, S., Stolk, R. P., Wijmenga, C., & Feskens, E. J. M. (2015). Cohort profile: LifeLines DEEP, a prospective, general population cohort study in the northern Netherlands: Study design and baseline characteristics [Publisher: British Medical Journal Publishing Group Section: Epidemiology]. *BMJ Open*, *5*(8), e006772. https://doi.org/10.1136/bmjopen-2014-006772

Traag, V. A., Waltman, L., & van Eck, N. J. (2019). From Louvain to Leiden: Guaranteeing well-connected communities [Number: 1 Publisher: Nature Publishing Group]. *Scientific Reports*, *9*(1), 5233. https://doi.org/10.1038/s41598-019-41695-z

Tsiligkaridis, T., Hero III, A. O., & Zhou, S. (2013). Convergence Properties of Kronecker Graphical Lasso Algorithms [arXiv:1204.0585 [stat]]. *IEEE Transactions on Signal Processing*, *61*(7), 1743–1755. https://doi.org/10.1109/TSP.2013.2240157

Wang, Y., & Hero, A. (2021). SG-PALM: A Fast Physically Interpretable Tensor Graphical Model [ISSN: 2640-3498]. *Proceedings of the 38th International Conference on Machine Learning*, 10783–10793. Retrieved February 27, 2023, from https://proceedings.mlr.press/v139/wang21k.html

Wang, Y., Jang, B., & Hero, A. (2020, February). The Sylvester Graphical Lasso (SyGlasso) [arXiv:2002.00288 [cs, stat]]. https://doi.org/10.48550/arXiv.2002.00288

Yoon, J. H., & Kim, S. (2020). EiGLasso: Scalable Estimation of Cartesian Product of Sparse Inverse Covariance Matrices [ISSN: 2640-3498]. *Proceedings of the 36th Conference on Uncertainty in Artificial Intelligence (UAI)*, 1248–1257. Retrieved February 24, 2023, from https://proceedings.mlr.press/v124/ho-yoon20a.html

Chapter 3

Gaussian graphical modelling without independence assumptions for uncentered data

The following paper arose out of a problem with a consistent assumption made about datasets in a multi-axis context: that they are centered (i.e. assuming that the mean is zero). This assumption was applied in all Kronecker-sum-structured prior work, including our work in Chapter 2. Not only is this a clearly false assumption, but it has drastic effects on the accuracy of the results (as we will show in the this paper).

For single-axis work, assuming a mean of zero is easily justifiable - and this is likely why it was overlooked in multi-axis work. After all, if one has many samples, it is easy to find and remove the mean. The crucial difference in the multi-axis case is that one does not have many samples - there is only one sample (the matrix itself). There are a few ad-hoc methods to address this; in this paper, we introduce a statistically rigorous method, and show how it outperforms those ad-hoc methods. We developed a solution as a wrapper around existing algorithms - thus, one could use it with GmGM, TeraLasso, or any other existing Kronecker-sum-structured method.

While the paper solves an important problem, it is not efficiently scalable to large datasets - the algorithm flip-flops between mean estimation and conditional dependency estimation, and thus requires running GmGM (or TeraLasso) multiple times to arrive at a solution. However, we did future-proof the method by ensuring its compatibility with our improved version of GmGM (Chapter 6) which was in concurrent development with this method, and we showed in this paper's appendix (Appendix B) that one could still get the majority of the benefit by only flip-flopping a handful of times. It would not scale to millions of cells, but in concert with the developments of Chapter 6 it should still be scalable far beyond what state-of-the-art was before this thesis.

The paper was published in the Proceedings of the AAAI Conference on Artificial Intelligence in 2025. It is presented unchanged from the published version, except for formatting changes to fit the thesis style. The supplementary material for the paper is available in Appendix B, which contains further details and experiments. For further details and experiments that came up during the viva, see Appendix H.

Gaussian Graphical Modelling Without Independence Assumptions for Uncentered Data

Bailey Andrew, David R Westhead, Luisa Cutillo

Published in the Proceedings of the 39th Annual AAAI Conference on Artificial Intelligence

Abstract. The independence assumption between random variables is a useful tool to increase the tractability of a modelling framework. However, this assumption can be too simplistic; failing to take dependencies into account can cause models to fail dramatically. The field of multi-axis graphical modelling (also called multi-way modelling, Kronecker-separable modelling) has seen growth over the past decade, but these models require that the data have zero mean. In the multi-axis case, inference is typically done in the single sample scenario, making mean inference impossible.

In this paper, we demonstrate how the zero-mean assumption can cause egregious modelling errors for Kronecker-sum-decomposable Gaussian graphical models, as well as propose a relaxation to the zero-mean assumption that allows the avoidance of such errors. Specifically, we propose the "Kronecker-sum-structured mean" assumption, which leads to models with nonconvex-but-unimodal log-likelihoods that can be solved efficiently with coordinate descent.

3.1 Introduction

We often wish to find networks ('graphs') that describe our data. For example, we may be interested in gene regulatory networks in biology, or social interaction networks in epidemiology. In these cases, the graph itself is an object of interest. When the goal of an analysis does not involve a graph, the creation of one can still be useful as a preprocessing step for further insights, such as for clustering. Their use is not limited to clustering; the first step of the popular dimensionality reduction method UMAP, for example, is to create a (typically nearest neighbors) graph McInnes et al., 2020, but one could also experiment with other graphs.

While there are many types of graphs, this paper will focus on *conditional dependency graphs*. Intuitively, two vertices are connected in such a graph if they are still statistically dependent after conditioning out the other variables. We denote the property that x and y are conditionally independent, given z, as $x \perp\!\!\!\perp y \mid z$. We will focus on the case where we condition over all other variables in the dataset $\mathcal{D}$, i.e. where $z = \mathcal{D}_{\setminus x \setminus y} \stackrel{def}{=} \mathcal{D} - \{x, y\}$.

$$x \perp\!\!\!\perp y \mid z \stackrel{def}{\iff} \mathbb{P}\left[x, y \mid z\right] = \mathbb{P}\left[x \mid z\right] \mathbb{P}\left[y \mid z\right]$$

Conditional dependency graphs have several convenient properties. They are interpretable, and intuitively capture the 'direct effect' of two variables on each other (rather than 'indirect effects' passing through confounders and mediators). They are typically sparse, and are related to causality[1]. They also have a very useful property in multivariate Gaussian datasets, namely that two Gaussian variables are conditionally independent if and only if the corresponding term in the inverse covariance matrix is zero.

$$x \perp\!\!\!\perp y \mid \mathcal{D}_{\setminus x \setminus y} \stackrel{Gaussian}{\iff} \mathbf{\Sigma}^{-1}_{xy} = 0$$

The inverse covariance matrix is also called the precision matrix, and we will denote it $\mathbf{\Psi}$. It can be interpreted as an adjacency matrix for the graph of conditional dependencies. Due to this convenient correspondence, and due to the fact that practically all multi-axis work has so far been limited to the Gaussian case, we will assume in this paper that our data is Gaussian. This type of model, without the Gaussian assumption, is more generally known as a Markov random field.

Through tools such as the Nonparanormal Skeptic Liu et al., 2012, this assumption can be weakened to the Gaussian copula assumption (intuitively, the variables have arbitrary continuous marginals but still interact 'Gaussian-ly').

When making independence assumptions, this problem is solved by well-known methods such as the Graphical Lasso Friedman et al., 2008. However, independence assumptions are often false in practice. They become particularly egregious in the case of omics data, such as single cell RNA-sequencing (scRNA-seq). These datasets take the form of a cells-by-genes matrix. We might be interested in a gene regulatory network, in which case we assume the cells are independent. Alternatively, we might be interested in the cellular microenvironment, in which case we assume the genes are independent. Whichever one we want to learn, we are put in the awkward position of assuming the other does not exist.

To avoid independence assumptions altogether, we can vectorize our dataset $\mathbf{X}$: $\text{vec}\left[\mathbf{X}\right] \sim \mathcal{N}(\boldsymbol{\mu}, \mathbf{\Psi}^{-1})$. Instead of n samples of m features, we have 1 sample of nm features (each element of the data matrix is a feature). For scRNA-seq, the precision matrix is of size $O(n^2m^2)$ and represents the dependencies between different (cell, gene) pairs. The problem becomes both computationally and statistically intractable; we quickly run out of space to store such matrices, and have no way of being confident in our results to any degree of statistical certainty. We need to find a middle ground between independence and full dependence.

To do this, we can take advantage of matrix structure. We are not truly interested in a graph of (cell, gene) pairs, but rather a graph of cells and a graph of genes. Thus, we can make a 'graph decomposition assumption'; our (cell, gene) pair graph should be able to be factored (for some definition of factoring ζ) into the cell graph and the gene graph: $\text{vec}\left[\mathbf{X}\right] \sim \mathcal{N}(\boldsymbol{\mu}, \zeta\left(\mathbf{\Psi}_{cells}, \mathbf{\Psi}_{genes}\right)^{-1})$.

[1] Under modest assumptions, that can be violated in some real-world scenarios, conditional dependency graphs form the 'skeleton' of a causal directed acyclic graph, i.e. the causal graph with directionality removed.

Several choices of decomposition have appeared in the literature, such as the Kronecker product Dutilleul, 1999 and squared Kronecker sum decompositions Wang et al., 2020; in this paper, we focus on the Kronecker sum decomposition Kalaitzis et al., 2013 due to its popularity, convexity, relationship to conditional dependence, interpretability as a Cartesian product, and correspondence to a maximum entropy distribution. The Kronecker sum is denoted $\oplus$ and is defined as $\mathbf{A}_{a\times a} \oplus \mathbf{B}_{b\times b} = \mathbf{A}_{a\times a} \otimes \mathbf{I}_{b\times b} + \mathbf{I}_{a\times a} \otimes \mathbf{B}_{b\times b}$, where $\otimes$ is the Kronecker product. The operation is associative, allowing a straightforward generalization to more than two axes (i.e. tensor-variate datasets); this was first explored by Greenewald et al. (2019). Under the zero-mean and Kronecker sum decomposability assumptions, the model for a dataset $\mathcal{D}$ is as follows:

$$\begin{aligned} \mathcal{D} &\sim \mathcal{N}_{KS}\left(\mathbf{0}, \{\mathbf{\Psi}_\ell\}_\ell\right) \\ \iff \operatorname{vec}\left[\mathcal{D}\right] &\sim \mathcal{N}\left(\mathbf{0}, \left(\bigoplus_\ell \mathbf{\Psi}_\ell\right)^{-1}\right) \end{aligned}$$

The zero-mean assumption is conspicuous, but necessary; recall that we transformed our dataset from n samples and m features to a single sample of nm features. Estimating the mean of a single sample is a recipe for disaster; when one centers the dataset, they will be left with a constant zero vector.

We could consider decomposing the mean of our model, just like for our precision matrices. A natural and analogous decomposition would be that our mean takes the form $\boldsymbol{\mu}_{\text{cells}} \widetilde{\oplus} \boldsymbol{\mu}_{\text{genes}} = \boldsymbol{\mu}_{\text{cells}} \otimes \mathbf{1} + \mathbf{1} \otimes \boldsymbol{\mu}_{\text{genes}}$; this operation intermixes the means in the same way the Kronecker sum intermixes the variances. For Kronecker *product* models (not Kronecker sum models), this has been considered before Allen and Tibshirani, 2010. It was shown that the maximum likelihood for μ_ℓ has the following form (in the two-axis case, where d_ℓ represents the number of elements in axis ℓ):

$$\boldsymbol{\mu}_1 = \frac{\mathbf{1}^T\left(\mathbf{X} - \mathbf{1}\boldsymbol{\mu}_2^T\right)}{d_2} \qquad\qquad \boldsymbol{\mu}_2 = \frac{\mathbf{1}^T\left(\mathbf{X} - \mathbf{1}\boldsymbol{\mu}_1^T\right)}{d_1}$$

We will see later that such formulas are not as simple in the Kronecker sum case. First, though, let us note that the estimate of $\boldsymbol{\mu}_\ell$ does not depend on the dependencies in the data ($\mathbf{\Psi}_\ell$). This is a mathematically convenient property to have, but is philosophically troubling. Suppose we have a dataset of only two points, both distinct and independent. The true mean value is just their average. Now, suppose we duplicate the first point several times, perhaps with some modest noise added - the mean estimate will lie much closer to the first point.

In fact, we can choose how many times to duplicate the first or second points in order to place the estimated mean anywhere between the original points. However, this estimated mean is an artifact of the dependencies in our data! The true mean, ultimately, still lies at its original location.

3.1.1 Our contributions

We are naturally led to the following question:

Problem 1. *For the noncentral Kronecker sum model, does using the mean decomposition $\widetilde{\bigoplus}_\ell \boldsymbol{\mu}_\ell$ allow its parameters to be effectively estimated?*

In this paper, we show that the answer to this question is yes. For the Kronecker-sum-structured model, the estimators for $\boldsymbol{\mu}_\ell$ depend on $\mathbf{\Psi}_\ell$. However, due to this dependency, our problem is no longer convex; there could be some non-global minima!

Problem 2. *Is our estimator for the parameters of the noncentral Kronecker-sum-structured normal distribution guaranteed to be the global maximum likelihood estimator?*

We also show that the answer to this question is affirmative; in fact, there are no non-global minima, despite the nonconvexity.

One may also wonder if the zero-mean assumption is truly problematic. In fact, the zero-mean assumption can have an extremely dramatic negative effect on performance. In the simple scenario in

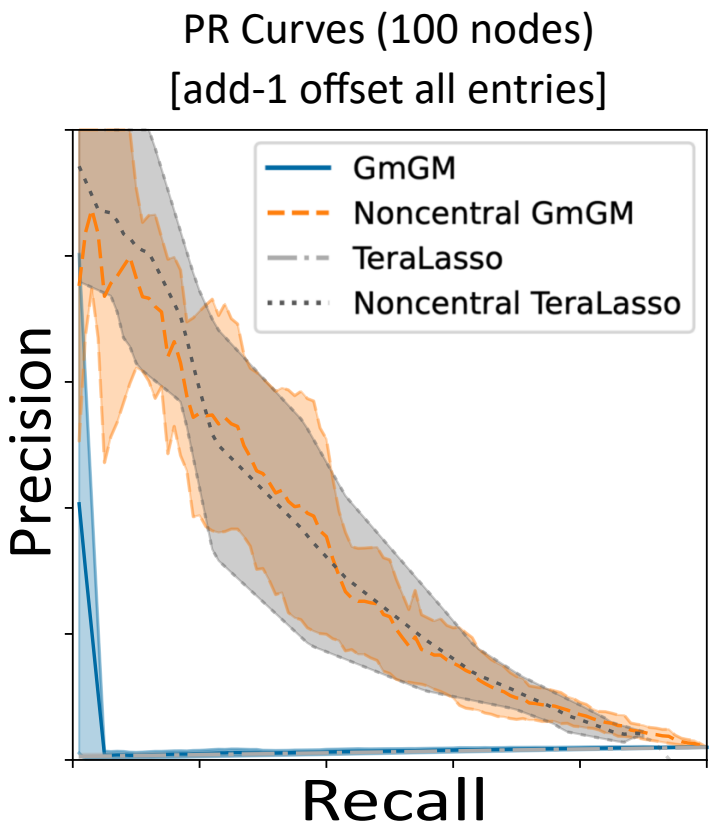


Figure 3.1: Precision and recall for a synthetic dataset in which data was generated from a Kronecker-sum normal distribution with a constant value of 1 added to every entry of the matrix. Error bars are the best/worst performance over 10 trials; the center line is average performance.

which we add a constant offset to every element, we can see that methods that do not try to estimate the mean are practically worthless, whereas those that do (which we shall call 'noncentral') maintain reasonable performance (Figure 3.1).

3.1.2 Notation

In this paper, lowercase letters a will refer to scalars, lowercase bold $\mathbf{v}$ will refer to vectors, uppercase bold $\mathbf{M}$ will refer to matrices, and uppercase calligraphic $\mathcal{T}$ will refer to tensors. We will conceptualize our dataset to be a tensor $\mathcal{D}$ with axes lengths d_ℓ for each axis ℓ. $d_{<\ell}$ and $d_{>\ell}$ are shorthands for the product of lengths of axes before and after ℓ, respectively. $d_{\setminus\ell}$ is the product of all axes other than ℓ; from the perspective of ℓ, this is the number of samples for that axis. $d_\forall$ is the product of all d_ℓ. We typically follow the notation of Kolda and Bader (2009) for tensor operations.

Let $\mathbf{I}_a$ and $\mathbf{1}_a$ be the $a \times a$ identity matrix and length a vector of ones, respectively. We formally define the Kronecker sum of matrices and analogous operation on vectors as follows:

$$\bigoplus_\ell \mathbf{\Psi}_\ell = \sum_\ell \mathbf{I}_{d_{<\ell}} \otimes \mathbf{\Psi}_\ell \otimes \mathbf{I}_{d_{>\ell}}$$

$$\widetilde{\bigoplus_\ell} \boldsymbol{\mu}_\ell = \sum_\ell \mathbf{1}_{d_{<\ell}} \otimes \boldsymbol{\mu}_\ell \otimes \mathbf{1}_{d_{>\ell}}$$

We will let $\mathbf{\Psi}_{\setminus\ell}$ be a shorthand for $\bigoplus_{\ell' \neq \ell} \mathbf{\Psi}_{\ell'}$, with an analogous definition for $\boldsymbol{\mu}_{\setminus\ell}$. Kronecker products, and by extension Kronecker sums, have a convenient permutation-invariance property; we can swap the order of summands without affecting the results, as long as we perform this swap consistently across an equation and permute matrices accordingly; we can typically rewrite equations involving

$\bigoplus_\ell \mathbf{\Psi}_\ell$ into those involving $\mathbf{\Psi}_\ell \oplus \mathbf{\Psi}_{\setminus\ell}$. We will define $\mathbf{\Omega} = \bigoplus_\ell \mathbf{\Psi}_\ell$ and $\boldsymbol{\omega} = \widetilde{\bigoplus}_\ell \boldsymbol{\mu}_\ell$. We define noncentral Kronecker sum normal distribution as:

$$\operatorname{vec}[\mathcal{D}] \sim \mathcal{N}\left(\boldsymbol{\omega}, \mathbf{\Omega}^{-1}\right)$$

We let $\mathbb{K}_{\mathbf{M}}$ be the space of Kronecker-sum-decomposable matrices[2], i.e. $\mathbf{\Omega} \in \mathbb{K}_{\mathbf{M}}$. Likewise, $\boldsymbol{\omega} \in \mathbb{K}_{\mathbf{v}}$ is the space of Kronecker-sum-decomposable vectors. Finally, let $\mathbb{K}_{\mathbf{Mv}}$ be the space of vectors of the form $\mathbf{Xy}$, where $\mathbf{X} \in \mathbb{K}_{\mathbf{M}}, \mathbf{y} \in \mathbb{K}_{\mathbf{v}}$. All of these spaces are linear subspaces of $\mathbb{R}^{d_\forall}$ or $\mathbb{R}^{d_\forall \times d_\forall}$. We may use $\mathbb{K}$ as a shorthand when discussing properties that apply to both $\mathbb{K}_{\mathbf{M}}$ and $\mathbb{K}_{\mathbf{v}}$.

It is important to note that the parameterization of $\mathbb{K}$ we have used so far is not identifiable. If $\sum_\ell c_\ell = 0$, then $\bigoplus_\ell (\mathbf{\Psi}_\ell + c_\ell \mathbf{I}) = \bigoplus_\ell \mathbf{\Psi}_\ell$. Greenewald et al. (2019) use the identifiable representation $\tau \mathbf{I}_\forall + \bigoplus_\ell \widetilde{\mathbf{\Psi}}_\ell$, where $\operatorname{tr}\left[\widetilde{\mathbf{\Psi}}_\ell\right] = 0$. Likewise, we can identifiably represent our mean decomposition as $m\mathbf{1} + \widetilde{\bigoplus}_\ell \widetilde{\boldsymbol{\mu}}_\ell$, where $\mathbf{1}^T \widetilde{\boldsymbol{\mu}}_\ell = 0$.

The last important property we will introduce is the matrix representation of Kronecker products of vectors: note that $\boldsymbol{\mu}_\ell \otimes \mathbf{1}_{d_{\setminus\ell}} = \left(\mathbf{I}_{d_\ell} \otimes \mathbf{1}_{d_{\setminus\ell}}\right) \boldsymbol{\mu}_{\boldsymbol{\ell}}$.

3.2 Estimation method

As a shorthand, let $\mathbf{x} = \operatorname{vec}[\mathcal{D}]$. Our negative log likelihood looks just like that of our normal distribution, except that the parameters are restricted to lie in $\mathbb{K}$. To ensure existence of $\mathbf{\Omega}$, it is necessary to either add a regularization penalty, as in TeraLasso, or restrict $\mathbf{\Omega}$ to a low-rank subspace, as proposed by Andrew et al. (2024b). These are details of the estimator for $\mathbf{\Omega}$, which do not affect the estimator of $\boldsymbol{\omega}$; we do not focus on these aspects of the problem.

$$\mathrm{p}(\mathcal{D}) = \frac{\sqrt{|\mathbf{\Omega}|}}{(2\pi)^{\frac{d_\forall}{2}}} e^{\frac{-1}{2}(\boldsymbol{x}-\boldsymbol{\omega})^T \mathbf{\Omega}(\boldsymbol{x}-\boldsymbol{\omega})}$$

$$\mathrm{NLL}(\mathcal{D}) \propto \frac{-1}{2}\log|\mathbf{\Omega}| + \frac{1}{2}(\boldsymbol{x}-\boldsymbol{\omega})^T \mathbf{\Omega}(\boldsymbol{x}-\boldsymbol{\omega})$$

Despite its ubiquity, the normal distribution is actually fairly poorly behaved from an optimization perspective; its NLL is not convex, nor is it even geodesically convex Hosseini and Sra, 2015. Thankfully, the MLE of the mean (for the unrestricted normal) has a closed form solution, $\omega = \frac{1}{n}\sum_i^n \mathbf{x}_i$, so this nonconvexity does not affect results in practice. Unfortunately, due to our $\mathbb{K}_{\mathbf{v}}$ restriction, this solution no longer holds.

While it is not convex in $(\mathbf{\Omega}, \boldsymbol{\omega})$ jointly, it is convex in each argument individually - i.e, it is biconvex. This suggests a flip-flop optimization scheme. For fixed $\boldsymbol{\omega}$, the estimation of $\mathbf{\Omega}$ reduces to the noncentral case (by subtracting out $\boldsymbol{\omega}$); several algorithms already exist to solve this. Thus, we focus on optimization when $\mathbf{\Omega}$ is fixed.

Optimization w.r.t. $\boldsymbol{\omega}$ can be framed as a convex constrained quadratic programming (QP) problem. However, it will turn out to be convenient to further break the problem up into its constituent parts $\left(m, \{\widetilde{\boldsymbol{\mu}}_\ell\}\right)$, and minimize them each individually. Let $\theta_\ell = \mathbf{1}_{d_{\ell'}}^T \mathbf{\Psi}_{\ell'} \mathbf{1}_{d_{\ell'}}$, $\theta_{\setminus\ell} = \sum_{\ell' \neq \ell} \frac{d_\forall}{d_\ell d_{\ell'}} \theta_\ell$, and $\mathbf{x}_\ell$ be the vectorization of $\mathcal{D}$ done in the order as if axis ℓ were the first axis. To preserve space, we will defer the algebraic manipulations to the appendix, and simply note here that the optimal value of m, with all other values fixed, is:

$$m = \frac{\left(\mathbf{x} - \widetilde{\bigoplus}_\ell \widetilde{\boldsymbol{\mu}}_\ell\right)\left(\widetilde{\bigoplus} \mathbf{\Psi}_\ell \mathbf{1}_{d_\ell}\right)}{\sum_\ell d_{\setminus\ell}\theta_\ell} \tag{3.1}$$

Observe that, when the elements of $\mathbf{x}$ are independent, this formula expresses the average distance of the dataset from 0 (after accounting for the axis-wise means $\widetilde{\boldsymbol{\mu}}_\ell$).

[2]Technically, we need to specify the dimensionality of each term in the decomposition to define this space, but this would be notationally cumbersome and is always clear from context.

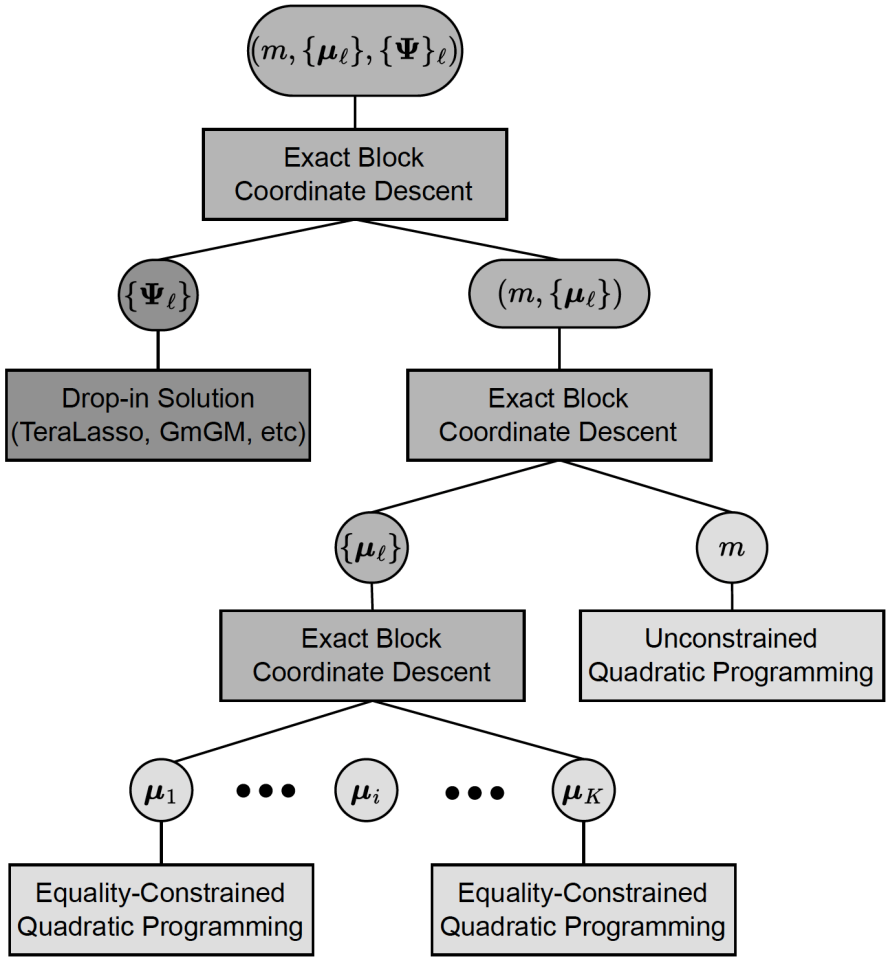


Figure 3.2: A graphical summary of the manner in which we divide our problem into sub-problems. While the main problem is nonconvex, all sub-problems are convex, and by Theorem 4 the main problem has a unique solution.

For $\widetilde{\boldsymbol{\mu}}_\ell$ (with $\widetilde{\boldsymbol{\mu}}_{\setminus\ell}$ fixed, and again deferring the derivation to the appendix) we have the following QP problem:

$$\begin{aligned}
&\underset{\widetilde{\boldsymbol{\mu}}_\ell}{\operatorname{argmin}}\ \widetilde{\boldsymbol{\mu}}_\ell^T \mathbf{A}_\ell \widetilde{\boldsymbol{\mu}}_\ell + \mathbf{b}_\ell^T \widetilde{\boldsymbol{\mu}}_\ell \\
&\text{where } \mathbf{A}_\ell = d_{\setminus\ell}\boldsymbol{\Psi}_\ell + \theta_{\setminus\ell}\mathbf{I}_{d_\ell} \\
&\qquad \boldsymbol{b}_\ell = m d_{\setminus\ell}\mathbf{1}_{d_\ell}^T\boldsymbol{\Psi}_\ell + m\theta_{\setminus\ell}\mathbf{1}_{d_\ell} + \left[\widetilde{\boldsymbol{\mu}}_{\setminus\ell}^T\boldsymbol{\Psi}_{\setminus\ell}\mathbf{1}_{d_{\setminus\ell}}\right]\mathbf{1}_{d_\ell} \\
&\qquad\quad - \mathbf{x}_\ell^T\left(\boldsymbol{\Psi}_\ell \otimes \mathbf{1}_{d_{\setminus\ell}}\right) - \mathbf{x}_\ell^T\left(\mathbf{1}_\ell \otimes \boldsymbol{\Psi}_{\setminus\ell}\mathbf{1}_{d_{\setminus\ell}}\right) \\
&\qquad \widetilde{\boldsymbol{\mu}}_\ell^T\mathbf{1}_{d_\ell} = \mathbf{0}
\end{aligned}$$

Parts of the definition of $\mathbf{b}_\ell$ can be simplified further, leading to more efficient computations, but such manipulations are notationally complicated to express; we defer them to the appendix, where we also show that $\mathbf{A}_\ell$ is guaranteed to be invertible. QP problems with linear equality constraints have closed-form solutions. In particular, we have that:

$$\widetilde{\boldsymbol{\mu}}_\ell = \frac{\mathbf{1}_{d_\ell}^T\boldsymbol{A}_\ell^{-1}\boldsymbol{b}_\ell}{\mathbf{1}_{d_\ell}^T\mathbf{A}_\ell^{-1}\mathbf{1}_{d_\ell}}\boldsymbol{A}_\ell^{-1}\mathbf{1}_{d_\ell} - \mathbf{A}_\ell^{-1}\boldsymbol{b}_\ell \tag{3.2}$$

As we have closed-form solutions to all our coordinatewise minima, block coordinate descent is a natural solution. While typically it is advisable to avoid directly calculating inverses $\mathbf{A}_\ell^{-1}$ in the solution to QP problems, inversion here is likely to be cheap; most solvers for $\boldsymbol{\Psi}_\ell$ require calculating eigendecompositons of $\boldsymbol{\Psi}_\ell$. Thus, the eigendecomposition, and hence inverse, of $\mathbf{A}_\ell$ is readily available. This answers Problem 1.

Some methods, such as GmGM Andrew et al., 2024a, require only a single eigendecomposition, and then find the optimum while staying in 'eigen-space'. They use the eigenvectors of covariance matrices; when the mean is updated, the new covariance matrices can be expressed in terms of rank-one-updates of the original covariance matrix.

Rank-one updates of eigendecompositions are cheap (comparatively); thus, for each flip-flop of our algorithm, if GmGM is used as the solver for $\boldsymbol{\Psi}_\ell$, then it can stay entirely in eigenspace - we preserve the 'only-one-eigendecomposition' property of GmGM. Other methods, such as TeraLasso, will still require multiple eigendecompositions, but these will remain useful for our calculation of $\mathbf{A}_\ell^{-1}$. It is because of these convenient computational properties that we have chosen a flip-flop route for estimation of $\widetilde{\boldsymbol{\mu}}_\ell$, despite the fact that the optimization problem is jointly convex in $\{\widetilde{\boldsymbol{\mu}}_\ell\}_\ell$.

See Algorithm 1 for a pseudocode presentation of our algorithm. Note that the pseudocode does not take into account potential speedups that come from sharing information between the precision matrix and mean estimation tasks, such as sharing eigenvectors. We wanted our method to be able to be used as a 'drop-in wrapper' for pre-existing methods. Accordingly, we have presented and implemented our methodology in its most general form. For a graphical overview of how we optimize, see Figure 3.2.

3.3 Global optimality and complexity

The optimization procedure we consider is not convex, merely biconvex. How do we know if the solution we find is globally optimal? In this section, we will show that, despite the non-convexity, the problem still has the property that all local minima are global minima. In fact, when identifiable representations are used, we will see that there is exactly one minimum. As in the last section, we will defer the more mechanical details of the proof to the appendix, and rather give a sketch of the main ideas here.

Theorem 4. *The maximum likelihood of the noncentral Kronecker-sum-structured normal distribution has a unique maximum, which is global. The estimator defined by Algorithm 1 converges to this.*

Proof. See appendix. Sketch given below. □

The general idea is to perform a one-to-one transformation on the problem to turn it into a strictly convex problem in a different set of parameters - specifically, an instance of conic programming with

Algorithm 1 Noncentral KS-structured Gaussian MLE

Input: Dataset $\mathcal{D}$ of size $d_1, ..., d_K$
Input: Sub-procedure $\bigoplus$GRAPH that estimates $\boldsymbol{\Psi}_\ell$
such as TeraLasso, GmGM
Output: Identifiable MLE $\left(m, \widetilde{\boldsymbol{\mu}}_\ell, \tau, \widetilde{\boldsymbol{\Psi}}_\ell\right)$

#**Initialize Parameters with Reasonable Guesses**
$m^{(0)} \leftarrow \frac{1}{d_\forall} \sum_{i=1}^{d_\forall} \text{vec}\,[\mathcal{D}]_i$
for $\ell \in \{1, ..., K\}$
Let $\mathbf{x}_\ell$ be vectorized $\mathcal{D}$ with ℓ as first axis.
Initial guess is the mean along axis ℓ:
$\widetilde{\boldsymbol{\mu}}_\ell^{(0)} \leftarrow \frac{1}{d_{\setminus\ell}} \sum_{i=1}^{d_{\setminus\ell}} \mathbf{x}_{\ell,i} - m^{(0)}$
end for
#**Iterate Until Convergence**
while not converged
$\left\{\boldsymbol{\Psi}_\ell^{(i+1)}\right\}_\ell \leftarrow \bigoplus \text{GRAPH}\left(\mathcal{D} - m^{(i)}\mathbf{1} - \widetilde{\bigoplus}_{\ell'} \boldsymbol{\mu}_\ell^{(i)}\right)$
while not converged
#**Find Mean Parameters**
$m^{(i+1)} \leftarrow$ (Equation 3.1)
for $\ell \in \{1, ..., K\}$
$\widetilde{\boldsymbol{\mu}}_\ell^{(i+1)} \leftarrow$ (Equation 3.2)
end for
end while
end while
Map $\{\boldsymbol{\Psi}_\ell\}$ to identifiable parameterization $\left(\left\{\widetilde{\boldsymbol{\Psi}}_\ell\right\}, \tau\right)$

linear constraints. Letting $\mathfrak{A}$ be our original problem, $\mathfrak{A}'$ be a slight modification of it, and $\mathfrak{B}$ be the conic problem, we show the following chain of implications:

$$\begin{aligned}
& (\boldsymbol{\omega}, \boldsymbol{\Omega}) \in \text{local minima}\,(\mathfrak{A}) \\
\iff & (\boldsymbol{\omega}, \boldsymbol{\Omega}, s = 1) \in \text{local minima}\left(\mathfrak{A}'\right) \\
\iff & (\boldsymbol{\Gamma}) \in \text{local minima}\,(\mathfrak{B}) \\
\iff & (\boldsymbol{\Gamma}) \in \text{global minima}\,(\mathfrak{B}) \qquad \text{(convexity)}
\end{aligned}$$

$\boldsymbol{\Gamma}$ is a one-to-one function of $\boldsymbol{\omega}, \boldsymbol{\Omega}$, and s, where s is somewhat like a slack variable. We show that local minima always occur at $s = 1$, and furthermore that at local minima, the value of the objective function of $\mathfrak{A}$ equals the value of the objective function of $\mathfrak{B}$ for the corresponding parameters. Thus, all local minima of $\mathfrak{A}$ must obtain the same value, i.e. they are global minima. In fact, because the transformation is one-to-one, and $\mathfrak{B}$ has a unique global minimum, there is exactly one local minimum of $\mathfrak{A}$.

The transformation we use is:

$$\begin{aligned}
\boldsymbol{x} \sim \mathcal{N}\left(\boldsymbol{\omega}, \boldsymbol{\Omega}^{-1}\right) \to \begin{bmatrix} \boldsymbol{x} \\ 1 \end{bmatrix} &\sim \mathcal{N}\left(\mathbf{0}, \boldsymbol{\Gamma}^{-1}\right) \\
\boldsymbol{\Gamma} &= \begin{bmatrix} \boldsymbol{\Omega} & -\boldsymbol{\Omega}\boldsymbol{\omega} \\ -\boldsymbol{\omega}^T\boldsymbol{\Omega} & \frac{1}{s} + \boldsymbol{\omega}^T\boldsymbol{\Omega}\boldsymbol{\omega} \end{bmatrix} \\
s &> 0
\end{aligned}$$

We first came across a similar transformation in a paper by Hosseini and Sra (2015), although their transformation was for the covariance matrix; this transform here can be derived from theirs using block matrix inversion. The variable $\frac{1}{s}$ is used to allow $\boldsymbol{\Gamma}$ to be any positive definite matrix (subject

to constraints on $\mathbf{\Omega}, \boldsymbol{\omega}$). Otherwise, the value of the bottom right entry would be determined by the other entries. It should be clear that, as long as $\mathbf{\Omega}$ is positive definite, this transformation is one-to-one given a fixed s. If it is merely positive semidefinite, we cannot guarantee the global minimum is unique (but all local minima remain global); typically, solvers for Kronecker-sum-structured $\mathbf{\Omega}$ require it to be positive definite.

The strategy used to prove the chain of implications is to show that, when $s = 1$, $\mathfrak{A}$ and $\mathfrak{A}'$ correspond to the same optimization problem. Since $(\mathbf{\Omega}, \boldsymbol{\omega}, s) \to \mathbf{\Gamma}$ is a one-to-one mapping, as well as the fact that this mapping preserves the objective function's values, and finally the fact that $\mathbb{K}_{\mathbf{Mv}}$ is a linear subspace and hence preserves strict convexity of $\mathfrak{B}$, we have that there is exactly one minimum to $\mathfrak{B}$ which by the chain of implications is also the unique minimum of $\mathfrak{A}$.

All such claims are proven in the appendix. In particular, uniqueness of the global minimum holds when one uses convex regularization penalties on $\mathbf{\Omega}$, and furthermore holds (subject to a reasonable constraint on $\boldsymbol{\omega}$) when one restricts $\mathbf{\Omega}$ to be a low-rank matrix, such as that considered by a variant of GmGM Andrew et al., 2024b. The proof in the latter case is considerably more complicated, requiring mathematical machinery particularly from work by Hartwig (1976) and Holbrook (2018); it holds when we restrict $\boldsymbol{\omega}$ to be orthogonal to the nullspace of $\mathbf{\Omega}$ (a reasonable assumption since data generated from the distribution should also be orthogonal to the nullspace).

We claimed that our block coordinate descent algorithm converges to this unique optimal value. Block coordinate descent methods are not guaranteed to converge for smooth nonconvex problems. However, when there is a unique minimum for each block, as is the case here, block coordinate descent is known to converge Luenberger and Ye, 2008; Tseng, 2001.

For simplicity, we report runtimes in which all dimensions of the input tensor with k dimensions are the same size n. The runtime complexity of our model is $O(I(G + M))$, where I is the number of iterations, G is the cost of graph estimation, and M is the cost of mean estimation. There are several $O(n^3 + n^k)$ algorithms for graph estimation, such as TeraLasso. Our mean estimation step can also naively be done in $O(n^3)$ due to it requiring matrix inverses[3]. Thus, complexity is $O(I(n^3 + n^k))$. It's natural to ask how large I needs to be; in the supplementary material, we show empirically that $I = 2$ or 3 is usually sufficient.

3.4 Results

Figure	Mean Distribution
Figure 3.1	Constant mean of 1
Figure 3.3	$m + \widetilde{\bigoplus}_\ell \boldsymbol{\mu}_\ell$
	$\boldsymbol{\mu}_\ell \sim \mathcal{N}(\mathbf{0}, \mathbf{I}),\ m \sim \mathcal{N}(0, 1)$
Figure 3.4 (left)	$\boldsymbol{\omega} \sim \mathcal{N}(\mathbf{0}, \frac{1}{20}\mathbf{I})$
Figure 3.4 (right)	$\frac{\boldsymbol{\omega}}{14}$
	$\boldsymbol{\omega}_{ij} \sim \text{Poisson}(10)$

Table 3.1: The distributions of the means in each of our synthetic data experiments, and the corresponding figures. Both mean distributions for Figure 3.4 have a variance of 0.05. This variance had to be smaller for the unstructured distributions, otherwise the noise would overwhelm the signal in all algorithms.

All experiments were run on a 2020 MacBook Pro with an M1 chip and 8 GB of RAM. Our method was implemented using NumPy 1.25.2 Harris et al., 2020 and SciPy 1.12.0 Virtanen et al., 2020; for precision matrix routines, we used Greenewald et al.'s reference implementation of TeraLasso (2019) as well as GmGM 0.5.3. GmGM and TeraLasso are fairly similar algorithms; GmGM outputs positive definite matrices and does not use regularization, while TeraLasso uses regularization and does not require the outputs to be positive definite (as long as the Kronecker sum of the outputs is). By presenting results with each algorithm, we hope to show that our proposed mean estimation algorithm

[3]Most graph estimation algorithms produce the eigendecomposition of $\mathbf{\Psi}_\ell$, and hence $\mathbf{A}_\ell$, as a subroutine. Assuming this is then passed to the mean estimation step, mean estimation is only $O(n^2)$.

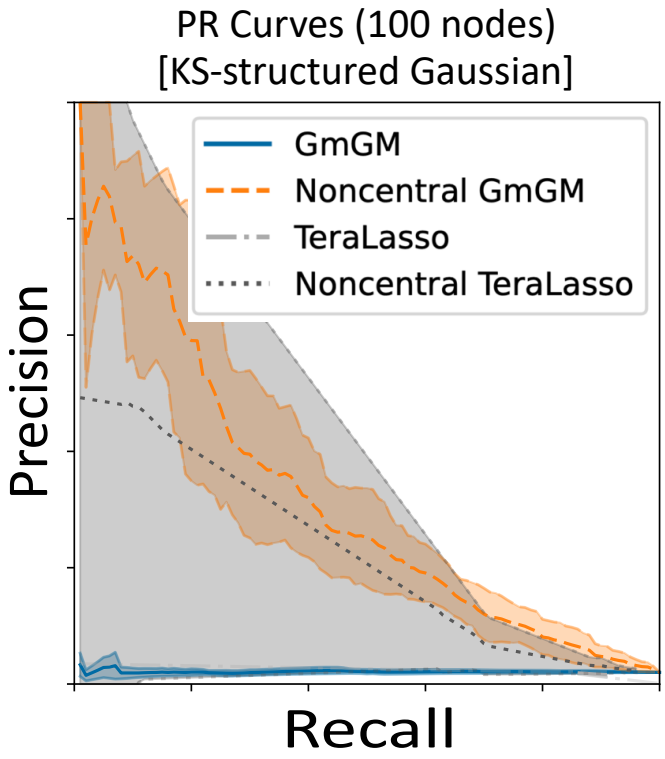


Figure 3.3: Precision and recall for on a synthetic dataset in which data was generated from a noncentral Kronecker-sum normal distribution. Noncentral TeraLasso's high susceptibility to changes in the regularization parameter led to our grid search being too coarse-grained during some trials.

is not specific to any one method of precision estimation. All code is provided in the supplementary material.

In this section, we compare the performance of precision matrix algorithms with and without our mean estimation wrapper across a variety of test suites. We first compare performances on synthetic data, in which the graphs are generated from a Barabasi-Albert (power-law) distribution. We generate means for our synthetic data from a variety of distributions; see Table 3.1. We also compare them on Erdos-Renyi graphs, achieving similar results, but for conciseness we defer those results to the appendix. Next, we compare performances on the real-world COIL-20 video dataset (Nene et al., 1996), demonstrating clear improvements. Finally, we show that, without properly taking into account the mean, precision matrix estimation will yield clearly wrong results on transcriptomics datasets such as the mouse embryo stem cell 'E-MTAB-2805' dataset Buettner et al., 2015.

For synthetic data, we saw the dramatic deterioration in performance of algorithms that make the zero-mean assumption in Figure 3.1, while mean-corrected algorithms still perform well. Zero-mean algorithms also perform poorly when we generate data from the noncentral Kronecker-sum-structured Gaussian distribution; understandably, our correction is immune to this problem (Figure 3.3).

Finally, we may wonder what happens if our mean has absolutely no structure, with every element of our generated matrix having a different, independent mean. We compared both Gaussian and Poisson-distributed random means (Figure 3.4) and found that there was still a drop-off in performance for Poisson means, but not nearly as dramatic as that seen in the other cases considered. There was no drop-off in the Gaussian case.

When we increased the variance of our means in our unstructured tests, both the standard algorithms and our mean-corrected algorithms dropped in performance equally. We suspect this is because, for large variances, the signal is overwhelmed by the noise. For small variances, since each row/column of the mean is uncorrelated, their effects average to zero, and thus our mean-corrected algorithms will also estimate means of zero (making them identical to the uncorrected case). The Poisson distribution behaves slightly differently, as the mean is instead a constant nonzero number, and hence our mean-

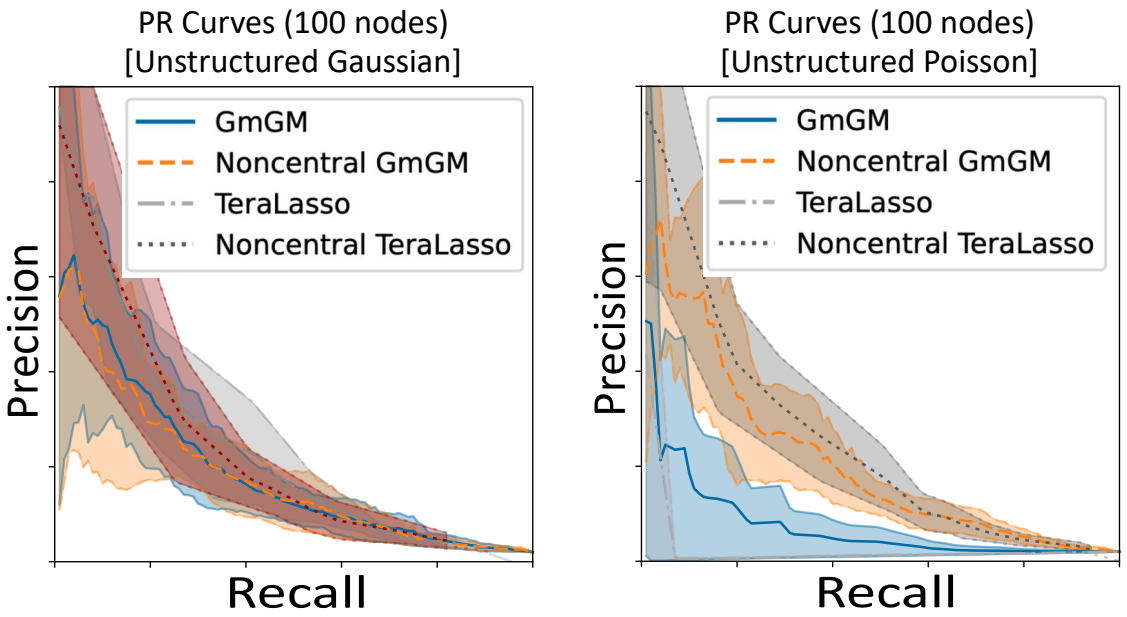


Figure 3.4: Precision and recall for on a synthetic dataset in which data was generated from a zero-mean Kronecker-sum normal distribution, with independent Gaussian (left) and Poisson (right) noise added to every element. Error bars are the best/worst performance over 10 trials; the center line is average performance.

corrected algorithm is able to correctly account for this, whereas the standard algorithms experience degradation in performance similar to that of the constant-mean offset test (Figure 3.1).

The unstructured mean experiment has important consequences: if real-world data has an unstructured mean, then it is less important to correct for the means! However, it is likely that the mean does have a structured component, as each axis will have its own latent factors affecting the outcome in addition to any unstructured noise that may or may not exist. To demonstrate this, we run our algorithm on two real-world datasets, showing that our mean correction truly does improve performance in practice.

The COIL-20 dataset consists of 20 videos (each with 72 frames) of objects rotating 360 degrees; it has become somewhat customary for multi-axis methods to present their performance on the rubber duck object of this dataset. The dataset is chosen because of its ease to validate: frames that are close in time (i.e. adjacent) should be connected in the graph. We present our results graphically in Figure 3.6, from which we can easily see the improvement. We pre-processed the data using the

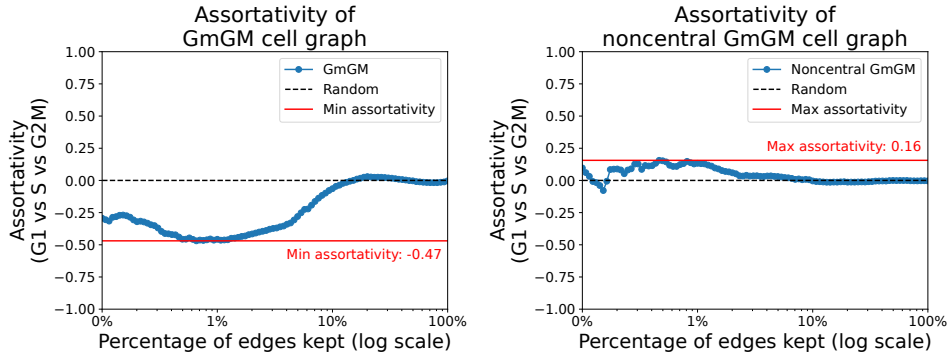


Figure 3.5: Assortativity of standard GmGM and our mean-corrected GmGM as we vary the number of edges kept, on the E-MTAB-2805 cell-cycle dataset. The cells were labeled by their stage in the cell cycle (G1/S/G2M).

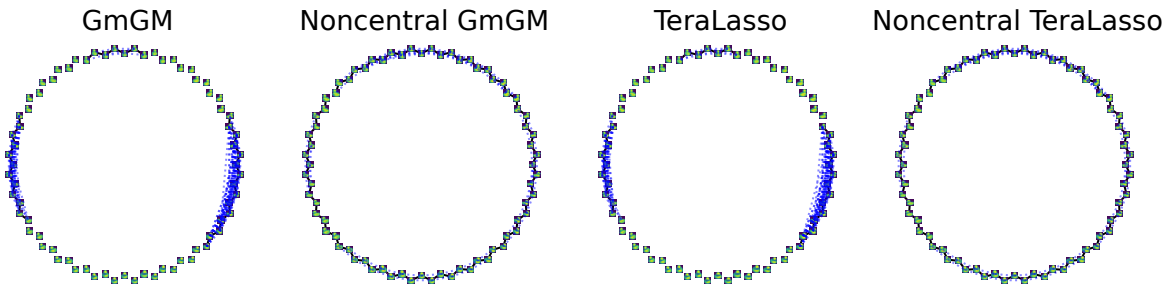


Figure 3.6: Each circle represents the results of an algorithm on the COIL-20 dataset; 'noncentral' denotes our mean estimation method. Dashed blue lines indicate connections between non-adjacent frames; solid black lines indicate adjacent connections.

nonparanormal skeptic Liu et al., 2012 to relax the Gaussian assumption of the models, and we chose thresholding/regularization parameters such that there would be approximately 144 edges.

We considered 'correct' edges to be those connecting adjacent frames or frames with a single frame in between. GmGM had 59/147 (40%) correct and TeraLasso had 55/142 (39%) correct. Once we wrap the methods in our mean estimation procedure, this rose to 127/147 (86%) and 126/141 (89%), respectively. Furthermore, all 'wrong' connections of our noncentral algorithms were nearly correct: with one exception, every connection was between frames with at most two frames between them. For the standard algorithms, more than a third of all edges failed this criteria. The noncentral algorithms also had high recall, at 88% for both noncentral GmGM and TeraLasso. The uncorrected algorithms had a recall of 41% and 38%. Overall, the results for this experiment are conclusively in favor of using our correction.

Our final experiment is on the E-MTAB-2805 scRNA-seq dataset. This dataset consists of 288 mouse embryo stem cells and 34,573 unique genes, with each cell being labeled by its stage in the cell cycle (G1, S, or G2M). It has previously been considered by the creators of scBiGLasso Li et al., 2022, in which they limited it to a hand-selected set of 167 mitosis-related genes - we limited it to the same set of genes. We would expect that cells at the same stage in the same cycle should have some similarities, and hence should have some tendency to cluster together in our learned graphs.

We can use 'assortativity' as a measure for the tendency for cells within a stage to connect: assortativity ranges from -1 to 1. When it is positive, it indicates the tendency of nodes in the same cell cycle stage to connect to each other. If it is negative, cells in differing cell cycle stages tend to be connected to each other. If zero, there is no tendency; the connections are random.

Without correcting for the mean, we find that there is a very strong tendency for cells in the same cell cycle *to not connect* (assortativity = -0.47). This runs contrary to intuition: cells with some factor in common should not be caused by that same factor to look different. When correcting for the means, we see a modest positive tendency (assortativity = 0.16), which is more reasonable. Figure 3.5 shows the change in assortativity for GmGM, with and without our correction, as we vary the threshold of how many edges to keep.

3.5 Conclusion

Current Kronecker-structured models do not take into account the mean of the data. As one might expect, this can lead to wildly incorrect inferences (Figures 3.1 and 3.5). In fact, it can result in inferences directly opposite to reality: as thresholding/regularization increased, zero-mean algorithms decreased in both precision and recall, and identified an extremely strong (and incorrect) repulsive force between cells in the same stage of the cell cycle.

In this paper, we have demonstrated that this can be fixed by adding a mean-estimating wrapper around standard algorithms. While the means and precisions are nonseperable in the KS-structured case, and hence coincide with a nonconvex optimization problem, we proved that there is a unique global optimum, to which our algorithm converges. We implemented our algorithm as a 'drop-in' wrapper (for easy use with any standard precision-estimating algorithm), but with minor modifications

it can be made to preserve the 'only-one-eigendecomposition' property of GmGM.

Bibliography for Paper 2

Allen, G. I., & Tibshirani, R. (2010). Transposable regularized covariance models with an application to missing data imputation. *The annals of applied statistics*, *4*(2), 764–790. https://doi.org/10.1214/09-AOAS314

Andrew, B., Westhead, D. R., & Cutillo, L. (2024a, January). GmGM: A fast multi-axis Gaussian graphical model [Conference Name: The 27th International Conference on Artificial Intelligence and Statistics ISSN: 2640-3498 Meeting Name: The 27th International Conference on Artificial Intelligence and Statistics Place: València, Spain Publisher: Proceedings of Machine Learning Research]. Retrieved July 16, 2024, from https://eprints.whiterose.ac.uk/208283/

Andrew, B., Westhead, D. R., & Cutillo, L. (2024b, July). Making Multi-Axis Gaussian Graphical Models Scalable to Millions of Samples and Features [arXiv:2407.19892 [cs, q-bio, stat]]. https://doi.org/10.48550/arXiv.2407.19892

Buettner, F., Natarajan, K. N., Casale, F. P., Proserpio, V., Scialdone, A., Theis, F. J., Teichmann, S. A., Marioni, J. C., & Stegle, O. (2015). Computational analysis of cell-to-cell heterogeneity in single-cell RNA-sequencing data reveals hidden subpopulations of cells [Number: 2 Publisher: Nature Publishing Group]. *Nature Biotechnology*, *33*(2), 155–160. https://doi.org/10.1038/nbt.3102

Dutilleul, P. (1999). The mle algorithm for the matrix normal distribution [Publisher: Taylor & Francis _eprint: https://doi.org/10.1080/00949659908811970]. *Journal of Statistical Computation and Simulation*, *64*(2), 105–123. https://doi.org/10.1080/00949659908811970

Friedman, J., Hastie, T., & Tibshirani, R. (2008). Sparse inverse covariance estimation with the graphical lasso. *Biostatistics*, *9*(3), 432–441. https://doi.org/10.1093/biostatistics/kxm045

Greenewald, K., Zhou, S., & Hero, A. (2019). Tensor graphical lasso (TeraLasso) [Publisher: [Royal Statistical Society, Oxford University Press]]. *Journal of the Royal Statistical Society. Series B (Statistical Methodology)*, *81*(5), 901–931. Retrieved July 16, 2024, from https://www.jstor.org/stable/26833080

Harris, C. R., Millman, K. J., van der Walt, S. J., Gommers, R., Virtanen, P., Cournapeau, D., Wieser, E., Taylor, J., Berg, S., Smith, N. J., Kern, R., Picus, M., Hoyer, S., van Kerkwijk, M. H., Brett, M., Haldane, A., del Río, J. F., Wiebe, M., Peterson, P., ... Oliphant, T. E. (2020). Array programming with NumPy [Number: 7825 Publisher: Nature Publishing Group]. *Nature*, *585*(7825), 357–362. https://doi.org/10.1038/s41586-020-2649-2

Hartwig, R. E. (1976). Singular Value Decomposition and the Moore–Penrose Inverse of Bordered Matrices [Publisher: Society for Industrial and Applied Mathematics]. *SIAM Journal on Applied Mathematics*, *31*(1), 31–41. https://doi.org/10.1137/0131003

Holbrook, A. (2018). Differentiating the pseudo determinant. *Linear Algebra and its Applications*, *548*, 293–304. https://doi.org/10.1016/j.laa.2018.03.018

Hosseini, R., & Sra, S. (2015). Matrix Manifold Optimization for Gaussian Mixtures. *Advances in Neural Information Processing Systems*, *28*. Retrieved July 16, 2024, from https://papers.nips.cc/paper_files/paper/2015/hash/dbe272bab69f8e13f14b405e038deb64-Abstract.html

Kalaitzis, A., Lafferty, J., Lawrence, N. D., & Zhou, S. (2013). The Bigraphical Lasso [ISSN: 1938-7228]. *Proceedings of the 30th International Conference on Machine Learning*, 1229–1237. Retrieved July 16, 2024, from https://proceedings.mlr.press/v28/kalaitzis13.html

Kolda, T. G., & Bader, B. W. (2009). Tensor Decompositions and Applications [Publisher: Society for Industrial and Applied Mathematics]. *SIAM Review*, *51*(3), 455–500. https://doi.org/10.1137/07070111X

Li, S., López-García, M., Lawrence, N. D., & Cutillo, L. (2022, March). Scalable Bigraphical Lasso: Two-way Sparse Network Inference for Count Data [arXiv:2203.07912 [cs, stat]]. https://doi.org/10.48550/arXiv.2203.07912

Liu, H., Han, F., Yuan, M., Lafferty, J., & Wasserman, L. (2012). The nonparanormal SKEPTIC. *Proceedings of the 29th International Conference on Machine Learning, ICML 2012*, 1415–1422. Retrieved July 16, 2024, from https://collaborate.princeton.edu/en/publications/the-nonparanormal-skeptic

Luenberger, D. G., & Ye, Y. (2008). *Linear and Nonlinear Programming* (Vol. 116). Springer US. https://doi.org/10.1007/978-0-387-74503-9

McInnes, L., Healy, J., & Melville, J. (2020, September). UMAP: Uniform Manifold Approximation and Projection for Dimension Reduction [arXiv:1802.03426 [cs, stat]]. https://doi.org/10.48550/arXiv.1802.03426

Nene, S. A., Nayar, S. K., & Murase, H. (1996). Columbia Object Image Library (COIL-20).

Tseng, P. (2001). Convergence of a Block Coordinate Descent Method for Nondifferentiable Minimization. *Journal of Optimization Theory and Applications*, *109*(3), 475–494. https://doi.org/10.1023/A:1017501703105

Virtanen, P., Gommers, R., Oliphant, T. E., Haberland, M., Reddy, T., Cournapeau, D., Burovski, E., Peterson, P., Weckesser, W., Bright, J., van der Walt, S. J., Brett, M., Wilson, J., Millman, K. J., Mayorov, N., Nelson, A. R. J., Jones, E., Kern, R., Larson, E., … van Mulbregt, P. (2020). SciPy 1.0: Fundamental algorithms for scientific computing in Python [Number: 3 Publisher: Nature Publishing Group]. *Nature Methods*, *17*(3), 261–272. https://doi.org/10.1038/s41592-019-0686-2

Wang, Y., Jang, B., & Hero, A. (2020). The Sylvester Graphical Lasso (SyGlasso) [ISSN: 2640-3498]. *Proceedings of the Twenty Third International Conference on Artificial Intelligence and Statistics*, 1943–1953. Retrieved July 16, 2024, from https://proceedings.mlr.press/v108/wang20d.html

Chapter 4

The strong product model for network inference without independence assumptions

This paper was written at the same time as the previous paper (Chapter 3). We had recently improved the scalability of GmGM significantly (Chapter 2), and had encouraging preliminary results for further scalability improvements (which would go on to form the basis of Chapter 6). With these improvements, we felt that multi-axis models were now feasible to use on real data - but were they the right tool for the job?

In light of this question, we began to more closely consider the fundamental assumptions our models were making: the zero-mean assumption (Chapter 3), the Gaussian copula assumption[1], and the Kronecker-sum assumption. It is this last assumption that this paper addresses.

As a reminder, the Kronecker-sum assumption has a very natural interpretation: genes can only be conditionally dependent other genes within the same cell (or the same gene in different cells). While we would expect the strongest dependencies to abide by this interpretation, it is easy to construct real examples where this fails - genes involved in signalling pathways would certainly have cross-cell effects.

In light of this, we replaced the Kronecker-sum assumption with a new assumption meant to capture cross-cell effects. The results were disappointing; accounting for cross-cell effects did not yield significant improvement over Kronecker-sum work. There was, however, a silver lining: it justifies the use of the Kronecker sum. While there may be some other potential model that could better account for cross-cell effects, we took the results of this paper as suggesting we were on the right track by working with the Kronecker sum. In light of this, the remaining two papers in this thesis were built on Kronecker-sum-based models[2].

The paper was published in the Proceedings of the International Conference on Artificial Intelligence and Statistics in 2025. It is presented unchanged from the published version, except for formatting changes to fit the thesis style. The supplementary material for the paper is available in Appendix C, which contains further details and experiments. For further details and experiments that came up during the viva, see Appendix I.

[1]Which we began to address, but did not have time to finish during this PhD - see Discussion Section 7.3.4

[2]We did look in to how one would approach fitting other models, which we discuss in Discussion Section 7.3.3.

The Strong Product Model for Network Inference without Independence Assumptions

Bailey Andrew, David R. Westhead, and Luisa Cutillo

Published in the Proceedings of the 28th International Conference on Artificial Intelligence and Statistics

Abstract. Multi-axis graphical modelling techniques allow us to perform network inference without making independence assumptions. This is done by replacing the independence assumption with a weaker assumption about the interaction between the axes; there are several choices for which assumption to use. In single-cell RNA sequencing data, genes may interact differently depending on whether they are expressed in the same cell, or in different cells. Unfortunately, current methods are not able to make this distinction. In this paper, we address this problem by introducing the strong product model for Gaussian graphical modelling.

4.1 Introduction

It is often the case that one wishes to learn a network ('graph') from their data. A common example is learning a gene regulatory network from a single-cell RNA-sequencing (scRNA-seq) dataset. This type of dataset takes the form of a matrix whose rows represent cells and whose columns represent genes, and whose entries quantify how much a given gene is expressed in a given cell. There are numerous methods that perform network inference across a range of scenarios, such as the Graphical Lasso ('*glasso*', Friedman et al., 2008), neighborhood selection (Meinshausen & Bühlmann, 2006), and SIMoNe (Ambroise et al., 2009).

However, such methods make an independence assumption: learning the gene regulatory network requires assuming that the cells in your body do not interact! This assumption is clearly faulty. There are two fundamentally different types of dependencies existing in our dataset - the dependencies between genes, and the dependencies between cells. More generally, matrix-variate datasets have row-dependencies and column-dependencies.

We can replace the independence assumption with a weaker assumption about how these two types of dependencies interact. We can imagine a larger graph of dependencies between matrix elements, which is some product graph of the row-dependencies and column-dependencies - for example, by using the Cartesian product. We will make this more formal in Section 4.2. In scRNA-seq data, the larger graph corresponds to connections between (cell, gene) pairs.

There are many choices of graph product to use, but none of them make the distinction between within-row and between-row interactions. A gene X might up-regulate another gene Y when both are present in the same cell, but not have any affect on the expression of Y in some other cell (due to the gene transcript never leaving the cell membrane). Likewise, gene X may *only* affect the expression of gene Y in different cells; perhaps gene X encodes a surface protein or causes an enzyme to be secreted which affects the behavior of nearby cells.

By taking the differences between within-row and between-row interactions into account, we can arrive at more biologically meaningful results. In this paper, we develop a model, inspired by the 'strong product' graph product, that successfully differentiates and learns these types of dependencies.

Decomposition	Definition	Justification	Rule
Kronecker Product	$\mathbf{\Psi}_{\text{rows}} \otimes \mathbf{\Psi}_{\text{cols}}$	Both	$\text{row}[x] \sim \text{row}[y] \wedge \text{col}[x] \sim \text{col}[y]$
Kronecker Sum	$\mathbf{\Psi}_{\text{rows}} \oplus \mathbf{\Psi}_{\text{cols}}$	Graph Product	either $\text{row}[x] \sim \text{row}[y] \wedge \text{col}[x] = \text{col}[y]$ or $\text{row}[x] = \text{row}[y] \wedge \text{col}[x] \sim \text{col}[y]$
Sylvester Product	$(\mathbf{\Psi}_{\text{rows}} \oplus \mathbf{\Psi}_{\text{cols}})^2$	Generative Process	N/A

Table 4.1: A comparison of the three decompositions in common use. The Kronecker product has a long history of use, but its use in graphical modelling comes from Tsiligkaridis et al. (2012); the other two are from Kalaitzis et al. (2013) and Wang et al. (2020), respectively. The 'rule' column represents the procedure by which to derive the connections between matrix elements x and y in the full graph $\mathbf{\Omega}$; $\sim$ is used as a shorthand for 'is connected to'. The Sylvester Product can also be expressed as a graph product, but one needs to use second-order connections of the axis graphs to do so; its primary motivation is the generative process that yields it.

4.2 Background

One of the most popular methods of network inference is the Graphical Lasso (Friedman et al., 2008). The general idea is to perform regularized maximum likelihood estimation for the inverse covariance matrix (the 'precision matrix') of a Gaussian distribution. Accordingly, the method assumes your data is Gaussianly distributed, but this can be replaced with the weaker 'Gaussian copula' assumption through preprocessing tools such as the nonparanormal skeptic (Liu et al., 2012). Intuitively, the nonparanormal skeptic allows these techniques to be used on data with arbitrary marginals, as long as relationships between the datapoints 'behave Gaussianly'.

$$\underset{\mathbf{\Psi}}{\text{argmin}} \;\; \log|\mathbf{\Psi}| - \text{tr}\left[\mathbf{\Psi}\mathbf{x}\mathbf{x}^T\right] + \rho\|\mathbf{\Psi}\|_{1,od} \qquad \text{(Graphical Lasso)}$$

The choice to model the data as a Gaussian is not just for computational convenience; it has the convenient theoretical property that $\mathbf{\Psi}$ encodes the *graph of conditional dependencies*. Let x, y be two variables from a dataset $\mathcal{D}$ ($\mathcal{D}_{\backslash x \backslash y} = \mathcal{D} - \{x, y\}$), then we have the following relation for Gaussian data:

$$\mathbb{P}\left(x, y|\mathcal{D}_{\backslash x \backslash y}\right) = \mathbb{P}\left(x|\mathcal{D}_{\backslash x \backslash y}\right)\mathbb{P}\left(y|\mathcal{D}_{\backslash x \backslash y}\right) \iff \mathbf{\Psi}_{xy} = 0$$

Conditional dependencies encode the concept of independence after conditioning out the rest of the dataset. Intuitively, it captures the *direct relation* of two variables on each other - this is in contrast to correlation, which captures indirect relations as well. The classic example is the correlation between shark attacks and ice cream sales due to the confounding factor of it being summer; conditioning out the season, the two variables become independent.

Graphs of conditional dependencies should typically be sparse, by nature of their capturing the direct effect. This forms part of the justification for the use of regularization penalties, but penalties are also necessary since in the small-sample scenario the unregularized maximum likelihood estimate will typically not exist.

A drawback of the Graphical Lasso is that it assumes independence of the samples. In reality, there may be dependencies between rows $\mathbf{\Psi}_{\text{rows}}$ and dependencies between columns $\mathbf{\Psi}_{\text{cols}}$. To solve this, we can 'vectorize' our data and consider the graph of (row, column) pairs (i.e. the graph connecting all matrix elements).

The vectorization of a $d_1 \times d_2$ matrix $\mathbf{X}$, denoted $\text{vec}[\mathbf{X}]$, is the 'flattening' of the matrix into a length $d_1 d_2$ vector. Whether this flattening is done in a rows-first or columns-first manner does not matter as long as it is done consistently. We can view our data as being a set of d_1 length-d_2 samples $\mathbf{x}_i$ or alternatively as a $d_1 \times d_2$ matrix $\mathbf{X}$.

$$\mathbf{x}_i \underset{i.i.d.}{\sim} \mathcal{N}\left(\mathbf{0}, \mathbf{\Psi}_{\text{cols}}^{-1}\right) \qquad \text{(independence assumption)}$$

$$\text{vec}[\mathbf{X}] \sim \mathcal{N}\left(\mathbf{0}, \mathbf{\Omega}^{-1}\right) \qquad \text{(no assumption)}$$

The first model considers each row to be a separate sample, and each column corresponds to an element of this vector. The second model considers our entire dataset to be a single sample, and each element of our dataset is a (row, column) pair. We can then discuss the dependencies of such pairs, $\mathbf{\Omega}$, rather than being limited to column-column dependencies.

While this removes the independence assumption, it comes with its own drawbacks. In the independence scenario, we have d_1 samples of d_2 features, but in the non-independence scenario we have 1 sample of $d_1 d_2$ features! $\mathbf{\Omega}$ has $O(d_1^2 d_2^2)$ parameters to learn, and we must do so with a single sample. Not only is this intractable from a computational complexity perspective, but also from a statistical perspective; how could we hope to have any degree of confidence in our results?

To resolve the problem, we can impose some structure on $\mathbf{\Omega}$, such as it being composed out of the axis-wise graphs: $\mathbf{\Omega} = \zeta(\mathbf{\Psi}_{\text{rows}}, \mathbf{\Psi}_{\text{cols}})$. Several choices for ζ exist, which we describe in Table 4.1. They can broadly be separated into two categories; those based on graph products and those based on generative processes. They all use the Kronecker product $\otimes$ as a building block, and some further use the Kronecker sum $\oplus$: $\mathbf{A} \oplus \mathbf{B} = \mathbf{A} \otimes \mathbf{I} + \mathbf{I} \otimes \mathbf{B}$. Our work exists within the framework of Kronecker-separable modelling; for an overview, see the review by Wang et al. (2022).

Of the decompositions in prior work, two admit a straightforward interpretation as an operation on graphs: the Kronecker product corresponds to the tensor product of graphs, while the Kronecker sum corresponds to the Cartesian product. The Kronecker product only takes into account between-row connections, i.e. to determine if two genes interact, we only consider the relationship of their expressions in different cells. The Kronecker sum only takes into account within-row connections: to determine if two genes interact, we only consider the relationship of their expressions in the same cell. Ideally, we would be able to capture both types of relationships.

The Sylvester product, when forced into the framework of graph products, does capture both types of relationships - but only through 'second-order' connections in the graph (by 'second order', we mean that $x \sim y \sim z$ implies x and z have a second-order connection, where $\sim$ is notation for 'is connected to'). This is difficult to interpret through the lens of conditional dependency, where we are after the direct connections.

4.3 Our contributions

In this paper, we introduce a new graph decomposition framework for conditional dependency estimation (Section 4.4.2). It is based on the 'strong product' from graph theory, and hence we call our model the 'strong product model'. We show that it captures and distinguishes between both the within-row and between-row dependencies, which is particularly beneficial in use cases such as scRNA-seq. We demonstrate that one can perform inference with this model efficiently (Section 4.4.3), and prove that there exists a unique solution to the problem (Section 4.4.7). Finally, we demonstrate practical utility on real-world datasets (Sections 4.5.2 and 4.5.3). These results hold *even in the single-sample case*.

4.4 Methodology

This section will focus on the derivation of the strong product model. We will first introduce our notation in Section 4.4.1. Then, we will motivate the structure of our model in Section 4.4.2. Next, we will derive an efficient maximum likelihood estimator (MLE) in Section 4.4.3, and finally in Sections 4.4.6 and 4.4.7 we will prove key properties of our algorithm to guarantee that its solutions are reasonable and well-founded.

4.4.1 Notation

In this paper, lower-case letters a represent scalars, lower-case bold $\mathbf{v}$ represent vectors, and upper-case bold $\mathbf{M}$ represent matrices. $\mathbf{I}_a$ represents an $a \times a$ identity matrix, although we will frequently omit the subscript as the shape will be clear from context. d_1 will represent the number of rows in the dataset, and d_2 will represent the number of columns.

We will denote the precision matrix of the rows as $\mathbf{\Psi}_1$. The precision matrix of the columns will be split into two parts, those that focus on the within-row dependencies $\mathbf{\Psi}_{2,w}$ and the between-row dependencies $\mathbf{\Psi}_{2,b}$. It will often be convenient to consider the matrix $\mathbf{\Theta} = \mathbf{\Psi}_{2,b} + \mathbf{I}$. We will let $\mathbf{\Omega}$ represent the whole precision matrix, which we are assuming can be factorized as $\mathbf{\Psi}_1 \oplus \mathbf{\Psi}_{2,w} + \mathbf{\Psi}_1 \otimes \mathbf{\Psi}_{2,b}$.

We also define the empirical gram matrices $\mathbf{S}_1, \mathbf{S}_2$, which will represent sufficient statistics for our model. These are defined as $\mathbf{S}_1 = \mathbf{X}\mathbf{X}^T$ and $\mathbf{S}_2 = \mathbf{X}^T\mathbf{X}$, where $\mathbf{X}$ is the data matrix. Finally, we define the stridewise trace tr^{d_1} and the blockwise trace tr_{d_2}, which show up frequently when working with derivatives involving log-determinants of Kronecker products (below, $\mathbf{J}^{ij}$ is the matrix of zeros except for a 1 at (i, j)).

$$\mathrm{tr}^{d_1}\left[\mathbf{A}\right]_{ij} = \mathrm{tr}\left[\mathbf{A}\left(\mathbf{I}\otimes\mathbf{J}^{ij}\right)\right]$$
$$\mathrm{tr}_{d_2}\left[\mathbf{A}\right]_{ij} = \mathrm{tr}\left[\mathbf{A}\left(\mathbf{J}^{ij}\otimes\mathbf{I}\right)\right]$$

4.4.2 The strong product model

As mentioned in Section 4.2, the Kronecker sum $\mathbf{\Psi}_1 \oplus \mathbf{\Psi}_2$ and Kronecker product $\mathbf{\Psi}_1 \otimes \mathbf{\Psi}_2$ only take into account the within-row dependencies (Kronecker sum) and between-row dependencies (Kronecker product), rather than both simultaneously. It is tempting to add these together: $\mathbf{\Psi}_1 \oplus \mathbf{\Psi}_2 + \mathbf{\Psi}_1 \otimes \mathbf{\Psi}_2$, and the instinct turns out to be correct. This corresponds to the *strong product* from graph theory. If we split $\mathbf{\Psi}_2$ into separate matrices, each measuring either the within- or between- row dependencies, then we arrive at $\mathbf{\Psi}_1 \oplus \mathbf{\Psi}_{2,w} + \mathbf{\Psi}_1 \otimes \mathbf{\Psi}_{2,b}$. It will turn out to be more convenient to parameterize this in terms of $\mathbf{\Theta} = \mathbf{\Psi}_{2,b} + \mathbf{I}$, i.e. as $\mathbf{\Psi}_1 \otimes \mathbf{\Theta} + \mathbf{I} \otimes \mathbf{\Psi}_{2,w}$.

$$\mathrm{vec}\left[\mathbf{X}\right] \sim \mathcal{N}\left(\mathbf{0}, \left(\mathbf{\Psi}_1 \otimes \mathbf{\Theta} + \mathbf{I} \otimes \mathbf{\Psi}_{2,w}\right)^{-1}\right) \qquad \text{(Strong product model)}$$
$$\begin{aligned}\mathrm{NLL} = &-\frac{1}{2}\log\left|\mathbf{\Psi}_1 \otimes \mathbf{\Theta} + \mathbf{I} \otimes \mathbf{\Psi}_{2,w}\right| \\ &+ \frac{1}{2}\mathrm{tr}\left[\mathbf{\Psi}_{2,w}\mathbf{S}_2\right] + \frac{1}{2}\mathrm{tr}\left[\mathbf{\Psi}_1\mathbf{X}\mathbf{\Theta}\mathbf{X}^T\right]\end{aligned} \tag{4.1}$$

We will restrict all $\mathbf{\Psi}$ to be positive definite. This is a common, but not universal, restriction. Precision matrices must be positive definite (so $\mathbf{\Psi}_1 \otimes \mathbf{\Theta} + \mathbf{I} \otimes \mathbf{\Psi}_{2,w}$ must be), but in theory we do not have to enforce positive definiteness of the factor matrices. The difference can have theoretical implications - not everything that is representable as a composition of non-pos.-def. matrices is representable as a composition of pos. def. matrices (Song & Hero, 2023). We enforce positive definiteness as it leads to better interpretability (the matrices correspond to real precision matrices).

Positive-definiteness of the factor matrices comes at a price. Any sparsifying regularizer (whether it be Lasso or a non-convex regularizer such as SCAD (Fan & Li, 2001) or MCP (Zhang, 2010)) have proximal operators that output indefinite matrices for positive definite inputs - both in theory (Guillot & Rajaratnam, 2015) and in practice (Xue et al., 2012). Furthermore, the unregularized model is not convex, but it is *geodesically convex* (see Section 4.4.7 for details), whereas the Lasso penalty is convex, but not geodesically convex (Duembgen & Tyler, 2016). This mismatch makes establishing useful properties such as existence and uniqueness harder. Here, we do not consider sparsifying regularizers.

In this paper, we follow Deng and Tsui (2013) by imposing a Frobenius norm penalty $\|\log \mathbf{\Gamma}\|_F^2$ on the logarithms of our parameters. This penalty corresponds to the geodesic distance between our parameter and the identity matrix (Moakher (2005), Eq. 2.9); it has the effect of normalizing our matrix towards assuming independence, and thus is a natural penalty to impose.

4.4.3 Derivation of the method

To minimize the NLL, we will use Riemannian gradient descent. The problem is not convex, but it is geodesically convex. See Algorithm 2 for pseudocode. We chose to parameterize our problem in terms of $\mathbf{\Theta}$, as it consistently led to simpler expressions.

Without any simplification, these gradients require computing the inverse of the strong product. The computation of such a matrix is prohibitively expensive ($O(d_1^3 d_2^3)$ runtime and $O(d_1^2 d_2^2)$ space). For Kronecker-sum methods, this is often circumvented by eigendecomposing the inputs (Greenewald et al., 2019; Li et al., 2022). The eigenvectors can be extracted from the Kronecker sum, reducing the problem to the computation of the inverse of a (large) diagonal matrix.

The strong product is not amenable to the same trick. For other models, there has always been only a single parameter for each axis. In our case, the columns have two parameters, $\mathbf{\Psi}_{2,w}$ and $\mathbf{\Theta}$.

They must be *simultaneously diagonalized*. Diagonalization by orthonormal vectors is only possible for commuting matrices, which rarely holds. However, for any pair of positive definite matrices, they can be *simultaneously diagonalized via congruence*, i.e. $\mathbf{\Theta} = \mathbf{P}\mathbf{P}^T$ and $\mathbf{\Psi}_{2,w} = \mathbf{P}\mathbf{D}\mathbf{P}^T$.

$\mathbf{P}^T \neq \mathbf{P}^{-1}$, unlike in the case of eigendecomposition. Regardless, they can still be factored out of the strong product. In fact, by Lemma 2 in Dahl et al. (2013), we can even factor $\mathbf{P}$ and $\mathbf{V}$ (where $\mathbf{\Psi}_1 = \mathbf{V}\mathbf{\Lambda}\mathbf{V}^T$) out of the stridewise-blockwise trace. This yields the following gradients, in which the portion corresponding to the regularizer is shaded grey.

$$\begin{aligned}
\frac{d}{d\mathbf{\Psi}_1}\text{NLL} &= -\mathbf{V}\text{tr}_{d_2}\left[(\mathbf{\Lambda}\oplus\mathbf{D})^{-1}\right]\mathbf{V}^T \\
&\quad + \mathbf{X}\mathbf{\Theta}\mathbf{X}^T + 2\rho\mathbf{\Psi}_1^{-1}\log\mathbf{\Psi}_1 && \text{(grad1)} \\
\frac{d}{d\mathbf{\Psi}_{2,w}}\text{NLL} &= -\mathbf{P}^{-T}\text{tr}^{d_1}\left[(\mathbf{\Lambda}\oplus\mathbf{D})^{-1}\right]\mathbf{P}^{-1} \\
&\quad + \mathbf{S}_2 + 2\rho\mathbf{\Psi}_{2,w}^{-1}\log\mathbf{\Psi}_{2,w} && \text{(grad2)} \\
\frac{d}{d\mathbf{\Theta}}\text{NLL} &= -\mathbf{P}^{-T}\text{tr}^{d_1}\left[(\mathbf{\Lambda}\oplus\mathbf{D})^{-1}(\mathbf{\Lambda}\otimes\mathbf{I})\right]\mathbf{P}^{-1} \\
&\quad + \mathbf{X}^T\mathbf{\Psi}_1\mathbf{X} + 2\rho\mathbf{\Theta}^{-1}\log\mathbf{\Theta} && \text{(grad3)}
\end{aligned}$$

4.4.4 Geodesic convexity

Our optimization problem is not convex, due to the inclusion of a Kronecker product of two parameters. However, it is *geodesically convex*, or *g-convex*. In brief, geodesic convexity is the analogous concept to standard convexity when optimization takes place on a manifold (such as the set of positive-definite matrices), and many useful properties are preserved - in particular, the fact that sums of g-convex functions are g-convex and that all local minima are global.

The field of geodesic convexity is too broad to overview here, so we refer the reader to the relevant literature (Absil et al., 2009; Vishnoi, 2018). There are multiple metrics one could apply to the set of positive definite matrices (see (Han et al., 2021) for a comparison), but by far the most common is the one with geodesic distance $\gamma(\mathbf{\Sigma}_1, \mathbf{\Sigma}_2) = \mathbf{\Sigma}_1^{\frac{1}{2}}\left(\mathbf{\Sigma}_1^{\frac{-1}{2}}\mathbf{\Sigma}_2\mathbf{\Sigma}_1^{\frac{-1}{2}}\right)^t\mathbf{\Sigma}_1^{\frac{1}{2}}$. We'll denote it $\mathbb{S}_{++}$, and denote the geometry of tuples of positive definite matrices (each equipped with the same geodesic along the relevant coordinates) as $\mathbb{S}_{++}^{\otimes}$. These geometries have several key properties:

1. Traces $\text{tr}\left[\mathbf{\Sigma}\mathbf{X}\right]$ are g-convex ($\mathbb{S}_{++}$)
2. Log determinants $-\log|\mathbf{\Sigma}|$ are g-convex ($\mathbb{S}_{++}$)
3. Orthogonal invariance and convexity in $\log\mathbf{\Sigma}$ implies g-convexity ($\mathbb{S}_{++}$) (Yi & Tyler, 2020)
4. $\|\log\mathbf{\Sigma}\|_F^2$ is *strictly* g-convex ($\mathbb{S}_{++}$) (Yi & Tyler, 2020)
5. Log determinants of sums $-\log\left|\sum_i \mathbf{\Sigma}_i\right|$ are g-convex ($\mathbb{S}_{++}^{\otimes}$) (Wiesel, 2012a)
6. If $f(\mathbf{\Omega})$ is g-convex ($\mathbb{S}_{++}$), then $g(\mathbf{\Sigma}_1, \mathbf{\Sigma}_2) = f(\mathbf{\Sigma}_1 \otimes \mathbf{\Sigma}_2)$ is g-convex ($\mathbb{S}_{++}^{\otimes}$) (Wiesel, 2012b)

Let us parameterize our problem as $\mathbf{\Psi}_1 \otimes \mathbf{\Theta} + \mathbf{\Gamma} \otimes \mathbf{\Psi}_{2,b}$, where $\mathbf{\Gamma}$ is constrained to be the identity matrix. (such a constraint is g-convex). Then, the negative log-likelihood is a sum of traces (g-convex), terms of the form $\|\log\mathbf{\Sigma}\|_F^2$ (strictly g-convex), and a log determinant of a sum of Kronecker products (g-convex). Thus, our problem is g-convex.

In fact, the problem is *strongly* g-convex, as $\|\log\mathbf{\Sigma}\|_F^2$ is strongly g-convex (See Equation 18 of Darmwal and Rajawat (2023)). Accordingly, we can expect fast (linear) convergence from gradient descent. This can be seen in Figure 4.1.

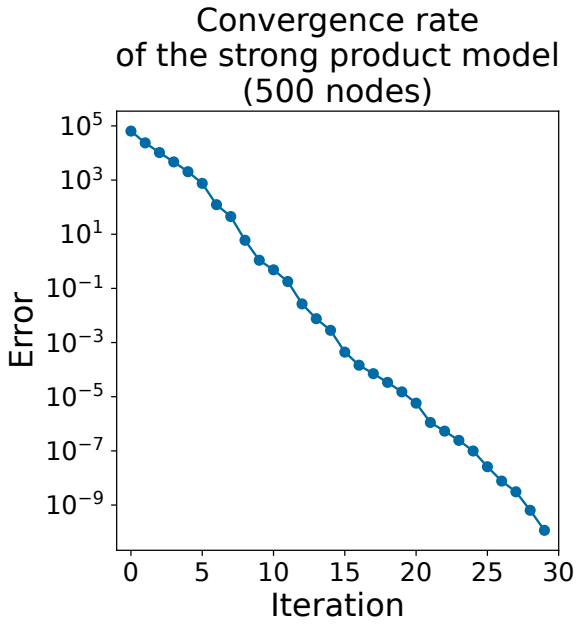


Figure 4.1: Rate of decrease in error using Algorithm 2. Ran on synthetic data, see Section 4.5.1 for generation details.

4.4.5 Riemannian gradient descent

Rather than using traditional gradient descent, the geodesic convexity motivates the use of *Riemannian* gradient descent. This is analogous to traditional gradient descent, but follows the direction of steepest descent under the definition of 'steepest' intrinsic to the manifold (in this case, the set of positive semi-definite matrices). We refer the reader to several papers which we found beneficial for understanding how to utilize the method in practice, in particular the following: Hosseini and Sra (2015), Duembgen and Tyler (2016), Jeuris et al. (2012), and Darmwal and Rajawat (2023).

Here, we will only state the changes that need to be made to gradient descent to fit the Riemannian context. Rather than using the 'Euclidean' gradient of f at a point $\mathbf{X}$, $\operatorname{grad}^{\mathbb{E}}_{\mathbf{X}} f = \frac{d}{d\mathbf{X}} f(\mathbf{Y})\Big|_{\mathbf{Y}=\mathbf{X}}$, the correct gradient is $\operatorname{grad}^{\mathbb{S}^{++}}_{\mathbf{X}} f = \mathbf{X}\left(\operatorname{grad}^{\mathbb{E}}_{\mathbf{X}} f\right)\mathbf{X}$. The second required concept is the 'retraction' at $\mathbf{X}$, $\mathrm{R}^{\mathbb{S}^{++}}_{\mathbf{X}}(\boldsymbol{\xi}) = \mathbf{X}e^{\mathbf{X}^{-1}\boldsymbol{\xi}}$. It maps steps in a direction (typically the gradient) to a position on the relevant geodesic. See Sra and Hosseini (2015) for details.

One step of Riemannian gradient descent with stepsize η can be expressed as $\mathbf{X}_{t+1} = \mathrm{R}^{\mathbb{S}^{++}}_{\mathbf{X}_t}\left(-\eta \operatorname{grad}^{\mathbb{S}^{++}}_{\mathbf{X}_t} f\right)$. To determine an effective step size, we use backtracking line search with Armijo's rule (Armijo, 1966). Line search is more expensive in the Riemannian case, due to the need to repeatedly compute the retraction, which involves matrix exponentiation; modifications can be made to mitigate this (Yamakawa et al., 2023), but for pedagogical reasons we shall stick with the most basic form.

4.4.6 Identifiability

Kronecker models often are not identifiable. For our *penalized* problem, this is not an issue; we will see in Section 4.4.7 that the minimum is unique. However, this is due to the strict g-convexity of our penalty function; it is still of interest to understand the extent to which our original problem is identifiable. Below, we list the non-identifiabilities of common Kronecker decompositions, where c is any constant.

$$\boldsymbol{\Psi}_1 \otimes \boldsymbol{\Psi}_2 = \frac{1}{c}\boldsymbol{\Psi}_1 \otimes c\boldsymbol{\Psi}_2 \qquad \text{(Kronecker Product)}$$

$$\boldsymbol{\Psi}_1 \oplus \boldsymbol{\Psi}_2 = (\boldsymbol{\Psi}_1 + c\mathbf{I}) \oplus (\boldsymbol{\Psi}_2 - c\mathbf{I}) \qquad \text{(Kronecker Sum)}$$

Algorithm 2 Strong Product Model MLE

Input: Initial guesses $\mathbf{\Psi}_1^{(0)}, \mathbf{\Theta}^{(0)}, \mathbf{\Psi}_{2,w}^{(0)}$, data $\mathbf{X}$
Input: Regularization parameters $\rho_{\psi_1}, \rho_\theta, \rho_{\psi_{2,w}}$
Input: Line search parameters $\eta_{\text{init}}, \beta, \tau$
  while not converged
    #Riemannian Gradient Descent
    $\mathcal{P} \leftarrow \left\{\mathbf{\Psi}_1, \mathbf{\Theta}, \mathbf{\Psi}_{2,w}\right\}$
    for parameter $\mathbf{\Gamma} \in \mathcal{P}$
      $\mathbf{G} \leftarrow$ gradient at $\mathbf{\Gamma}^{(t)}$ (grad1)/(grad2)/(grad3)
      $\overline{\mathbf{G}}_{\mathbf{\Gamma}} \leftarrow \mathbf{\Gamma}^{(t)}\mathbf{G}\mathbf{\Gamma}^{(t)}$
      $\mathbf{\Gamma}^{(t+1)} \leftarrow \mathbf{\Gamma}^{(t)} e^{-\eta_t \mathbf{\Gamma}_{(t)}^{-1}\overline{\mathbf{G}}_{\mathbf{\Gamma}}}$
    end for
    #Riemannian Armijo Line Search
    $\eta_t \leftarrow \eta_{\text{init}}$
    while $f\left(\mathcal{P}^{(t+1)}\right) > f\left(\mathcal{P}^{(t)}\right) - \sum_{\mathbf{\Gamma}} \eta_t \beta \|\overline{\mathbf{G}}_{\mathbf{\Gamma}}\|_F^2$
      $\eta_t \leftarrow \tau\eta_t$
      $\mathbf{\Gamma}^{(t+1)} \leftarrow \mathbf{\Gamma}^{(t)} e^{-\eta_t \mathbf{\Gamma}_{(t)}^{-1}\overline{\mathbf{G}}}$
    end while
  end while

$$(\mathbf{\Psi}_1 \oplus \mathbf{\Psi}_2)^2 = \left((\mathbf{\Psi}_1 + c\mathbf{I}) \oplus (\mathbf{\Psi}_2 - c\mathbf{I})\right)^2 \qquad \text{(Sylvester Sum)}$$

It is not immediately obvious what the non-identifiabilities of our model are. We summarize them with the following proposition:

Proposition 1. *Let* $(\mathbf{\Psi}_1, \mathbf{\Psi}_{2,w}, \mathbf{\Theta})$ *and* $(\overline{\mathbf{\Psi}}_1, \overline{\mathbf{\Psi}}_{2,w}, \overline{\mathbf{\Theta}})$ *be two different parameterizations. Then, they have the same strong product if and only if one of the following conditions hold for some* c_1, c_2*:*

1. $\mathbf{\Psi}_1 = c_1\mathbf{I}$, $\overline{\mathbf{\Psi}}_1 = c_2\mathbf{I}$, $c_1\mathbf{\Theta} + \mathbf{\Psi}_{2,w} = c_2\overline{\mathbf{\Theta}} + \overline{\mathbf{\Psi}}_{2,w}$
2. $\mathbf{\Psi}_1 = c_1\mathbf{I} + c_2\overline{\mathbf{\Psi}}_1$, $\overline{\mathbf{\Theta}} = c_2\mathbf{\Theta}$, $\overline{\mathbf{\Psi}}_{2,w} = \mathbf{\Psi}_{2,w} + c_1\mathbf{\Theta}$

Case 1 tells us that we cannot identify differences between within-row and between-row column dependencies when the rows are independent (when $\mathbf{\Psi}_1 \propto \mathbf{I}$). This is to be expected; it doesn't make sense to talk of between-row dependencies when the rows are independent. Additionally, when the rows are independent, algorithms that assume independence, such as *glasso*, should be preferred.

On the surface, case 2 seems much more complicated; it is capturing the mixing of Kronecker product and Kronecker sum non-identifiabilities. Consider that, when $c_1 = 0$, this reduces to Kronecker product non-identifiability. When $c_2 = 1$, it's not quite the Kronecker sum non-identifiability but is analogous to it.

Case 2 also has an intuitive interpretation: as the variance of our rows decreases (c_1 is negative), our within-row graph looks more like our between-row graph (since approximately $\overline{\mathbf{\Psi}}_{2,w} \propto \mathbf{\Theta}$ for large negative c_1). This makes sense intuitively: if there were no row-variance (i.e. the columns are all identical), then there is no difference between between-row and within-row relationships; the value in a cell in a given column is the same regardless of which row it is in.

Proof.

$$\begin{aligned}
&\mathbf{\Psi}_1 \otimes \mathbf{\Theta} + \mathbf{I} \otimes \mathbf{\Psi}_{2,w} = \overline{\mathbf{\Psi}}_1 \otimes \overline{\mathbf{\Theta}} + \mathbf{I} \otimes \overline{\mathbf{\Psi}}_{2,w} \\
&\iff \mathbf{\Psi}_1 \otimes \mathbf{\Theta} = \overline{\mathbf{\Psi}}_1 \otimes \overline{\mathbf{\Theta}} + \mathbf{I} \otimes \left(\overline{\mathbf{\Psi}}_{2,w} - \mathbf{\Psi}_{2,w}\right)
\end{aligned}$$

Note that the left hand side of the equation is a rank-1 Kronecker decomposition and the right hand side is a rank-2 Kronecker decomposition (i.e. it is a sum of two Kronecker products). This can only hold if either the left terms $\{\mathbf{\Psi}_1, \overline{\mathbf{\Psi}}_1, \mathbf{I}\}$, or alternatively if the right terms $\{\mathbf{\Theta}, \overline{\mathbf{\Theta}}, \left(\overline{\mathbf{\Psi}}_{2,w} - \mathbf{\Psi}_{2,w}\right)\}$, form a rank-1 linearly dependent set. Let's assume the first set is linearly dependent. Then, they must all be multiples of the identity matrix:

$$c_1\mathbf{I} \otimes \mathbf{\Theta} = c_2\mathbf{I} \otimes \overline{\mathbf{\Theta}} + \mathbf{I} \otimes \left(\overline{\mathbf{\Psi}}_{2,w} - \mathbf{\Psi}_{2,w}\right)$$
$$\iff c_1\mathbf{\Theta} + \mathbf{\Psi}_{2,w} = c_2\overline{\mathbf{\Theta}} + \overline{\mathbf{\Psi}}_{2,w}$$

This yields case 1. Now let's consider what happens when the right terms are linearly dependent, i.e. if $\overline{\mathbf{\Psi}}_{2,w} - \mathbf{\Psi}_{2,w} = c_1\mathbf{\Theta}$, $\overline{\mathbf{\Theta}} = c_2\mathbf{\Theta}$. Then we must have:

$$\mathbf{\Psi}_1 = c_2\overline{\mathbf{\Psi}}_1 + c_1\mathbf{I}$$

□

These non-identifiabilities could be fixed with a trace constraint $\operatorname{tr}[\mathbf{\Gamma}] = \tau_{\mathbf{\Gamma}}$ for some chosen constant $\tau_{\mathbf{\Gamma}}$ for every factor matrix $\mathbf{\Gamma}$. Furthermore, the *support* of $\mathbf{\Psi}_1$ and $\mathbf{\Theta}$ is preserved; they always represents the same graph. Since we are penalizing our model by a strictly g-convex function, **it is not affected by these non-identifiabilities**.

4.4.7 Existence and uniqueness

Theorem 5. *There exists a unique solution to the penalized optimization problem given by Equation 4.1 equipped with the penalty $\rho\|\mathbf{\Gamma}\|_F^2$ in each of its parameters.*

Proof. Uniqueness follows from the strict g-convexity of our objective. Existence follows from the fact that our function is *g-coercive*. G-coercitivity guarantees for g-convex functions that there exists a compact set of minimizers (Duembgen & Tyler, 2016; Sra & Hosseini, 2015), and in fact for any g-convex function $f(\mathbf{\Sigma})$, $f(\mathbf{\Sigma}) + \|\log \mathbf{\Sigma}\|_F^2$ is g-coercive (see Lemma 4.7 of Duembgen and Tyler (2016)). Existence and uniqueness holds both coordinate-wise and jointly. □

4.5 Experimental results

We will now validate our methodology on synthetic and real-world datasets. In Section 4.5.1, we will demonstrate the performance of our algorithm on within-distribution synthetic data. Next, in Section 4.5.2, we will validate our model on a real-world dataset for which the results are easily interpretable without specialized domain knowledge. Finally, in Section 4.5.3, we will validate our model on a real-world scRNA-seq dataset. For real-world datasets $\mathbf{D}$, we used as input to the models $\mathbf{D} - \frac{1}{d_1 d_2}\mathbf{1}^T\mathbf{D}\mathbf{1}$, to lessen the impact of the zero-mean assumption. Code is available on GitHub (https://github.com/BaileyAndrew/Strong-Product-Model).

All experiments were run on a 2020 M1 Macbook Pro with 8GB of RAM. We implemented our algorithm using Python 3.9.18, NumPy 1.25.2, (Harris et al., 2020) and SciPy 1.12.0 (Virtanen et al., 2020). Our line search parameters were $\eta_{\text{init}} = 0.1$, $\beta = \frac{1}{2}$, and $\tau = \frac{1}{2}$, except in cases where we used a large regularization parameter, in which case we found we needed $\beta = 0$ to prevent early convergence (it is likely that a smaller, nonzero value would suffice). We always initialized our parameters to the identity matrix. $\rho = 0.1$, except for the COIL-20 experiment (Section 4.5.2), in which we found a slight improvement using $\rho = 10^6$.

For precision-recall (PR) curves, error bars indicate max/min performance over 10 trials. For other curves, they indicate standard deviation. As we are the first to introduce the strong product model, there is no direct prior work to compare against. We compared against GmGM, as it performs similarly to other multi-axis models (Andrew et al., 2024) and is by far the fastest, allowing us to run many high-dimensional trials. It is a Kronecker-sum structured model. PR curves were generated by varying the amount of thresholding applied to the outputs.

Since our model reduces (up to a constant shift in the diagonals) to both the Kronecker product and Kronecker sum models when $\mathbf{\Psi}_{2,b} = \mathbf{I}$ and $\mathbf{\Psi}_{2,w} = \mathbf{I}$, respectively, we also used these restricted models for comparison.

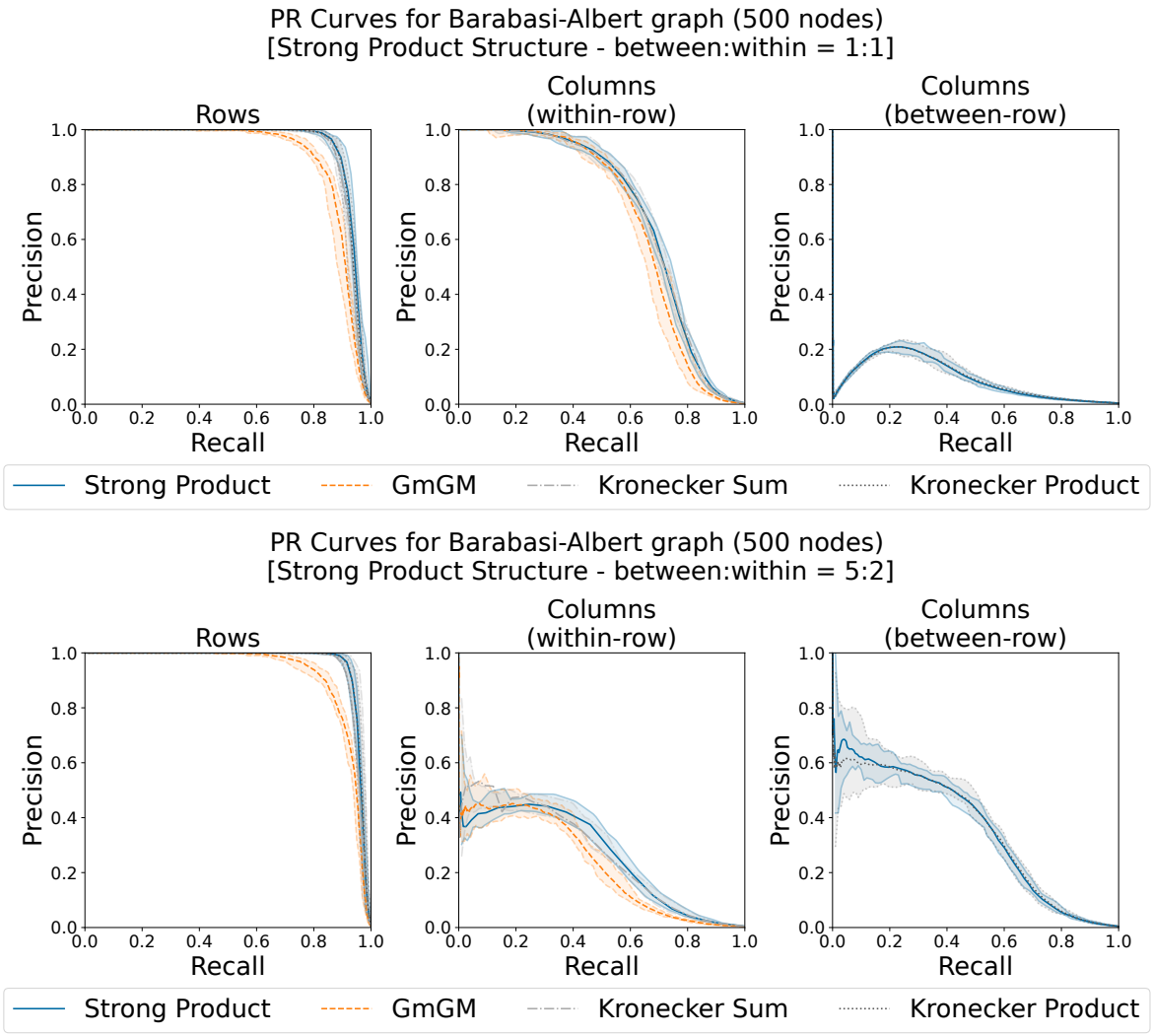


Figure 4.2: Precision-recall curves on random graphs with structure $\mathbf{A} \otimes x\mathbf{B} + \mathbf{A} \oplus \frac{1}{x}\mathbf{C}$. 'Strong Product' refers to our model; 'Kronecker Sum' and 'Kronecker Product' refer to our model with the relevant factor ($\mathbf{B}$ or $\mathbf{C}$) constrained to an identity matrix. GmGM and 'Kronecker Sum' do not estimate the between-row graph, and 'Kronecker Product' does not estimate the within-row graph; they have been ommitted from the relevant sub-plots. (Top) $x = 1$. (Bottom) $x = 2.5$

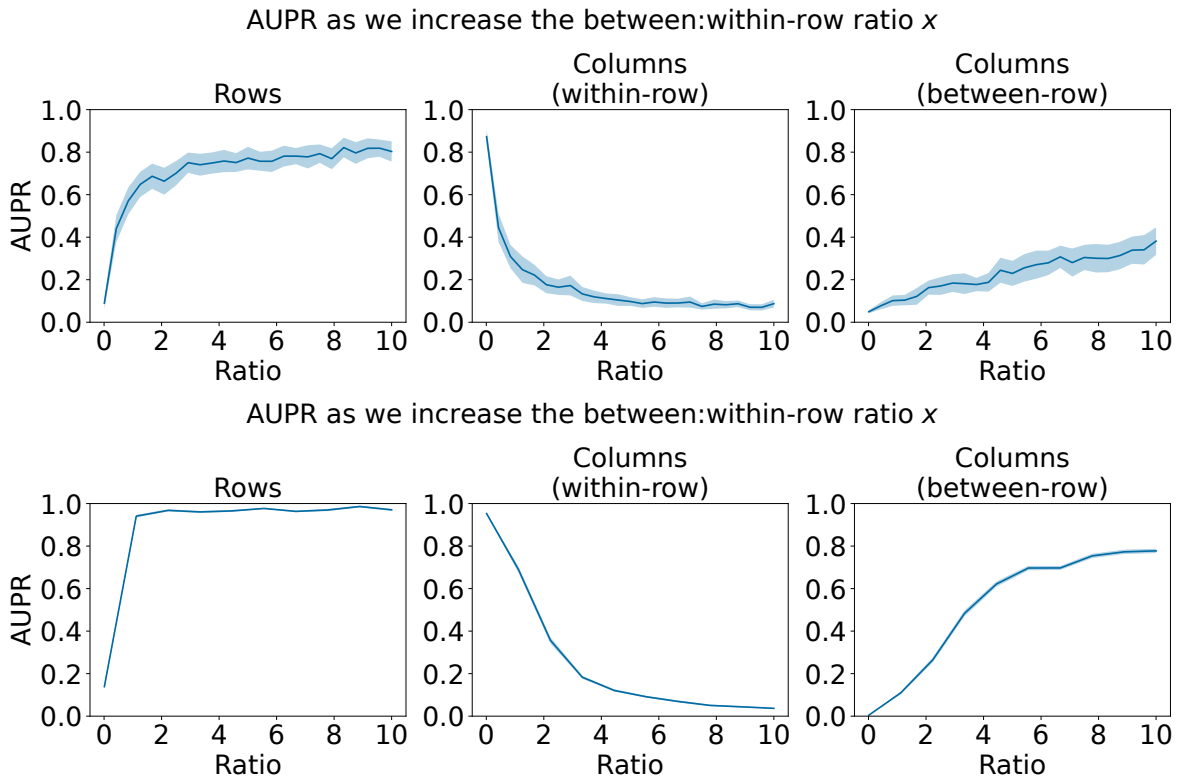


Figure 4.3: How AUPR varies as the ratio x varies, for data generated following the structure outlined in Section 4.5.1. (Top) 100-node graphs, 50 trials. (Bottom) 500-node graphs, 25 trials. Note that the deviations are much smaller for larger graphs.

4.5.1 Synthetic data

For our synthetic data, we generated our ground truth graphs to have the form $y\mathbf{A} \otimes x\mathbf{B} + y\mathbf{A} \oplus \frac{1}{x}\mathbf{C}$, where $\mathbf{A}, \mathbf{B}, \mathbf{C}$ are Barabasi-Albert random graphs with 500 nodes each. Barabasi-Albert is a power-law distribution; many real world datasets (approximately) follow power-law distributions, and hence we felt it was an adequate test case.

Results reported in the Figure 4.2 assume $y = 1$. We found that the ratio of strength of between-row and within-row column factors, x, had a large effect on performance. In Figure 4.3, we show how the area under PR curves (AUPR) changes as x varies.

In Figure 4.2, we report the PR curves on our performance on synthetic data, where $x = 1$ and $x = 2.5$ to both validate our model and demonstrate this effect. Note that we do not demonstrate any loss in performance compared to other models, and are the only model that estimates all three relevant graphs. Other models only estimate one type of column graph, and assume that the other does not exist.

We ran several other tests, but defer these to the supplementary material due to space concerns. The supplementary material contains a demonstration that performance improves as the number of nodes increases, an analysis of the effects of changing y, and a demonstration that our algorithm still performs well when run on data that does not follow a strong product structure. We also show that the regularization parameter has minimal effect on the results; its purpose is primarily to ensure existence of solutions.

The results presented here and in the supplementary material show that our model performs well on all graphical models considered by prior work, and is the only option that learns both types of column graph. There is nothing to be gained from assuming a more restricted structure, such as the Kronecker sum or Kronecker product. Of course, our results also show that models developed for such structures also perform well when applied to data generated from our proposed structure, except that they are unable to estimate either the between-row column graph (Kronecker sum) or the within-row column graph (Kronecker product).

4.5.2 COIL dataset

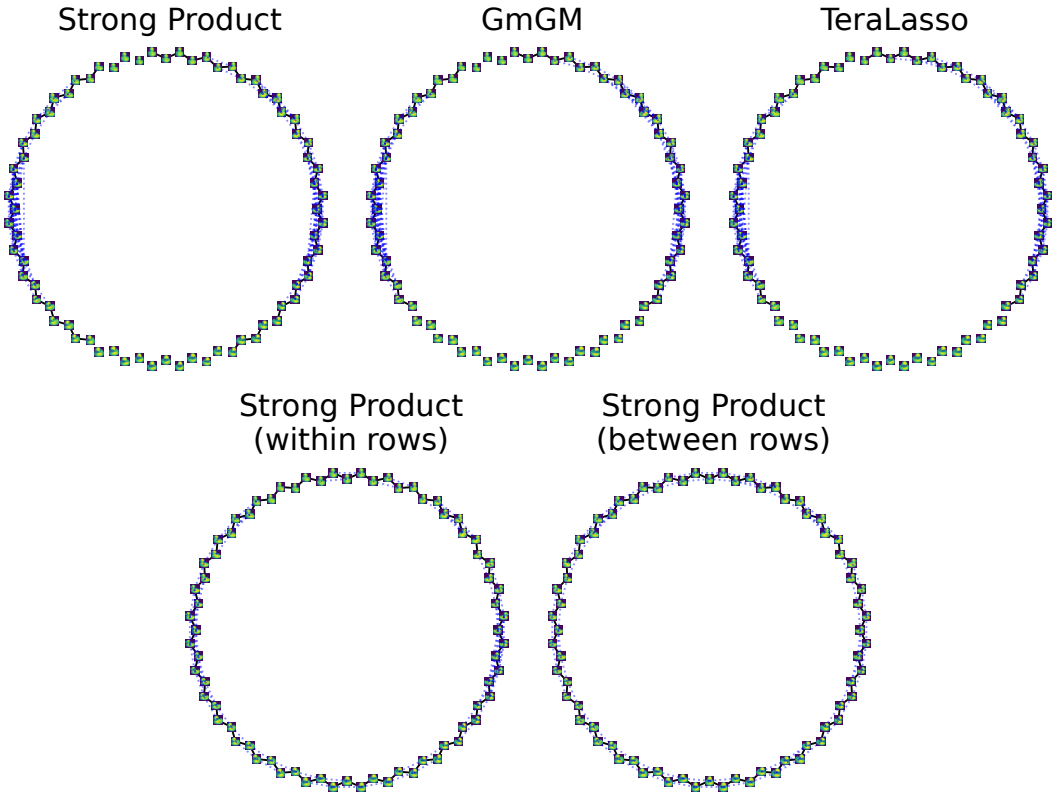


Figure 4.4: The learned connections between the video frames for various algorithms. Black connections indicate connections between adjacent frames, blue indicate non-adjacent frames. A larger version of this figure is available in the supplementary material. For the bottom row, we transposed the data and investigated the two different returned graphs.

The first real-world dataset we test on is the COIL-20 video dataset (Nene et al., 1996). This is somewhat of a 'classic' dataset for Kronecker-structured models, with several prior algorithms also demonstrating results on it. Its convenience comes from the fact that there is a very clear 'natural graph' we should expect in the data: a frame in the video should be connected only to the frames immediately before and immediately after itself. The video is that of a rotating duck, and hence we would expect the first and last frames to connect as well.

The video is a $72 \times 128 \times 128$ tensor; we down-sampled the pixels to $72 \times 16 \times 16$, and then flattened to 72×256. We compared our performance against GmGM and TeraLasso (Greenewald et al., 2019). TeraLasso is a Kronecker-sum-structure algorithm that performs similarly to GmGM. Its main differences are that it does not enforce positive-definite factor graphs and uses *glasso* regularization. Due to its similarity and comparatively slow runtime, we did not compare against it for our other, more computationally expensive tests.

We chose thresholding/regularization parameters so that all models output roughly 144 edges (exactly 144 was not always possible). Figure 4.4 displays a graphical interpretation of our results - an enlarged version of this figure is available in the supplementary material, and Table 4.2 contains the corresponding numeric metrics. We found that our model performed similarly to prior work. However, when we considered the frames to be the column dimension, the performance was much better than prior work.

4.5.3 scRNA-seq dataset

While informative, the COIL dataset is somewhat of a toy example. To get a better understanding of how our algorithm will perform on real data, we will use a mouse embryo stem cell scRNA-seq dataset

Algorithm	Precision	1-off Precision	Recall
Strong Product	38%	65%	78%
GmGM	33%	59%	67%
TeraLasso	36%	64%	69%
Within-Row	49%	85%	**100%**
Between-Row	**49%**	**94%**	**100%**

Table 4.2: Metrics for the COIL dataset experiment. 1-off precision refers to frames connected to adjacent frames and frames with one intermediate frame between them. As we chose to keep twice as many edges as exist, the highest possible value for precision is 50%.

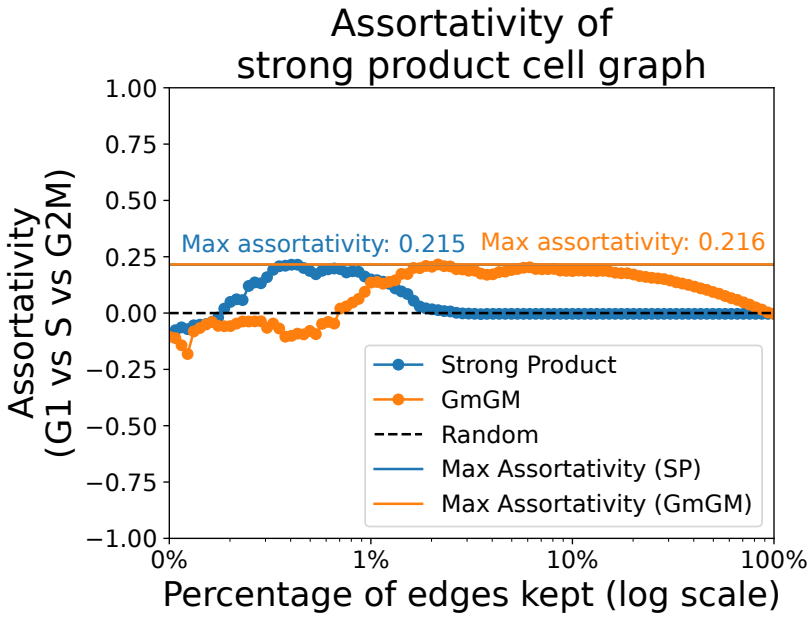


Figure 4.5: Assortativity as we vary thresholding, for the scRNA-seq dataset.

from Buettner et al. (2015), limited to only mitosis-related genes as in Li et al. (2022). The dataset consists of 288 cells and 167 genes, in which each cell has been labelled according to its stage in the cell cycle - either G1, S, or G2M.

To understand performance, we can use *assortativity* - this measures the tendency of nodes in the same class (cell cycle stage) to connect to each other in a graph, and ranges from -1 (tend not to connect) to 0 (no tendency) to 1 (tend to connect). We should expect there to be some positive assortativity, as there should be similarities between the mitotic gene expressions of cells in the same stage.

Interestingly, we found that our algorithm found assortativities very similar to prior work (GmGM), but on much sparser graphs. GmGM only had high assortativity when we only slightly thresholded the output graph; for sparse graphs, the assortativity was 0. In contrast, our model had high assortativity on strongly thesholded graphs, but had an assortativity of zero otherwise. This can be seen in Figure 4.5. The sparser regime is the more relevant regime here, as most real-world interaction networks tend to be sparse.

4.6 Conclusion

We have shown that our model is able to perform well on synthetic data generated from both the strong product model, the Kronecker sum model, and the Kronecker product model. Our model, like other Kronecker-structured models, improves in performance as the number of rows and columns increases, even if we only have one sample matrix. On real data, we still get reasonable results, and outperform prior work in some cases.

We are very interested in the performance of our model as the between-row/within-row column ratio changes, and the extent to which this ratio is identifiable in real data. In future work, we intend to prove statistical recovery rates, parameterized by this ratio. Additionally, graphically-focused Kronecker-structured models do not seem to have adopted the geodesic convexity framework for optimization. The fast convergence we experience is encouraging; we would like to incorporate sparsity-inducing regularizers into this framework.

Acknowledgements

There was no direct funding for this research. B.A. is supported by the UKRI Engineering and Physical Sciences Research Council (EPSRC) [EP/S024336/1]. We would like to thank David Marples for their ideas and insightful comments. Additionally, we would like to thank the reviewers for their active input and discussion; it was valuable and very much appreciated.

Bibliography for Paper 3

Absil, P.-A., Mahony, R., & Sepulchre, R. (2009, April). *Optimization Algorithms on Matrix Manifolds*. Princeton University Press. https://doi.org/10.1515/9781400830244

Ambroise, C., Chiquet, J., & Matias, C. (2009). Inferring sparse Gaussian graphical models with latent structure [Publisher: Institute of Mathematical Statistics and Bernoulli Society]. *Electronic Journal of Statistics*, *3*(none), 205–238. https://doi.org/10.1214/08-EJS314

Andrew, B., Westhead, D. R., & Cutillo, L. (2024, January). GmGM: A fast multi-axis Gaussian graphical model [Conference Name: The 27th International Conference on Artificial Intelligence and Statistics ISSN: 2640-3498 Meeting Name: The 27th International Conference on Artificial Intelligence and Statistics Place: València, Spain Publisher: Proceedings of Machine Learning Research]. Retrieved July 16, 2024, from https://eprints.whiterose.ac.uk/208283/

Armijo, L. (1966). Minimization of functions having Lipschitz continuous first partial derivatives. [Publisher: Pacific Journal of Mathematics, A Non-profit Corporation]. *Pacific Journal of Mathematics*, *16*(1), 1–3. Retrieved August 16, 2024, from https://projecteuclid.org/journals/pacific-journal-of-mathematics/volume-16/issue-1/Minimization-of-functions-having-Lipschitz-continuous-first-partial-derivatives/pjm/1102995080.full

Buettner, F., Natarajan, K. N., Casale, F. P., Proserpio, V., Scialdone, A., Theis, F. J., Teichmann, S. A., Marioni, J. C., & Stegle, O. (2015). Computational analysis of cell-to-cell heterogeneity in single-cell RNA-sequencing data reveals hidden subpopulations of cells [Number: 2 Publisher: Nature Publishing Group]. *Nature Biotechnology*, *33*(2), 155–160. https://doi.org/10.1038/nbt.3102

Dahl, A., Hore, V., Iotchkova, V., & Marchini, J. (2013, December). Network inference in matrix-variate Gaussian models with non-independent noise [arXiv:1312.1622 [stat]]. https://doi.org/10.48550/arXiv.1312.1622

Darmwal, Y., & Rajawat, K. (2023, December). Low-complexity subspace-descent over symmetric positive definite manifold [arXiv:2305.02041 [cs, eess, math, stat]]. https://doi.org/10.48550/arXiv.2305.02041

Deng, X., & Tsui, K.-W. (2013). Penalized Covariance Matrix Estimation Using a Matrix-Logarithm Transformation [Publisher: Taylor & Francis _eprint: https://doi.org/10.1080/10618600.2012.715556]. *Journal of Computational and Graphical Statistics*, *22*(2), 494–512. https://doi.org/10.1080/10618600.2012.715556

Duembgen, L., & Tyler, D. E. (2016, July). Geodesic Convexity and Regularized Scatter Estimators [arXiv:1607.05455 [stat]]. https://doi.org/10.48550/arXiv.1607.05455

Fan, J., & Li, R. (2001). Variable Selection via Nonconcave Penalized Likelihood and its Oracle Properties [Publisher: Taylor & Francis _eprint: https://doi.org/10.1198/016214501753382273]. *Journal of the American Statistical Association*, *96*(456), 1348–1360. https://doi.org/10.1198/016214501753382273

Friedman, J., Hastie, T., & Tibshirani, R. (2008). Sparse inverse covariance estimation with the graphical lasso. *Biostatistics*, *9*(3), 432–441. https://doi.org/10.1093/biostatistics/kxm045

Greenewald, K., Zhou, S., & Hero, A. (2019). Tensor graphical lasso (TeraLasso) [Publisher: [Royal Statistical Society, Oxford University Press]]. *Journal of the Royal Statistical Society. Series B (Statistical Methodology)*, *81*(5), 901–931. Retrieved July 16, 2024, from https://www.jstor.org/stable/26833080

Guillot, D., & Rajaratnam, B. (2015). Functions Preserving Positive Definiteness for Sparse Matrices [Publisher: American Mathematical Society]. *Transactions of the American Mathematical Society*, *367*(1), 627–649. Retrieved August 12, 2024, from https://www.jstor.org/stable/24512825

Han, A., Mishra, B., Jawanpuria, P. K., & Gao, J. (2021). On Riemannian Optimization over Positive Definite Matrices with the Bures-Wasserstein Geometry. *Advances in Neural Information Processing Systems*, *34*, 8940–8953. Retrieved August 14, 2024, from https://proceedings.neurips.cc/paper_files/paper/2021/hash/4b04b0dcd2ade339a3d7ce13252a29d4-Abstract.html

Harris, C. R., Millman, K. J., van der Walt, S. J., Gommers, R., Virtanen, P., Cournapeau, D., Wieser, E., Taylor, J., Berg, S., Smith, N. J., Kern, R., Picus, M., Hoyer, S., van Kerkwijk, M. H., Brett, M., Haldane, A., del Río, J. F., Wiebe, M., Peterson, P., ... Oliphant, T. E. (2020). Array programming with NumPy [Number: 7825 Publisher: Nature Publishing Group]. *Nature*, *585*(7825), 357–362. https://doi.org/10.1038/s41586-020-2649-2

Hosseini, R., & Sra, S. (2015). Matrix Manifold Optimization for Gaussian Mixtures. *Advances in Neural Information Processing Systems*, *28*. Retrieved August 15, 2024, from https://papers.nips.cc/paper_files/paper/2015/hash/dbe272bab69f8e13f14b405e038deb64-Abstract.html

Jeuris, B., Vandebril, R., & Vandereycken, B. (2012). A survey and comparison of contemporary algorithms for computing the matrix geometric mean. *Electronic Transactions on Numerical Analysis*, *39*, 379–402. Retrieved August 15, 2024, from https://etna.ricam.oeaw.ac.at/volumes/2011-2020/vol39/abstract.php?vol=39&pages=379-402

Kalaitzis, A., Lafferty, J., Lawrence, N. D., & Zhou, S. (2013). The Bigraphical Lasso [ISSN: 1938-7228]. *Proceedings of the 30th International Conference on Machine Learning*, 1229–1237. Retrieved August 8, 2024, from https://proceedings.mlr.press/v28/kalaitzis13.html

Li, S., López-García, M., Lawrence, N. D., & Cutillo, L. (2022, March). Scalable Bigraphical Lasso: Two-way Sparse Network Inference for Count Data [arXiv:2203.07912 [cs, stat]]. https://doi.org/10.48550/arXiv.2203.07912

Liu, H., Han, F., Yuan, M., Lafferty, J., & Wasserman, L. (2012). The nonparanormal SKEPTIC. *Proceedings of the 29th International Conference on Machine Learning, ICML 2012*, 1415–1422. Retrieved July 16, 2024, from https://collaborate.princeton.edu/en/publications/the-nonparanormal-skeptic

Meinshausen, N., & Bühlmann, P. (2006). High-Dimensional Graphs and Variable Selection with the Lasso [Publisher: Institute of Mathematical Statistics]. *The Annals of Statistics*, *34*(3), 1436–1462. Retrieved February 9, 2025, from https://www.jstor.org/stable/25463463

Moakher, M. (2005). A Differential Geometric Approach to the Geometric Mean of Symmetric Positive-Definite Matrices [Publisher: Society for Industrial and Applied Mathematics]. *SIAM Journal on Matrix Analysis and Applications*, *26*(3), 735–747. https://doi.org/10.1137/S0895479803436937

Nene, S. A., Nayar, S. K., & Murase, H. (1996). Columbia Object Image Library (COIL-20).

Song, D., & Hero, A. O. (2023, October). On Separability of Covariance in Multiway Data Analysis [arXiv:2302.02415 [math, stat]]. https://doi.org/10.48550/arXiv.2302.02415

Sra, S., & Hosseini, R. (2015). Conic Geometric Optimization on the Manifold of Positive Definite Matrices [Publisher: Society for Industrial and Applied Mathematics]. *SIAM Journal on Optimization*, *25*(1), 713–739. https://doi.org/10.1137/140978168

Tsiligkaridis, T., Hero, A. O., & Zhou, S. (2012). Kronecker graphical lasso [ISSN: 2373-0803]. *2012 IEEE Statistical Signal Processing Workshop (SSP)*, 884–887. https://doi.org/10.1109/SSP.2012.6319849

Virtanen, P., Gommers, R., Oliphant, T. E., Haberland, M., Reddy, T., Cournapeau, D., Burovski, E., Peterson, P., Weckesser, W., Bright, J., van der Walt, S. J., Brett, M., Wilson, J., Millman, K. J., Mayorov, N., Nelson, A. R. J., Jones, E., Kern, R., Larson, E., ... van Mulbregt, P. (2020). SciPy 1.0: Fundamental algorithms for scientific computing in Python [Number: 3 Publisher: Nature Publishing Group]. *Nature Methods*, *17*(3), 261–272. https://doi.org/10.1038/s41592-019-0686-2

Vishnoi, N. K. (2018, June). Geodesic Convex Optimization: Differentiation on Manifolds, Geodesics, and Convexity [arXiv:1806.06373 [cs, math]]. https://doi.org/10.48550/arXiv.1806.06373

Wang, Y., Jang, B., & Hero, A. (2020). The Sylvester Graphical Lasso (SyGlasso) [ISSN: 2640-3498]. *Proceedings of the Twenty Third International Conference on Artificial Intelligence and Statistics*, 1943–1953. Retrieved August 8, 2024, from https://proceedings.mlr.press/v108/wang20d.html

Wang, Y., Sun, Z., Song, D., & Hero, A. (2022). Kronecker-structured covariance models for multiway data [Publisher: Amer. Statist. Assoc., the Bernoulli Soc., the Inst. Math. Statist., and the Statist. Soc. Canada]. *Statistics Surveys*, *16*(none), 238–270. https://doi.org/10.1214/22-SS139

Wiesel, A. (2012a). On the convexity in Kronecker structured covariance estimation [ISSN: 2373-0803]. *2012 IEEE Statistical Signal Processing Workshop (SSP)*, 880–883. https://doi.org/10.1109/SSP.2012.6319848

Wiesel, A. (2012b). Geodesic Convexity and Covariance Estimation [Conference Name: IEEE Transactions on Signal Processing]. *IEEE Transactions on Signal Processing*, *60*(12), 6182–6189. https://doi.org/10.1109/TSP.2012.2218241

Xue, L., Ma, S., & Zou, H. (2012). Positive-Definite L1-Penalized Estimation of Large Covariance Matrices [Publisher: Taylor & Francis _eprint: https://doi.org/10.1080/01621459.2012.725386]. *Journal of the American Statistical Association*, *107*(500), 1480–1491. https://doi.org/10.1080/01621459.2012.725386

Yamakawa, Y., Sato, H., & Aihara, K. (2023, April). Modified Armijo line-search in Riemannian optimization with reduced computational cost [arXiv:2304.02197 [math]]. https://doi.org/10.48550/arXiv.2304.02197

Yi, M., & Tyler, D. E. (2020). Shrinking the Covariance Matrix Using Convex Penalties on the Matrix-Log Transformation [Publisher: Taylor & Francis _eprint: https://doi.org/10.1080/10618600.2020.1814788]. *Journal of Computational and Graphical Statistics*, *30*(2), 442–451. https://doi.org/10.1080/10618600.2020.1814788

Zhang, C.-H. (2010). Nearly unbiased variable selection under minimax concave penalty [Publisher: Institute of Mathematical Statistics]. *The Annals of Statistics*, *38*(2), 894–942. https://doi.org/10.1214/09-AOS729

Chapter 5

The Cartesian Gaussian additive noise model for directed network inference in omics data

Through the work in Chapter 4 and scalability improvements of Chapter 6 (which was being finished at the same time as this paper), we had confidence in the suitability and usability of Kronecker-sum-structured models. We had already been reconsidering and addressing the fundamental assumptions we were making in previous chapters (zero-mean assumption, Gaussian copula assumption, and Kronecker sum assumption), but one still remained: were conditional dependency networks the right networks to learn?

Conditional dependency networks have many advantages over correlation networks (discussed in Chapter 1), but there is one biological property that neither captures: gene regulatory networks are *directed*. In other words, gene A can regulate gene B without gene B having any regulatory effect on gene A. The most natural directed network is a causal network - which for Gaussian data is captured by the Cholesky factor of the precision matrix. Why not impose a Kronecker-sum structure to such factors? This would allow causal modelling in the multi-axis case.

Ultimately our results were limited, in large part because one is fundamentally restricted in how much causal inference one can do from observational data. We had to make some rather strong assumptions in order for the model to work - in particular, we assumed that the causal ordering was known beforehand (otherwise, the optimization problem becomes NP-Hard). As with most causal work, we assumed the causal graph was a directed acyclic graph (DAG), although genes are definitely capable of self-regulation.

In light of these restrictions, the following work should be considered a preliminary foray into the application of multi-axis methods to causal inference. It is a known (albeit surprising) fact that Gaussian distributions are not ideal for causal inference (Hoyer et al., 2008); this paper's 'CLGAM' model is very much a victim of this. Despite the limitations, the work in the paper does decisively demonstrate the benefits of multi-axis models; see for example Tables 5.1 and 5.2, in which our model handily outperforms its single-axis equivalent.

The paper is currently **accepted for publication** in BMC Medical Research Methodology. It is presented unchanged from the submitted version, except for formatting changes to fit the thesis style. The supplementary material for the paper is available in Appendix D, which contains further details and experiments. For further details and experiments that came up during the viva, see Appendix J.

The Cartesian Gaussian Additive Noise Model for Directed Network Inference in Omics Data

Bailey Andrew, David R. Westhead, and Luisa Cutillo

Under submission at BMC Medical Research Methodology

Abstract. Background Access to omics datasets, such as single-cell RNA-sequencing, enables us to estimate the regulatory networks governing the differentiation, proliferation, and interaction of cells in our body. Knowledge of such networks can give us valuable insight on the structure and progression of diseases; the difficulty is in estimating them. Most methods that estimate these networks either make an independence assumption ('the cells in your body do not interact'), or ignore the directional nature of gene regulation.

Methodology In this paper, we introduce the Cartesian Linear Gaussian Additive Noise Model to learn both cell-cell and gene-gene interactions. Our method is a statistical method; it is fit with maximum likelihood estimation, and we prove that a unique optimum always exists (under certain assumptions) using tools from high-dimensional statistics.

Results Our method differs from prior work in its lack of an independence assumption; we show that this leads to a real improvement in gene regulatory network and cell network reconstructions.

Conclusions We have developed and proved viable a novel method that learns directed gene regulatory networks, without assuming independence of cells. Our method is also extensible to more complicated omics datasets, such as longitudinal bulk RNA-sequencing datasets, through its ability to handle 'tensor-variate' datasets.

Trial Registration Clinical trial number: not applicable

5.1 Introduction

Knowledge of gene regulatory networks (GRNs) can give insights into potential therapeutic targets Madhamshettiwar et al., 2012, making them of keen interest to the fields of biology and medicine. There are many methods that estimate gene correlation networks (such as WCGNA Langfelder and Horvath, 2008), but such methods learn undirected networks that cannot not capture the directed nature of gene interactions. Others, such as ARACNE Margolin et al., 2006, are able to learn directed networks, but do not take into account that the cells come from a heterogeneous, interacting environment (i.e. it makes an *independence assumption*). This assumption weakens the utility of the learned network.

Recently, multi-axis methods like scBiGLasso Li et al., 2022 have been developed to account for this. Such methods, often called 'Kronecker-structured' methods due to their use of the Kronecker product, have the added benefit of *simultaneously learning a network of cells*. By analyzing such networks, the same tool can be used for cell subtyping and gene regulatory inference, and both networks benefit from knowledge of the other. However, none of these methods learn directed graphs.

In this paper, we present a method to simultaneously learn a directed gene and cell network from scRNA-seq data. In fact, our methodology can be applied to any matrix-variate dataset, such as other omics data, and even any 'tensor-variate dataset' (datasets that need three or more indices to fully describe, such as longitudinal bulk omics data). Our method enables us to extract meaningful insights on the genetic interactions and cellular makeup of our data.

We call our method the 'Cartesian linear Gaussian additive noise model', or $\boxplus$-LGAM for short. Our method is a type of additive noise model Hoyer et al., 2008.

5.2 Background: Kronecker-structured covariance estimation

Broadly, Kronecker-structured covariance estimation aims to estimate covariance (and related quantities) when the covariance matrix can be factorized according to some function. The most basic such model is the matrix normal distribution $\mathcal{MN}$. Letting $\text{vec}[\mathbf{X}]$ be the vector obtained by stacking the columns of a matrix, and $\otimes$ be the Kronecker product, the matrix normal distribution is:

$$\mathbf{X} \sim \mathcal{MN}\left(\mathbf{M}, \mathbf{\Sigma}_{\text{cells}}, \mathbf{\Sigma}_{\text{genes}}\right) \qquad \iff \text{vec}[\mathbf{X}] \sim \mathcal{N}\left(\text{vec}[\mathbf{M}], \mathbf{\Sigma}_{\text{cells}} \otimes \mathbf{\Sigma}_{\text{genes}}\right)$$

The factor matrices $\mathbf{\Sigma}_{\text{cells}}, \mathbf{\Sigma}_{\text{genes}}$ represent the covariances along their respective axes, and their Kronecker product represents the covariance structure of matrix elements. Covariance matrices imply a network structure over the genes (and cells): if $\mathbf{\Sigma}^{(genes)}_{ij} \neq 0$, then genes i and j are correlated, and hence connected in the correlation network.

Example 10. *The correlation of PAX5 and CD19 is found in $\mathbf{\Sigma}_{\text{genes}}$, the correlation of cell A and cell B is found in $\mathbf{\Sigma}_{\text{cells}}$, and the correlation between the expression of PAX5 in cell A and the expression of CD19 in cell B is found in $\mathbf{\Sigma}_{\text{cells}} \otimes \mathbf{\Sigma}_{\text{genes}}$.*

Rather than encoding Kronecker structure on the covariance matrix, it is often advantageous to encode it on the inverse covariance matrix (the 'precision' matrix). For Gaussian data, the sparsity structure of the precision matrix encodes the graph of 'conditional dependencies', which tend to be more meaningful than correlation.

Example 11. *In a dataset capturing ice cream sales, shark attacks, and seasonality, ice cream sales and shark attacks will be correlated (as both increase in the summer, due to heat and beach visits). However, they will be conditionally independent: once one conditions over seasonality, the correlation disappears.*

Various structures have been considered on precision matrices (Kronecker product, Kronecker sum (Kalaitzis et al., 2013), squared Kronecker sum (Wang & Hero, 2021; Wang et al., 2020), and the strong product (B. Andrew et al., 2025)). Such structures have also been imposed on the Laplacian of the graph of conditional dependencies (Shi & Mishne, 2024). For a more complete picture, we refer the reader to the excellent review by Wang et al., 2022.

Our model is similar to the squared Kronecker sum approach, although we optimize over directed acyclic graphs (DAGs) and they optimize over symmetric matrices. The Kronecker sum, denoted $\oplus$, is defined as $\mathbf{A} \oplus \mathbf{B} = \mathbf{A} \otimes \mathbf{I} + \mathbf{I} \otimes \mathbf{B}$. The Kronecker sum is interpretable as the *Cartesian product* ($\boxplus$) of the gene and cell graphs encoded by $\mathbf{A}$ and $\mathbf{B}$ (denoted $\mathfrak{A}$ and $\mathfrak{B}$); the graph of $\mathbf{A} \oplus \mathbf{B}$ is $\mathfrak{A} \boxplus \mathfrak{B}$. Imposing a Cartesian product structure is biologically meaningful; it corresponds to the assumption that genes can only directly regulate other genes in the same cell.

Example 12. *Suppose there are two cells and two genes of interest, PAX5 and CD19. In such a dataset, there are four different values being measured: PAX5 and CD19's expressions in the first cell, and PAX5 and CD19's expressions in the second.*

The Cartesian product assumption states that the expression of PAX5 in the first cell may be responsible for the level expression of CD19 in that same cell, but that it cannot be (directly) responsible for the level of expression of CD19 in the other cell. The level of PAX5 in both cells are allowed to be directly correlated (and likewise for CD19 in both cells); thus, indirect interactions may occur:

$$\text{PAX5}_{cell\ 1} \rightarrow \text{PAX5}_{cell\ 2} \rightarrow \text{CD19}_{cell\ 2}$$

It is only direct relationships that are forbidden.

Kronecker-structured problems often suffer from non-identifiability due to an underdetermined parameterization. For Kronecker sums (and hence our model), $\mathbf{A} \oplus \mathbf{B} = (\mathbf{A} + c\mathbf{I}) \oplus (\mathbf{B} - c\mathbf{I})$ for any c. Thankfully, this non-identifiability does not affect the sparsity structure of $\mathbf{A}, \mathbf{B}$ - only their diagonals. As we are only interested in the sparsity structure, this non-identifiability is not an issue.

5.3 Methodology: the Cartesian additive noise model

In this paper, we will assume that our scRNA-seq dataset takes the form of a matrix $\mathbf{X}$ with d_{cells} cells and d_{genes} genes (see Section 5.3.2 for a tensor-variate generalization). We then model that our dataset abides the following generative process with parameters $\{\mathbf{C}_i\}$:

$$\begin{aligned} \mathbf{X} &= \mathbf{C}_{\text{cells}}\mathbf{X} + \mathbf{X}\mathbf{C}_{\text{genes}}^T + \mathbf{E} \\ \mathbf{E}_{ij} &\sim_{i.i.d.} \mathcal{N}(0,1) \end{aligned} \tag{5.1}$$

This is a generalization of the common *linear Gaussian additive noise model* (LGAM), for which $\mathbf{X} = \mathbf{C}\mathbf{X} + \mathbf{E}$. We can also write the LGAM as:

$$\begin{aligned} x_0 &= \epsilon_0 \\ x_1 &= c_{10}x_0 + \epsilon_1 \\ &\vdots \\ x_N &= c_{N0}x_0 + \cdots + c_{N,N-1}x_1 + \epsilon_N \end{aligned} \tag{LGAM}$$

LGAM is a generative causal model. Let $\mathfrak{C}$ be the causal graph, i.e. $(j \to i) \in \mathfrak{C}$ means that 'gene j regulates gene i'. Under the LGAM assumption, then knowledge of $\mathbf{C}$ yields knowledge of $\mathfrak{C}$: $c_{ij} \neq 0 \iff (j \to i) \in \mathfrak{C}$ for $i \neq j$. c_{ii} controls the variance.

Example 13. *The causal graph $\{x_0 \to x_2 \to x_3$ and $x_1 \to x_3\}$ is represented as:*

$$\begin{aligned} x_0 &= \epsilon_0 \\ x_1 &= \epsilon_1 \\ x_2 &= c_{20}x_0 + \epsilon_2 \\ x_3 &= c_{31}x_1 + c_{32}x_2 + \epsilon_3 \end{aligned}$$

Learning causality from observational data is challenging; the gold standard will always be interventional experiments. However, there are so many potential gene-gene interactions that verifying all of them experimentally is prohibitive. Models such as LGAM can be used to shortlist potential candidates for such studies, and allow researchers to operate under plausible conjectures on the gene network in the meantime.

To learn the gene network, LGAM must assume that the cells are independent and identically distributed. Neither assumption is justifiable; cells come in distinct cell types and were sampled from a microenvironment in which they are all interacting. This motivates our model (Equation 5.1). Our model allows both the cells and the genes to have their own interactions, captured by $\mathbf{C}_{\text{cells}}$ and $\mathbf{C}_{\text{genes}}$. Our model makes four assumptions about the data:

Assumption 4. *The genes (resp. cells) interact linearly.*

Assumption 5. *The data is (or can be transformed to be) Gaussian.*

Assumption 6. *Gene/cell dependencies interact via the Cartesian product (see Example 12)*

Assumption 7. *$\mathfrak{C}_{\text{genes}}$ and $\mathfrak{C}_{\text{cells}}$ are directed acyclic graphs (DAGs) with known order.*

The first two assumptions are not egregious, and often hold approximately even when they fail. In Section 5.5.3, we will see that we still perform well on data that violates these assumptions. The third assumption is the result of the structure of our model; Equation 5.1 can be rewritten as:

$$\begin{aligned} \text{vec}[\mathbf{X}] &= \left(\mathbf{C}_{\text{genes}} \oplus \mathbf{C}_{\text{genes}}\right) \text{vec}[\mathbf{X}] + \text{vec}[\mathbf{E}] \\ \mathbf{E}_{ij} &\sim_{i.i.d.} \mathcal{N}(0,1) \end{aligned} \tag{5.2}$$

It is due to this relation that we call our model the *Cartesian Gaussian Additive Noise Model*, or $\boxplus$-LGAM for short. Making an assumption like this is unavoidable; the full space of possible interactions

in a scRNA-seq dataset is quite vast. In theory, any gene in any cell could interact with any other gene in any other cell! The problem is equivalent to estimating a covariance matrix from a single sample; without an assumption, this is statistically intractable. The Cartesian assumption is a natural one, and ensures tractability.

Our final assumption is that we know the causal order of genes, i.e. we know whether a gene is upstream or downstream of other genes in regulatory pathways. Unlike our other assumptions, this one is genuinely limiting. Without this assumption, fitting our model corresponds to a non-convex optimization problem. Furthermore, it is known that causal graphs for linear Gaussian data are fundamentally unlearnable beyond what is called the 'Markov equivalence class'; essentially, there would be multiple equally valid solutions. In Section 5.4.1, we present a method to infer causal ordering from the data, and in Section 5.5.2 we show that using the wrong order does not dramatically deteriorate performance.

5.3.1 Fitting the model

Given the causal ordering, we can without loss of generality assume $\mathbf{C}_\ell$ is lower triangular. For notational simplicity, let $d_{\setminus\ell} = \frac{\prod_{\ell'} d_{\ell'}}{d_\ell}$ and K be the number of axes in the data. For matrix-variate datasets, $K = 2$; see Section 5.3.2 for examples with larger K. Then, define $\mathbf{L}_\ell = \sqrt{d_{\setminus\ell}}\left(\frac{1}{K}\mathbf{I} - \mathbf{C}_\ell\right)$. $\mathbf{L}_\ell$ is a natural parameter to consider; it has a 1-to-1 relationship with $\mathbf{C}_\ell$ (in particular, it has the same sparsity structure) and corresponds to the 'Cholesky factor' of the precision matrix of the data. It is always invertible whenever $\mathbf{C}_\ell$ represents a DAG Shojaie and Michailidis, 2010 (which is guaranteed by being lower triangular). Finally, by defining $\mathbf{L} = \oplus_\ell \frac{1}{\sqrt{d_{\setminus\ell}}}\mathbf{L}_\ell$, we have that $\text{vec}\left[\mathbf{X}\right] \sim \mathcal{N}\left(\mathbf{0}, \mathbf{L}^{-T}\mathbf{L}^{-1}\right)$.

Letting $\mathbb{L}^{\oplus}_{++}$ be the space of all positive-diagonal lower-triangular matrices $\mathbf{L}$ that can be decomposed by the Kronecker sum, we can fit our model by finding the maximum likelihood of a normal distribution whose parameters lie in $\mathbb{L}^{\oplus}_{++}$:

$$\hat{\mathbf{L}} = \text{argmin}_{\mathbf{L}\in\mathbb{L}^{\oplus}_{++}} \|\mathbf{L}\text{vec}\left[\mathbf{X}\right]\|_2^2 - 2\sum_i \mathbf{L}_{ii} \tag{5.3}$$

Most real world networks are space; we also place an L1 penalty on the off-diagonals ($\lambda_\ell\|\mathbf{L}^{(\ell)}\|_{od}$) to ensure sparse solutions. $\lambda_\ell\|\cdot\|_{od}$ is frequently used in graphical estimation problems, such as the Graphical Lasso (Friedman et al., 2008). Additionally, we will see in Section 5.4 that, if our data is 'too lopsided' ($d_\ell \gg d_{\setminus\ell}$), a solution to this optimization problem is not guaranteed to exist. To counteract this, we add an optional Frobenius-norm penalty as well. Our final optimization problem, below, is thus analogous to ElasticNet (Zou & Hastie, 2005):

$$\hat{\mathbf{L}} = \text{argmin}_{\mathbf{L}\in\mathbb{L}^{\oplus}_{++}} \|\mathbf{L}\text{vec}\left[\mathbf{X}\right]\|_2^2 - 2\sum_i \mathbf{L}_{ii} + \sum_\ell \left(\rho_\ell\|\mathbf{L}^{(\ell)}\|_F^2 + \lambda_\ell\|\mathbf{L}^{(\ell)}\|_{od}\right) \tag{5.4}$$

5.3.2 Tensor-variate datasets

Most omics datasets are matrix-variate: a scRNA-seq dataset has genes and cells, scATAC-seq has peaks and cells, bulk RNA has samples and genes, etc. In these cases, the matrix-variate ⊞-LGAM will suffice. However, some datasets are more complicated. A longitudinal study of gene expression in a cohort of patients would naturally lend itself to a tensor-variate approach: the dataset $\mathcal{X}$ would have three axes, one for time, one for genes, and one for patients.

Our model is readily generalized to such a scenario. Tensor notation can be complex; we refer the reader to Kolda and Bader, 2009 for a full overview. Here, we need the concept of the ℓ-mode product $\times_\ell$, which intuitively corresponds to multiplying a tensor along the axis ℓ. The formal definition can be hard to parse; $[\mathcal{X} \times_\ell \mathbf{M}]_{i_1 i_2 \cdots i_\ell \cdots i_K} = \sum_{j_\ell} \mathcal{X}_{i_1 i_2 \cdots i_{\ell-1} j_\ell i_{\ell+1} \cdots i_K} \mathbf{M}_{j_\ell i_\ell}$. For matrices, $\mathbf{A} \times_1 \mathbf{B} = \mathbf{B}^T\mathbf{A}$ and $\mathbf{A} \times_2 \mathbf{B} = \mathbf{A}\mathbf{B}$.

$$\mathcal{X} = \left(\sum_{\ell}^{K} \mathcal{X} \times_{\ell} \mathbf{C}_{\ell}^{T} \right) + \mathcal{E} \qquad \text{(Tensor-variate } \boxplus\text{-LGAM)}$$

$$\mathcal{E}_{\mathbf{i}} \sim_{i.i.d.} \mathcal{N}(\mathbf{0}, \mathbf{I}) \tag{5.5}$$

Other than having more parameters to learn, the model is functionally identical to the matrix-variate case; it makes the same assumptions and is fit in the same manner.

5.4 Theory: establishing the efficacy of the method

In Section 5.3, we made four assumptions:

Assumption 1. *The genes (resp. cells) interact linearly.*

Assumption 2. *The data is (or can be transformed to be) Gaussian.*

Assumption 3. *Gene/cell dependencies interact via the Cartesian product (see Example 12)*

Assumption 4. $\mathfrak{C}_{\text{genes}}$ *and* $\mathfrak{C}_{\text{cells}}$ *are directed acyclic graphs (DAGs) with known order.*

These govern whether or not our model is correct to use on a dataset, but do not allow us to quantify *how well* our model can be expected to do. For this, we need to make two more assumptions.

Assumption 5. *Our dataset is not too lopsided, i.e. the largest axis* $d_{\max}$ *is not much greater than the other axes. Formally,* $K - 1 + \prod_{\ell} d_{\ell} \geq \sum_{\ell} \frac{d_{\ell}^2 - d_{\ell}}{2}$*, but the following are simpler sufficient conditions:*

- *For matrix-variate data,* $d_{\max} \leq d_{\min} + \frac{1+\sqrt{9+8d_{\min}}}{2}$ *(both necessary and sufficient)*
- *For tensor-variate data,* $d_{\setminus \max} \geq \frac{K}{2} d_{\max}$

Assumption 6. *Our dataset is* α*-sub-exponential, with* $\alpha \in (0, 2]$*.*

Definition (α-sub-exponentiality)**.** *A distribution* $\mathcal{D}$ *is* α*-sub-exponential if for* $\alpha > 0$*, some constant* c *and all* $t > 0$*,* $\mathbb{P}_{x \sim \mathcal{D}} \left(\left| x - \mathbb{E}[x] \right| \geq t \right) \leq 2e^{\frac{-t^{\alpha}}{c}}$*. In other words,* α*-sub-exponential distributions have light tails.*

Example 14. *All sub-Gaussian distributions, including the Gaussian distribution and all bounded distributions, are* $(\alpha \geq 2)$*-sub-exponential. The Poisson and negative binomial distributions are* $(\alpha = 1)$*-sub-exponential.*

Example 15. *The Cauchy, log-normal, and t distributions are* ***not*** α*-sub-exponential.*

Note that the α-sub-exponentiality assumption is more general than the Gaussian assumption, and encompasses many distributions typically used to model scRNA-seq data (such as the Poisson and negative binomial distributions). Making this assumption allows us to prove error bounds on our model, described by Theorem 6.

The statement of Theorem 6 is dense. The details may not be of interested to every user, so we will first summarize the essence of the statement. The theorem is proven in the supplementary material using standard high-dimensional statistics techniques, including the generalized Hanson-Wright inequalities of Sambale, 2023.

Theorem (Theorem 6, paraphrased)**.** *With exponentially high probability, we can bound the error of our model in terms of our regularization parameters* ρ*,* λ*. This gives us faith in our model's utility.*

Theorem 6. *Let* $\lambda_{\ell} = \lambda$ *and* $\rho_{\ell} = \rho$ *for each axis* ℓ*. Assume our data is* α*-sub-exponential with* $\alpha \in (0, 2]$*. Suppose the largest Frobenius norm of any factor of the oracle solution*[1] $\mathbf{L}_{\rho}^{*}$ *is* R_F*, the total*

[1] An 'oracle solution' is the best possible parameter choice (measured by Kullback-Leiber divergence from the true distribution), and equals the true parameter choice when the model is not misspecified.

amount of nonzero elements of the factors of the regularized oracle solution is s, and μ is the strong convexity constant of Equation 5.3. Let p be a parameter we choose to control the probability of success, and β be defined shortly. Let $\lambda \asymp \max_\ell \sqrt{\frac{(p \log d_\ell)^\beta}{d_{\setminus \ell}}} + \sum_{\ell' \neq \ell} \sqrt{\frac{(p \log d_\ell)^\beta}{d_\ell d_{\ell'}}}$, then with probability at least $1 - \sum_\ell 2^{1 - p \log d_\ell}$:

$$\|\hat{\mathbf{L}} - \mathbf{L}^*\|_{F,\oplus} \leq \frac{2K\rho}{\mu} R_F + \frac{3\lambda\sqrt{s}}{\mu + \rho}$$

For matrix-variate data, let $\lambda \asymp \max\left(\sqrt{\frac{(\log d_1)^\beta}{d_2}}, \sqrt{\frac{(\log d_2)^\beta}{d_1}}\right)$.

β is either 1 or $\frac{4}{\alpha}$. $\beta = 1$ corresponds to the 'typical regime', and $\beta = \frac{4}{\alpha}$ applies when p is chosen to be very large. We derive an expression for the boundary between the two regimes in the supplementary material, but, for reasonable[2] p, $\beta = 1$ should suffice.

Corollary 1. *Suppose that the smallest value of the oracle solution $\mathbf{L}^*$ is larger than $\frac{2K\rho}{\mu} R_F + \frac{3\lambda\sqrt{s}}{\mu+\rho}$. Then (under Theorem 6's assumptions), our model is 'sparsistent': with exponentially high probability, the support, and hence the true network $\mathfrak{C}$, is recovered correctly.*

Our other assumption concerns the 'lopsidedness' of our data. It is natural to expect some kind of restriction here: suppose we had 1,000,000 cells and 5 genes. How are we possibly going to estimate a cell network which has $\approx \frac{10^{12}}{2}$ parameters from only 5 genes? One axis cannot be too much larger than the other, or else it will be too difficult to estimate that axis' network. This is not as limiting as it first seems: we will see that, if we choose $\rho_\ell > 0$ in Equation 5.4, then lopsidedness is not a concern.

Theorem 7 (Lopsidedness Theorem)**.** *As long as the dataset is not too lopsided and the true distribution has positive definite covariance, then Equation 5.4 defines a function that is almost surely strongly convex when every $\rho_\ell = 0$. Equation 5.4 is always strongly convex when $\rho_\ell > 0$.*

Corollary 2. *When the conditions of Theorem 7 are satisfied, there exists a unique solution to Equation 5.4.*

The proof relies on Cover's Theorem Cover, 1965, and is given in the supplementary material. Even when Theorem 7 fails, a solution can still exist with high probability.

Theorem 8 (Excess Lopsidedness Theorem)**.** *Let $d_\forall = \prod_\ell d_\ell$ and $D = \sum_\ell \frac{d_\ell^2 - d_\ell}{2} - K + 1 - d_\forall$, and suppose $D > 0$; in other words, D represents the 'excess lopsidedness' of the data. For $(d_1, d_1 + \delta)$-shaped (matrix-variate) data, $D = \frac{\delta^2 - \delta}{2} - (d_1 + 1)$.*

With probability $1 - \frac{1}{2^{d_\forall - 1}} \sum_{i=0}^{\min(D, d_\forall))-1} \binom{d_\forall - 1}{i}$ a solution exists when $\rho_\ell = 0$ to Equation 5.4.

Example 16. *Assume Frobenius penalties ρ_ℓ of zero. For an arbitrary 100×114 dataset, Equation 5.4 almost surely has a solution. For a 100×263 dataset it has at least a 50% chance of having a solution. In all scenarios in which $\rho_\ell > 0$, a solution exists.*

Example 17. *Tensor-variate data is permissive. A solution almost surely exists for $100 \times 100 \times 6000$ and $100 \times 100 \times 100 \times 500000$ datasets.*

In Section 5.5.3, we will see that our model is able to perform well on lopsided data when $\rho_\ell > 0$. The dataset in question has 640 cells and 11 genes, in which case Equation 5.4 has *no solution* when all ρ_ℓ are 0 (Theorem 8).

5.4.1 Inferring the ordering

The most egregious assumption we make is knowledge of the ordering. When the order is unknown, we could change Equation 5.4 to optimize over the space of all DAGs, rather than lower-triangular matrices. However, this problem is NP-hard (Chickering et al., 2004). In this section, we'll show that, under certain conditions, the ordering can be recovered from the data.

For the LGAM model, the ordering of the variables can be inferred from the variance of the data.

[2] If α is close to 0 (i.e. the true distribution has heavier tails) or if the true distribution's covariance matrix is ill-conditioned, then $\beta = \frac{4}{\alpha}$ is more likely.

Theorem (Various Authors)**.** *Let* $\mathbf{x}$ *come from the LGAM model. Let* $i < i'$ *denote that there is a path from* i *to* i' *in the DAG, i.e.* i *comes earlier in the ordering. Then, under various conditions (Park, 2020; Peters & Bühlmann, 2014), we have that:*

$$i < i' \implies \text{Var}[\mathbf{x}_i] < \text{Var}[\mathbf{x}_{i'}]$$

One such condition is that the diagonals of each $\mathbf{C}_\ell$ *are homogeneous.*

In fact, any such result like this can be adapted to fit our ⊞-LGAM model.

Theorem 9. *Let* $\mathcal{X}$ *come from the ⊞-LGAM model. Under various conditions (Park, 2020; Peters & Bühlmann, 2014), we have that:*

$$i < i' \implies \mathbb{E}\left[\sum_{\mathbf{j}} \mathcal{X}_{i\mathbf{j}}^2\right] < \mathbb{E}\left[\sum_{\mathbf{j}} \mathcal{X}_{i'\mathbf{j}}^2\right]$$

One such condition is that the diagonals of each $\mathbf{C}_\ell$ *are homogeneous.*

Example 18. *Let's assume that* $\text{diag}[\mathbf{C}_\ell] = a_\ell \mathbf{I}$ *for some constants* $\{a_\ell\}$*. This is equivalent to assuming that every gene has the same variance and every cell has the same variance after conditioning out the effect of other genes/cells. Then one should order the terms (in ascending order) by their empirical variance.*

In the real world, gene variances are unlikely to be homogeneous. Other, more complicated assumptions on the variance will suffice for Theorem 9 (Park, 2020). Additionally, the variance heuristic can be combined with domain knowledge; one can first order genes by whether or not they are transcription factors before applying this heuristic, for example. In Section 5.5.2, we show that, even if the chosen ordering is *dramatically* wrong, the recovered network still maintains a good degree of accuracy.

5.5 Results

It is often hard to validate graphical models, given that we rarely have access to a ground truth network. We'll analyze our performance on two types of synthetic data: that which we generate ourselves from simple distributions to ensure a given structure (Section 5.5.1) and a simulated scRNA-seq dataset from the literature (Section 5.5.3). We'll call the former 'synthetic' and the latter 'simulated', to reflect the fact that the latter dataset is an expert-curated simulation of real-world data. We also investigate the effects of getting the ordering wrong (Section 5.5.2).

5.5.1 Synthetic data

Each of the synthetic datasets we will experiment with are generated from our ⊞-LGAM model. We focus on in-distribution data in this section to get an understanding of the idealized performance of our model; out-of-distribution data is the focus of Section 5.5.3. We do not explore the effect of ρ here, but will in Section 5.5.3. We let our L1, λ, penalty be the same for all axes.

We investigate many different scenarios, based on the size, sparsity and lopsidedness of the dataset. Typically, our ground truth is generated to be an Erdös-Renyi graph (i.e, each potential edge has a constant probability to be present in the graph, independent of other edges), although we also look at AR(1) graphs. We summarize results in Table 5.1, using standard LGAM and GmGM (E. B. Andrew et al., 2024) as baselines. GmGM estimates the (symmetric) graph of conditional dependencies (Section 5.2) rather than a directed graph, but it is also based on the Cartesian product.

Our method performs very well for tensor-variate and/or AR(1) data. In most cases, our method outperformed the baselines. Our method achieved worse results on highly-sparse-graph data compared to the baseline. This is not surprising; when the sample graph is highly sparse, the dataset is very close to being independent. When the samples truly are independent, LGAM suffices and our model may overfit. GmGM sometimes does best, but is typically much worse than LGAM and ⊞-LGAM.

When comparing the effect of dataset size on performance, we considered three different scenarios, going from a size of 50×50 to 100×100. These correspond to three different scenarios for the amount of edges s in the true growth.

Dimension	AUC-PR			MCC		
	LGAM	⊞-LGAM	GmGM	LGAM	⊞-LGAM	GmGM
$100_{\text{ER}} \times 50_{\text{AR(1)}}$	0.778	**0.823**	0.247	0.726	**0.781**	0.369
$40_{\text{ER}} \times 40_{\text{ER}} \times 40_{\text{ER}}$	0.791	**0.801**	0.648	0.726	**0.756**	0.674
$60_{\text{ER}} \times 60_{\text{ER}} \times 60_{\text{ER}}$	0.758	**0.775**	0.688	0.684	**0.707**	0.650
$50_{\text{ER}} \times 50_{\text{AR(1)}}$	0.680	**0.762**	0.370	0.670	**0.715**	0.552
$50_{\text{ER}} \times 100_{\text{AR(1)}}$	0.468	**0.598**	0.138	0.600	**0.676**	0.309
2.5% $100_{\text{ER}} \times 100_{\text{ER}}$	0.417	**0.427**	0.272	0.508	0.504	**0.527**
$100_{\text{AR(1)}} \times 50_{\text{ER}}$	0.367	**0.386**	0.288	0.344	**0.346**	0.275
$50_{\mathcal{I}} \times 50_{\text{ER}}$	0.348	**0.354**	0.316	**0.335**	0.330	0.328
$50_{\text{ER}} \times 50_{\text{ER}}$	0.315	0.328	**0.345**	0.291	0.321	**0.333**
5% $100_{\text{ER}} \times 100_{\text{ER}}$	0.270	**0.316**	0.272	0.293	0.347	**0.383**
$100_{\text{ER}} \times 50_{\text{ER}}$	0.297	**0.308**	0.276	0.313	**0.333**	0.283
10% $100_{\text{ER}} \times 100_{\text{ER}}$	0.250	**0.279**	0.239	0.241	**0.264**	0.237
$50_{\text{ER}} \times 100_{\text{ER}}$	0.209	**0.226**	0.175	0.188	**0.207**	0.152
$50_{\text{AR(1)}} \times 100_{\text{ER}}$	**0.197**	0.190	0.164	**0.175**	0.159	0.133

Table 5.1: Results on synthetic data, ordered by problem difficulty. We report both the area under the precision-recall curve (AUC-PR) and the best Matthew's correlation coefficient (MCC) achieved by the methods. Results are reported on the second axis of the dataset - for the lopsided 100×50 and 50×100 datasets, note that the first-axis performance of 50×100 is the same as the second-axis performance of 100×50. All ground truth graphs were 10% dense, except if labeled 5% dense (matching Scenario 2), or 2.5% dense (matching Scenario 3). The subscript of an axis indicates the type of graph used (ER is Erdos-Renyi, AR(1) is autoregressive, and $\mathcal{I}$ is independent/an empty graph).

Ordering	AUC-PR		MCC	
	LGAM	⊞-LGAM	LGAM	⊞-LGAM
True Ordering	0.778	**0.823** (↑6%)	0.726	**0.781** (↑8%)
20% Shuffled	0.776	**0.824** (↑6%)	0.726	**0.777** (↑7%)
50% Shuffled	0.730	**0.773** (↑6%)	0.709	**0.751** (↑6%)
100% Shuffled	0.728	**0.798** (↑10%)	0.745	**0.768** (↑3%)

Table 5.2: Results on synthetic data; the same dataset is used for each, with only the ordering changed. We report both the area under the precision-recall curve (AUC-PR) and the best Matthew's correlation coefficient (MCC). Results are reported on the second axis of the dataset. All tests were run on the $100_{\text{ER}} \times 50_{\text{AR(1)}}$ Gaussian test from Table 5.1. An ordering is x% shuffled if we randomly shuffled the first x% of the ordering.

Scenario 1. *The percentage of extant edges stays the same;* $s = O(d_\ell^2)$.

Scenario 2. *The average degree of nodes stays the same;* $s = O(d_\ell)$.

Scenario 3. *The amount of total edges stays the same;* $s = O(1)$.

According to Theorem 6, the error for a $d \times d$ dataset should grow as $O(\sqrt{s})$; thus, we'd expect the performance to decrease in Scenario 1 and increase in Scenario 3; this is exactly what we see.

5.5.2 Effect of ordering

When the ordering is unknown, we'd like to know how much the choice of ordering is likely to affect the sparsity structure of the recovered graph. We empirically test the effect of getting the ordering wrong on synthetic data, in Table 5.2. These results demonstrate that getting the ordering wrong

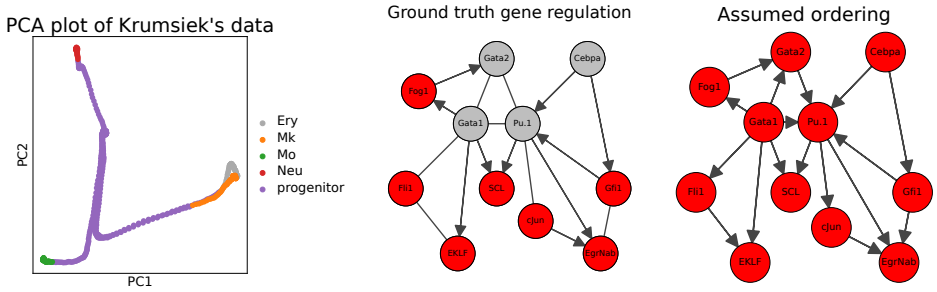


Figure 5.1: (Left) The first two PCA components of the Krumsiek dataset, demonstrating clear developmental trajectories. (Center) The regulatory network used to generate the dataset. Some edges in their network were bi-directional; these have been represented as undirected edges. Gata1, Gata2, Pu.1, and Cebpa have 'self loops'. (Right) The ground truth graph with our assumed ordering.

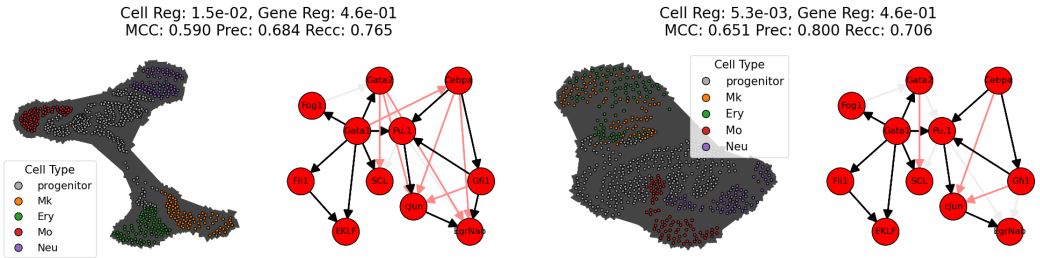


Figure 5.2: (Left) A gene graph arising from our ⊞-LGAM's grid search. Chosen for display as its cell graph qualitatively captures cell development the best. (Right) The best gene graph arising from our ⊞-LGAM model's grid search. (All) Black edges are true positives, near-white edges are false negatives, pale red edges are false positives.

does not dramatically affect structure recovery. If the ordering unknown, heuristics and theorems that allow us to approximate the ordering, such as Theorem 9, are helpful, but not necessary - even with a completely random ordering, structure recovery capability is mostly preserved.

5.5.3 Simulated scRNA-seq data

In this section, we apply our method to the simulated dataset of Krumsiek et al., 2011. The dataset in question (the 'Krumsiek dataset') was generated from a literature-curated regulatory network of 11 genes involved in blood cell differentiation. The regulatory network is not acyclic: it contains some loops (Figure 5.1(Center)).

The dataset contains 640 cells, which are labeled by cell type (erythrocyte, megakaryocyte, monocyte, neutrophil, and progenitor). The dataset has a notable 'developmental structure': progenitor cells differentiate into the other cells; in Figure 5.1(Left) we can see this developmental process manifest in PCA-space.

We compare our performance against the LGAM baseline. The task for the gene axis is clear; we wish to know how well our method recovers the structure of the ground truth graph. For the cell axis, this is less clear. Given that we have access to cell labels, we evaluate how well cells of the same type cluster in our graphs. We are also interested on how well, qualitatively, our graphs recover the developmental structure of the dataset.

5.5.4 Establishing causal ordering

To determine if a gene comes before another, we apply the following three rules in order of priority:

1. Must gene A come before gene B due to prior knowledge?
2. If a gene self-regulates, it comes before genes that do not.

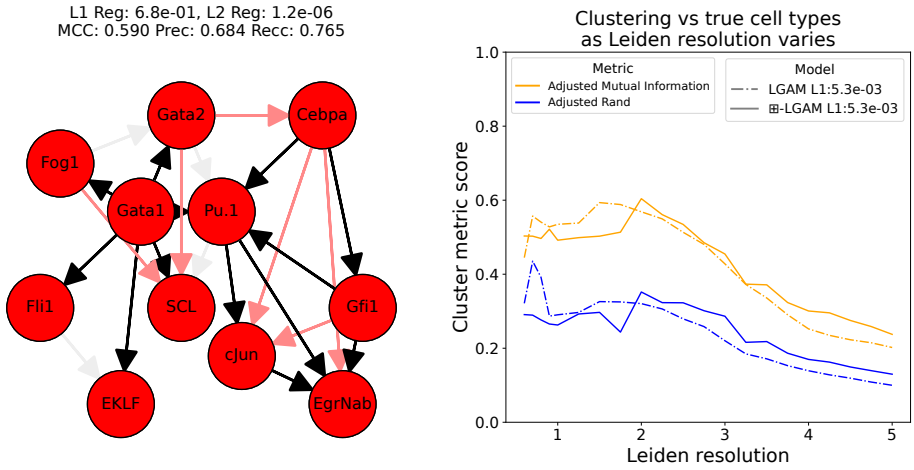


Figure 5.3: (Left) The best gene graph arising from the baseline LGAM's grid search. Black edges are true positives, near-white edges are false negatives, pale red edges are false positives. (Right) Clustering metrics comparing the best ⊞-LGAM results to the best baseline LGAM result.

3. Use Theorem 9 to determine the ordering.

As the ground truth gene graph was constructed by Krumsiek et al., 2011 using the academic literature, we'll assume knowledge of the orderings that are entailed by the graph. The ground truth graph contains both uni-directional and bi-directional edges, so not every ordering is known. For example, the graph contains Gata1 → Fog1 and Gata1 ↔ Fli1. Thus, Gata1 must come before Fog1, but we do not know the ordering of Gata1 and Fli1. We will also assume knowledge about which genes self-regulate (Gata1, Gata2, Pu.1, Cebpa). These genes are likely to have higher variance due to this, which would interfere with Theorem 9. After following this process, we get the ordering:

$$\text{Gata1}\rightarrow\text{Fog1}\rightarrow\text{Gata2}\rightarrow\text{Cebpa}\rightarrow\text{Gfi1}\rightarrow\text{Pu.1}\rightarrow\text{EKLF}\rightarrow\text{cJun}\rightarrow\text{EgrNab}\rightarrow\text{SCL}$$

What about the ordering of the cells? We order our cells by their 'pseudotime'. Pseudotime is a measure of how far along its developmental trajectory a given cell is. scRNA-seq datasets contain only a single snapshot in time (hence the 'pseudo' in 'pseudotime'), but not every cell ends up developing at the same rate. Thus, scRNA-seq datasets can often capture the developmental trajectory of a cell type (in this case, the differentiation of progenitors into the other cell types).

To estimate pseudotime, we used PAGA (Wolf et al., 2019) to construct the coarse, large-scale structure of development and DPT (Haghverdi et al., 2016) to assign each cell a distance along this trajectory. We chose this method as it is the standard workflow in the scanpy package (Wolf et al., 2018), which we used for our experiments.

5.5.5 Choosing the parameters

For our ⊞-LGAM model, we first grid-searched over the two L1 penalties λ_1, λ_2, with fixed ρ. As we don't have a quantitative metric for the cell graph's performance, we used the performance (Matthew's correlation coefficient) on the gene graph to rank results. We then picked the best-performing result, and performed an extended grid-search over ρ. For the baseline, we grid-searched over λ and ρ. See Figure A in the supplementary material for grid search results, and Figure 5.3(Left) for the best baseline gene graph.

To investigate the ability of the cell graph to qualitatively capture the developmental structure, we looked at the three best performing cell graphs from the original grid search, as well as the best performing cell graph from the extended grid search. Figure 5.2(Left) displays the one we, subjectively, felt captured developmental structures the best. For a more objective measure, we evaluated cluster quality when using the Leiden clustering algorithm (Traag et al., 2019), given in Figure 5.3(Right).

One benefit of our model is that the gene and cell objectives are linked; if we have a metric for gene graph performance, we can optimize for that and produce a corresponding cell graph. For standard

LGAM, we have no way of doing so. To be able to compare to a baseline, we ran LGAM three times, once for each of the top 3 cell parameters from our ⊞-LGAM grid search.

The results are favorable to our model; our best gene graph achieved an MCC of 0.651, compared to 0.590 for the baseline. By optimizing only over gene graph performance, we automatically learned cell graphs as well, which in most cases outperformed the baseline in clustering quality measured by adjusted mutual information and adjusted Rand scores.

5.6 Discussion

This paper addresses the problem of directed regulatory network inference. In particular, we have generalized the classical linear Gaussian additive noise model using a Kronecker-separable framework. Our method achieves a standard statistical recovery rate; we have not lost anything asymptotically by generalizing the model (Theorem 6). By learning all graphs simultaneously, it allows metrics for one graph to be used to learn the other graphs (Section 5.5.3). Our model is a generative model; thus, it can be used to generate plausible causal hypotheses for further investigation.

5.7 Declarations

5.7.1 Ethis approval and consent to participate

Not applicable

5.7.2 Consent for publication

Not applicable

5.7.3 Availability of data and materials

The code and seeds for datasets generated and/or analysed during the current study are available in our GitHub repository, https://github.com/BaileyAndrew/Cartesian-LGAM, including that of Krumsiek et al. Krumsiek et al., 2011.

We implemented our algorithm in Python, using NumPy (Harris et al., 2020), on a personal computer (2020 M1 Mac with 8GB of RAM); the code for all experiments is available at the same repository.

5.7.4 Competing interests

No competing interest is declared.

5.7.5 Funding

There was no direct funding for this research. B.A. is supported by the UKRI Engineering and Physical Sciences Research Council (EPSRC) [EP/S024336/1].

5.7.6 Author contributions statement

B.A. conceived the model, and wrote the proofs and manuscript. L.C. contributed to the theory and experiments. D.W. provided guidance on applications and experiments.

5.7.7 Acknowledgements

Thank you to all the people and animals who have donated their time and data for the furthering of medical research.

Bibliography for Paper 4

Andrew, B., Westhead, D. R., & Cutillo, L. (2025). The Strong Product Model for Network Inference without Independence Assumptions [ISSN: 2640-3498]. *Proceedings of The 28th International Conference on Artificial Intelligence and Statistics*, 5230–5238. Retrieved July 14, 2025, from https://proceedings.mlr.press/v258/andrew25a.html

Andrew, E. B., Westhead, D., & Cutillo, L. (2024). GmGM: A fast multi-axis Gaussian graphical model [ISSN: 2640-3498]. *Proceedings of The 27th International Conference on Artificial Intelligence and Statistics*, 2053–2061. Retrieved September 8, 2025, from https://proceedings.mlr.press/v238/andrew24a.html

Chickering, D. M., Heckerman, D., & Meek, C. (2004). Large-Sample Learning of Bayesian Networks is NP-Hard. *Journal of Machine Learning Research*, *5*(Oct), 1287–1330. Retrieved April 7, 2025, from https://www.jmlr.org/papers/v5/chickering04a.html

Cover, T. M. (1965). Geometrical and Statistical Properties of Systems of Linear Inequalities with Applications in Pattern Recognition. *IEEE Transactions on Electronic Computers*, *EC-14*(3), 326–334. https://doi.org/10.1109/PGEC.1965.264137

Friedman, J., Hastie, T., & Tibshirani, R. (2008). Sparse inverse covariance estimation with the graphical lasso. *Biostatistics (Oxford, England)*, *9*(3), 432–441. https://doi.org/10.1093/biostatistics/kxm045

Haghverdi, L., Büttner, M., Wolf, F. A., Buettner, F., & Theis, F. J. (2016). Diffusion pseudotime robustly reconstructs lineage branching [Publisher: Nature Publishing Group]. *Nature Methods*, *13*(10), 845–848. https://doi.org/10.1038/nmeth.3971

Harris, C. R., Millman, K. J., van der Walt, S. J., Gommers, R., Virtanen, P., Cournapeau, D., Wieser, E., Taylor, J., Berg, S., Smith, N. J., Kern, R., Picus, M., Hoyer, S., van Kerkwijk, M. H., Brett, M., Haldane, A., del Río, J. F., Wiebe, M., Peterson, P., ... Oliphant, T. E. (2020). Array programming with NumPy [Publisher: Nature Publishing Group]. *Nature*, *585*(7825), 357–362. https://doi.org/10.1038/s41586-020-2649-2

Hoyer, P., Janzing, D., Mooij, J. M., Peters, J., & Schölkopf, B. (2008). Nonlinear causal discovery with additive noise models. *Advances in Neural Information Processing Systems*, *21*. Retrieved April 6, 2025, from https://papers.nips.cc/paper_files/paper/2008/hash/f7664060cc52bc6f3d620bcedc94a4b6-Abstract.html

Kalaitzis, A., Lafferty, J., Lawrence, N. D., & Zhou, S. (2013). The Bigraphical Lasso [ISSN: 1938-7228]. *Proceedings of the 30th International Conference on Machine Learning*, 1229–1237. Retrieved March 15, 2025, from https://proceedings.mlr.press/v28/kalaitzis13.html

Kolda, T. G., & Bader, B. W. (2009). Tensor Decompositions and Applications. *SIAM Rev.*, *51*(3), 455–500. https://doi.org/10.1137/07070111X

Krumsiek, J., Marr, C., Schroeder, T., & Theis, F. J. (2011). Hierarchical Differentiation of Myeloid Progenitors Is Encoded in the Transcription Factor Network [Publisher: Public Library of Science]. *PLOS ONE*, *6*(8), e22649. https://doi.org/10.1371/journal.pone.0022649

Langfelder, P., & Horvath, S. (2008). WGCNA: An R package for weighted correlation network analysis. *BMC bioinformatics*, *9*, 559. https://doi.org/10.1186/1471-2105-9-559

Li, S., López-García, M., Lawrence, N. D., & Cutillo, L. (2022). Two-way Sparse Network Inference for Count Data [ISSN: 2640-3498]. *Proceedings of The 25th International Conference on Artificial Intelligence and Statistics*, 10924–10938. Retrieved November 7, 2025, from https://proceedings.mlr.press/v151/li22g.html

Madhamshettiwar, P. B., Maetschke, S. R., Davis, M. J., Reverter, A., & Ragan, M. A. (2012). Gene regulatory network inference: Evaluation and application to ovarian cancer allows the prioritization of drug targets. *Genome Medicine*, *4*(5), 41. https://doi.org/10.1186/gm340

Margolin, A. A., Nemenman, I., Basso, K., Wiggins, C., Stolovitzky, G., Dalla Favera, R., & Califano, A. (2006). ARACNE: An algorithm for the reconstruction of gene regulatory networks in a mammalian cellular context. *BMC bioinformatics*, *7 Suppl 1*(Suppl 1), S7. https://doi.org/10.1186/1471-2105-7-S1-S7

Park, G. (2020). Identifiability of Additive Noise Models Using Conditional Variances. *Journal of Machine Learning Research*, *21*(75), 1–34. Retrieved April 8, 2025, from http://jmlr.org/papers/v21/19-664.html

Peters, J., & Bühlmann, P. (2014). Identifiability of Gaussian structural equation models with equal error variances. *Biometrika*, *101*(1), 219–228. https://doi.org/10.1093/biomet/ast043

Sambale, H. (2023). Some Notes on Concentration for α-Subexponential Random Variables. In R. Adamczak, N. Gozlan, K. Lounici, & M. Madiman (Eds.), *High Dimensional Probability IX* (pp. 167–192). Springer International Publishing. https://doi.org/10.1007/978-3-031-26979-0_7

Shi, C., & Mishne, G. (2024). Learning Cartesian Product Graphs with Laplacian Constraints [ISSN: 2640-3498]. *Proceedings of The 27th International Conference on Artificial Intelligence and Statistics*, 2521–2529. Retrieved March 23, 2025, from https://proceedings.mlr.press/v238/shi24a.html

Shojaie, A., & Michailidis, G. (2010). Penalized likelihood methods for estimation of sparse high-dimensional directed acyclic graphs. *Biometrika*, *97*(3), 519–538. https://doi.org/10.1093/biomet/asq038

Traag, V. A., Waltman, L., & van Eck, N. J. (2019). From Louvain to Leiden: Guaranteeing well-connected communities [Publisher: Nature Publishing Group]. *Scientific Reports*, *9*(1), 5233. https://doi.org/10.1038/s41598-019-41695-z

Wang, Y., & Hero, A. (2021). SG-PALM: A Fast Physically Interpretable Tensor Graphical Model [ISSN: 2640-3498]. *Proceedings of the 38th International Conference on Machine Learning*, 10783–10793. Retrieved March 15, 2025, from https://proceedings.mlr.press/v139/wang21k.html

Wang, Y., Jang, B., & Hero, A. (2020). The Sylvester Graphical Lasso (SyGlasso) [ISSN: 2640-3498]. *Proceedings of the Twenty Third International Conference on Artificial Intelligence and Statistics*, 1943–1953. Retrieved March 15, 2025, from https://proceedings.mlr.press/v108/wang20d.html

Wang, Y., Sun, Z., Song, D., & Hero, A. (2022). Kronecker-structured covariance models for multiway data [Publisher: Amer. Statist. Assoc., the Bernoulli Soc., the Inst. Math. Statist., and the Statist. Soc. Canada]. *Statistics Surveys*, *16*(none), 238–270. https://doi.org/10.1214/22-SS139

Wolf, F. A., Angerer, P., & Theis, F. J. (2018). SCANPY: Large-scale single-cell gene expression data analysis. *Genome Biology*, *19*(1), 15. https://doi.org/10.1186/s13059-017-1382-0

Wolf, F. A., Hamey, F. K., Plass, M., Solana, J., Dahlin, J. S., Göttgens, B., Rajewsky, N., Simon, L., & Theis, F. J. (2019). PAGA: Graph abstraction reconciles clustering with trajectory inference through a topology preserving map of single cells. *Genome Biology*, *20*(1), 59. https://doi.org/10.1186/s13059-019-1663-x

Zou, H., & Hastie, T. (2005). Regularization and Variable Selection Via the Elastic Net. *Journal of the Royal Statistical Society Series B: Statistical Methodology*, *67*(2), 301–320. https://doi.org/10.1111/j.1467-9868.2005.00503.x

Chapter 6

Making multi-axis Gaussian graphical models scalable to millions of samples and features

This chapter and Chapter 2 neatly bookend the thesis. Chapter 2 improved the scalability of Kronecker-sum-structured models, but not to the point of being applicable on the largest of datasets. The paper in this chapter goes much further than the previous, with the improvement of GmGM to such a point that it can handle millions of cells in minutes. At the start of this thesis, state-of-the-art algorithms such as TeraLasso began struggling in the low thousands of cells, but over the course of the PhD we continually improved our methodology to the point where it can run on any current public scRNA-seq dataset; we prove this by running our method on one of the largest such datasets.

While this paper is primarily a methodology paper, it also contains the most in-depth experiments on real datasets performed during this thesis. This mirrors the fact that multi-axis methods have now become applicable to real data; before, toy examples were prevalent due to methodological limitations.

The paper is currently **under submission** in Bioinformatics. It is presented unchanged from the submitted version, except for formatting changes to fit the thesis style. The supplementary material for the paper is available in Appendix E, which contains further details and experiments. For further details and experiments that came up during the viva, see Appendix K.

Making Multi-Axis Gaussian Graphical Models Scalable to Millions of Cells

Bailey Andrew, Erica L. Harris, Hames A. Poulter, David R. Westhead, Luisa Cutillo

Under submission at Bioinformatics

Abstract. Motivation: Networks underlie the generation and interpretation of many biological datasets: gene networks shed light on the regulatory structure of the genome, and cell networks can capture structure of the tumor micro-environment. However, most methods that learn such networks make the faulty 'independence assumption'; to learn the gene network, they assume that no cell network exists. 'Multi-axis' methods, which do not make this assumption, fail to scale beyond a few thousand cells or genes. This limits their applicability to only the smallest datasets.

Results: We develop a multi-axis method capable of processing million-cell datasets within minutes. This was previously impossible, and unlocks the use of such methods on modern scRNA-seq datasets, as well as more complex datasets. We show that our method yields novel biological insights from real single-cell data, and compares favorably to the existing hdWGCNA methodology. In particular, it identifies long non-coding RNA genes that potentially have a regulatory or functional role in neuronal development.

Availability and implementation: Our methodology is available as a Python package GmGM on PyPI (https://pypi.org/project/GmGM/0.5.3/). The code for all experiments performed in this paper is available on GitHub (https://github.com/BaileyAndrew/GmGM-Bioinformatics).

Contact: sceba@leeds.ac.uk

Supplementary information: Our proofs, and some additional experiments, are available in the supplementary material.

6.1 Introduction

Learning a network ('graph') that describes a dateset can improve understanding of the systems that generated it. A common example is learning a gene association network from a single-cell RNA-sequencing (scRNA-seq) dataset. These datasets typically come in the form of a cells $\times$ genes matrix, with the individual matrix entries of gene expression values; learning a gene network from this type of data gives insight into the regulatory pathways governing cell behavior. There are many methods to learn networks, such as the graphical lasso (Friedman et al., 2008), WGCNA and its single-celled variant hdWGCNA (Langfelder & Horvath, 2008; Morabito et al., 2023), and neighborhood selection (Meinshausen & Bühlmann, 2006).

However these methods all make the 'independence assumption'; to learn the gene network, they must assume that the cells do not interact. This is clearly false; cells come from a highly interactive micro-environment, and come in a variety of cell types - all of this information is useful for the construction of gene networks, and should not be thrown away. In this paper, we propose a scalable method to learn gene networks while still capturing cellular heterogeneity; as a byproduct, our method also produces a cell network that can be used for cell type inference. By learning both networks simultaneously, one can expect to get better results than learning them sequentially - not only does our method not make the independence assumption, but it can share information between the networks when learning them, leading to better estimates of both.

Our method is a type of 'multi-axis' method (which are often called 'Kronecker-separable' methods), named so because they consider the dependencies within all *axes* of a dataset; in contrast, methods that make the independence assumption are 'single-axis'. In general, for any matrix-variate dataset there are two potential *kinds of dependencies* one must consider: sample ('row', 'cell') and feature ('column', 'gene') dependencies. This concept, and our method, can be extended to more complex ('tensor-variate') datasets, which occasionally show up in bioinformatics (Example 21).

Example 19. *scRNA-seq datasets have two axes (cells, genes).*

Example 20. *scATAC-seq datasets have two axes (cells, peaks).*

Example 21. *A longitudinal bulk RNA study of multiple patients has three axes (timepoints, patients, genes).*

Also in scope are datasets with 'shared axes'. A shared axis is an axis that appears in two or more matrices at a time. The goal is to learn a network describing each axis, without making an independence assumption on any.

Example 22. *Paired scRNA-seq/scATAC-seq datasets have three axes (genes, peaks, and a cells axis shared between the two modalities).*

Example 23. *A 5 patient scRNA-seq dataset has six axes (one for the cells of each patient, and a gene axis shared between them all).*

The standard independence assumption is that there are no row dependencies (i.e. there are no dependencies between cells in the scRNA-seq case), which is tractable but inaccurate. One could instead allow 'full dependence', i.e. every matrix element (i_1, j_1) could potentially be dependent on any other matrix element (i_2, j_2). This would allow very granular statements to be made about the data, such as 'gene A in cell X affects gene B in cell Y'. Sadly, inference of such a network is statistically intractable: a $d \times d$ matrix would have $O(d^4)$ potential dependencies, and only one 'sample' (the matrix itself) from which to learn such dependencies.

To strike a balance between model accuracy and model tractability, the independence assumption is replaced with a weaker, more realistic assumption about how the axes may interact. In this paper, we make the **Cartesian product assumption**, which is easiest to describe in terms of a concrete scRNA-seq example: 'a gene can only be (conditionally) dependent on the expression of *other genes within the same cell*, or the *same gene in other cells*'. This assumption is weaker than the independence assumption, yet like the independence assumption only a more tractable $O(d^2)$ possible dependencies need be considered.

The networks considered in this paper are *conditional dependence* networks; two genes are conditionally dependent, given the rest of the genes in the dataset, if they are statistically dependent on one another after removing the effect of the other genes. Conditional dependence captures the concept

of 'direct correlation'; if A and B are only correlated indirectly due to their mutual correlation with another gene C, then they would be conditionally independent. Thus, conditional dependency networks tend to be sparser and more meaningful than co-expression networks. For Gaussian data, the inverse covariance matrix $\mathbf{\Psi}$ encodes the conditional dependency graph as follows:

$$\mathbf{\Psi}_{ij} = 0 \iff \text{gene } i \text{ is conditionally independent from gene } j$$

This convenient relationship allows our assumptions about the data to be expressed as a probabilistic model, where $\mathbf{\Psi}^{(cells)}, \mathbf{\Psi}^{(genes)}$ are the cell and gene conditional dependency matrix and d_1, d_2 are the numbers of cells and genes, respectively. In doing so, our model is amenable to maximum likelihood estimation (MLE). Letting $\mathbf{X}$ be our dataset, $\mathbf{I}$ and $\mathbf{0}$ be the identity matrix and zero matrix, vec be the operation that flattens matrices into vectors, and $\otimes$ be the 'Kronecker product' operation, traditional single-axis models and our model can be expressed as follows (with $\mathcal{N}(\boldsymbol{\mu}, \mathbf{\Sigma})$ being the normal distribution with mean $\boldsymbol{\mu}$ and covariance $\mathbf{\Sigma}$):

$$\text{vec}[\mathbf{X}] \sim \mathcal{N}\left(\mathbf{0}_{d_1 d_2}, \left(\mathbf{\Psi}^{(genes)}_{d_2 \times d_2} \otimes \mathbf{I}_{d_1 \times d_1} + \mathbf{I}_{d_2 \times d_2} \otimes \mathbf{\Psi}^{(cells)}_{d_1 \times d_1}\right)^{-1}\right) \quad \text{(Cartesian product [our method])}$$

$$\text{vec}[\mathbf{X}] \sim \mathcal{N}\left(\mathbf{0}_{d_1 d_2}, \left(\mathbf{\Psi}^{(genes)}_{d_2 \times d_2} \otimes \mathbf{I}_{d_1 \times d_1}\right)^{-1}\right) \quad \text{(Independence of cells [glasso, hdWGCNA, etc.])}$$

The operation $\mathbf{F} \otimes \mathbf{I} + \mathbf{I} \otimes \mathbf{G}$ is common enough that it is known as the 'Kronecker sum' $\mathbf{F} \oplus \mathbf{G}$. For two graphs $\mathfrak{F}, \mathfrak{G}$ encoded as matrices $\mathbf{F}, \mathbf{G}$, the Kronecker sum $\mathbf{F} \oplus \mathbf{G}$ encodes the Cartesian product of $\mathfrak{F}$ and $\mathfrak{G}$.

A Gaussian assumption would not be correct for most biological datasets, particularly those comprising of binary or count data. However, this assumption can be replaced by the much weaker 'Gaussian copula' assumption, which allows the data to follow any distribution as long as the genes 'interact Gaussianly'; in particular, if genes were to follow a negative binomial or Poisson distribution, they can still fit the Gaussian copula assumption. This is discussed further in Section 6.2.2.

This paper is not the first to consider the Cartesian product assumption; there are several methods to learn such graphs. The most scalable include TeraLasso (Greenewald et al., 2019), which was the first to generalize to tensor-variate data, and GmGM (Andrew et al., 2024), which further generalized the scenario to the 'shared axis' case[1]. However, TeraLasso is only scalable to hundreds of cells, and GmGM to low thousands. Modern scRNA-seq datasets have hundreds of thousands of cells. In fact, there are 11 Human Cell Atlas datasets with more than a million cells (Regev et al., 2017). Extrapolating their runtimes to a million-cell dataset, TeraLasso would take 15,000 years to run and GmGM would take 200. Both would require at least 16 terabytes of memory. Even with access to a high-performance-computing (HPC) cluster, the usability of these algorithms on such datasets is dubious.

Due to these scalability constraints, multi-axis models cannot be applied to modern scRNA-seq datasets. In this paper, we make significant alterations to the GmGM algorithm to improve its scalability, resulting in an algorithm that can run on million-cell datasets in a matter of minutes. Our contributions unlock the use of multi-axis models on modern datasets.

6.2 Materials and methods

6.2.1 Notation

Because our model is built to work on a wide variety of datasets (including those with shared axes and more than two axes), the notation can be quite dense; in particular, many variables of the form d_y^x are defined below. For most users, only the matrix-variate case (which includes scRNA-seq) will be of interest, for which the notification can be drastically simplified. See Section 6.2.1 for such a simplification. Furthermore, Figure 6.1 gives a graphical overview of the most complicated bit of the notation.

[1] Other methods that work with shared axes in multi-modal datasets do exist, but they all make the independence assumption on one of the axes; for example, the group graphical lasso and fused graphical lasso (Cai et al., 2016; Danaher et al., 2014), and Gaussian chain graphical models (McCarter & Kim, 2014).

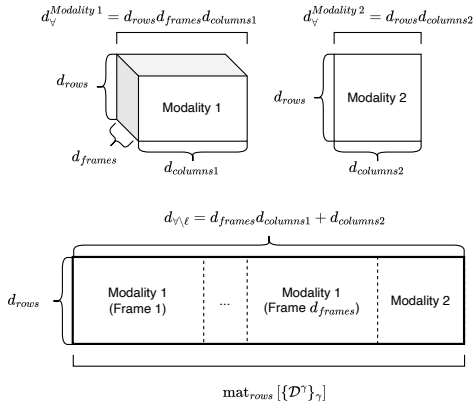


Figure 6.1: An example two-modality dataset with a few relevant properties highlighted. Also contains a graphical definition of $\mathrm{mat}_\ell\left[\{\mathcal{D}^\gamma\}_\gamma\right]$ (bottom).

Axes are denoted using subscripts (often with the symbol ℓ); for example, $\mathbf{\Psi}_{\text{cells}}$ would be the precision matrix of the cells axis. Modalities will be represented with superscripts (often with the symbol γ); in a multi-patient scRNA-seq study, $\mathbf{D}^{\text{patient1}}$ would represent the cells $\times$ genes matrix of the first patient. $\ell \in \gamma$ represents the statement that an axis is recorded in a modality; in a paired scRNA/ATAC-seq dataset, "genes $\in$ scRNA-seq modality" but "genes $\notin$ scATAC-seq modality".

Let d_ℓ be the size of axis ℓ, $d_\forall^\gamma = \prod_{\ell\in\gamma} d_\ell$ be the size of modality γ, $d_{\setminus\ell}^\gamma = \frac{d_\forall^\gamma}{d_\ell}$ be the number of samples from ℓ's perspective in γ, and $d_{\setminus\ell}^\forall = \sum_{\gamma|\ell\in\gamma} d_{\setminus\ell}^\gamma$ be the number of samples from ℓ's perspective overall. r_ℓ and analogous variables are defined to represent the number of principal components chosen for axis ℓ.

$\det^\dagger$ represents the *pseudo-determinant* and vec represents vectorization. $\mathrm{mat}_\ell\left[\mathcal{X}^\gamma\right]$ represents the 'matricization' of a tensor $\mathcal{X}^\gamma$, which reshapes the data into a $d_\ell \times d_{\setminus\ell}^\gamma$ matrix. Matricization of a dataset, $\mathrm{mat}_\ell\left[\{\mathcal{X}^\gamma\}_\gamma\right]$, reshapes the data into a $d_\ell \times d_{\setminus\ell}^\forall$ matrix. See Figure 6.1 for a graphical example of these definitions. $\mathbf{S}_\ell = \mathrm{mat}_\ell\left[\{\mathcal{X}^\gamma\}_\gamma\right]\mathrm{mat}_\ell\left[\{\mathcal{X}^\gamma\}_\gamma\right]^T$ is the covariance matrix for axis ℓ. Its top r_ℓ eigenvectors and eigenvalues are $\mathbf{V}_\ell^{(r_\ell)}$ and $\mathbf{E}_\ell^{(r_\ell)}$, respectively.

The final definition needed is that of the stridewise-blockwise trace $\mathrm{tr}_b^a\left[\mathbf{X}\right]$; it arises in the gradient of terms involving Kronecker products. Letting $\mathbf{J}^{(ij)}$ is a matrix of zeros except for a single 1 in position (i,j), we have:

$$\mathrm{tr}_b^a\left[\mathbf{X}\right]_{ij} = \mathrm{tr}\left[\mathbf{X}\left(\mathbf{I}_{a\times a} \otimes \mathbf{J}^{(ij)} \otimes \mathbf{I}_{b\times b}\right)\right]$$

Simplified notation

If one is only interested in matrix-variate datasets, the notation can be simplified. For any such dataset, there is only one modality, so all superscripts γ can be ignored. Furthermore, there are only two axes (of sizes d_1, d_2), and $d_1 = d_{\setminus 2} = d_{\setminus 2}^\forall$ (with analogous relationships for d_2). $d_\forall = d_1 d_2$, and for a dataset $\mathbf{X}$ it holds that $\mathrm{mat}_1\left[\mathbf{X}\right] = \mathbf{X}^T$ and $\mathrm{mat}_2\left[\mathbf{X}\right] = \mathbf{X}$. Any kind of product over $\ell \in \gamma$ is just a product over the two axes; $\ell \in \{1,2\}$.

6.2.2 Assumptions

The five assumptions of this work are listed below, citing where they have been used for multi-axis modelling before.

Assumption 7. *The data's dependencies are Kronecker-sum structured. First used by Kalaitzis et al. (2013).*

Assumption 8. *The data has a Gaussian copula, also called the 'nonparanormality assumption'. Used by Li et al. (2022). This assumption allows the data to have any marginal distribution (such as negative binomial), as long as the genes interact 'Gaussian-ly'.*

Assumption 9. *Conditional dependency graphs with d vertices have $O(d)$ edges ,i.e. are 'linearly sparse'. Follows from Assumption A4 of Greenewald et al. (2019).*

Assumption 10. *The largest nonzero eigenvalue of $\mathbf{\Psi}_\ell$ is at least ϵ_ℓ, for some small ϵ_ℓ. Follows from Assumption A2 of Greenewald et al. (2019).*

Assumption 11 (New)**.** *The input data and true graphs are well-approximated by low-rank matrices. This assumption is common throughout data science (see Appendix E.1) but has not been applied to improve performance for multi-axis models.*

Our method inherits the nonparanormality assumption from prior work. Intuitively, this corresponds to a multivariate distribution with arbitrary marginals but whose variables interact with each other 'like Gaussian variables'. This is done by only considering rank statistics of gene expression, rather than the exact expression values. See Liu et al. (2012) for more details. To our knowledge, there has been no Kronecker-structured work that does not make this or a stronger assumption. If there were, it would likely lack the crucial properties of the Gaussian copula that make it efficient to scale (namely the relationship between $\mathbf{\Psi}$ and the graph of conditional dependencies) - although it is certainly a worthwhile avenue for future work to explore.

The main bottleneck to scalability in the prior version of GmGM was that its memory complexity was $O(\sum_\ell d_\ell^2)$. Without making any additional assumptions, this memory complexity is optimal; a set of K fully dense $d_\ell \times d_\ell$ precision matrices require exactly $\sum_\ell \frac{d_\ell^2+d_\ell}{2}$ values to specify. However, when one is working with conditional dependencies, the output graphs will typically be sparse. The 'linearly sparse' assumption makes it at least theoretically possible to achieve $O(\sum d_\ell)$ memory complexity. Real world networks are often linearly sparse. Consider the case of a social network; no matter how many people p there are in the world, there is (sadly) a constant upper bound on the possible number of friends someone can make during their life. This yields a graph with $O(p)$ edges.

While linear sparsity makes a linear memory cost possible, all prior work relies on eigendecompositions. This entails the use of the matrix of eigenvectors, one for each axis, requiring $\sum_\ell d_\ell^2$ elements. To achieve scalable memory usage, this must be avoided. Thankfully, if the dataset is well-approximated by a low-rank matrix, then only the first few eigenvectors are actually meaningful! The full eigendecomposition is not needed. This assumption is easily verifiable on real data; PCA is a standard part of many data analysis pipelines, with there being many techniques to estimate the number of principal components (i.e. eigenvectors) required to capture most of the variance in the data.

6.2.3 The model

Our model is similar in structure to prior work; however, to take advantage of Assumption 11 a '*singular*' probability distribution is used; see Appendix E.2 for an explanation and derivation. Letting r_ℓ be the rank chosen for each axis, and $r_\forall^\gamma$ be the product of ranks of all axes occurring in modality γ, our model can be expressed as follows:

$$\begin{aligned}\mathrm{pdf}_{\mathrm{normal}}(\mathcal{X}^\gamma) &= \frac{\sqrt{\det{}^\dagger\left(\bigoplus_{\ell\in\gamma}\mathbf{\Psi}_\ell\right)}}{(2\pi)^{\frac{r_\forall^\gamma}{2}}} e^{-\frac{1}{2}\mathrm{vec}[\mathcal{X}^\gamma]^T\left(\bigoplus_{\ell\in\gamma}\mathbf{\Psi}_\ell\right)\mathrm{vec}[\mathcal{X}^\gamma]} \\ &= \frac{\sqrt{\det{}^\dagger\left(\bigoplus_{\ell\in\gamma}\mathbf{\Psi}_\ell\right)}}{(2\pi)^{\frac{r_\forall^\gamma}{2}}} e^{-\frac{1}{2}\sum_{\ell\in\gamma}\mathrm{tr}\left[\mathbf{S}_\ell^\gamma\mathbf{\Psi}_\ell\right]} \end{aligned} \tag{6.1}$$

When given a multi-modal dataset, the model is:

$$\mathrm{pdf}_{\mathrm{normal}}\left(\{\mathcal{X}^\gamma\}\right) = \prod_\gamma \frac{\sqrt{\det{}^\dagger\left(\bigoplus_{\ell\in\gamma}\mathbf{\Psi}_\ell\right)}}{(2\pi)^{\frac{r^\gamma_\forall}{2}}} e^{-\frac{1}{2}\sum_{\ell\in\gamma}\mathrm{tr}\left[\mathbf{S}^\gamma_\ell\mathbf{\Psi}_\ell\right]}$$

Above, the data is assumed to be Gaussian ($\mathbf{x}\sim\mathcal{N}(\cdot,\cdot)$). To achieve the weaker 'Gaussian copula' assumption (Assumption 8), which states that the dataset is a *monotonic transformation* f of a Gaussian dataset ($f^{-1}(\mathbf{x})\sim\mathcal{N}(\cdot,\cdot)$), we follow Li et al. (2022) in using the Nonparanormal Skeptic (Liu et al., 2012) to first compute the sufficient statistics $\{\mathbf{S}_\ell\}$, before applying our model.

6.2.4 Theory

The aim of this paper is to find the maximum likelihood estimator (MLE) of Equation 6.1 efficiently. A major bottleneck in prior work was the need to re-compute eigenvectors every iteration. Appendix E.7 contains the proof for the following theorem:

Theorem 10. *Let the rank-r_ℓ partial eigendecomposition of $\mathbf{S}_\ell$ be $\mathbf{V}^{(r_\ell)}_\ell\mathbf{E}^{(r_\ell)}_\ell\mathbf{V}^{(r_\ell),T}_\ell$. Then $\mathbf{V}^{(r_\ell)}_\ell$ are also eigenvectors of the maximum likelihood estimate $\hat{\mathbf{\Psi}}_\ell$.*

This reduces the problem to finding the eigenvalues, which lack an exact solution but can be solved for iteratively due to the convexity of the problem. The next theorem shows that this is a well-founded problem:

Theorem 11. *There exists a unique MLE that estimates the precision matrix of the singular Kronecker-sum-structured normal distribution, provided $r_\ell\leq\mathrm{rank}\left[\mathbf{S}_\ell\right]$.*

Note that $r_\ell\leq d^\forall_{\backslash\ell}$ suffices.

Kronecker-structured models often suffer identifiability issues; in Appendix E.3 it is shown that there exists an identifiable parameterization. In Appendix E.9, a hypothesis testing procedure is derived to allow one to deduce the statistically significant edges.

Theorem 12. *Under the null hypothesis $\bigoplus_{\ell\in\gamma}\mathbf{\Psi}_\ell=\frac{1}{(\sigma^\gamma)^2}\mathbf{I}_{d^\gamma_\forall}$, the following distribution holds for each $\psi_{\ell_{ij}}$ independently:*

$$\sqrt{\frac{\sum_{\gamma|\ell\in\gamma}d^\gamma_\forall\left(\sigma^\gamma\right)^4}{d_\ell}}\frac{\psi_{\ell_{ij}}}{2}\dot{\sim}\mathcal{N}\left(0,1\right)$$

In practice, we have found this test to be quite weak, so thresholding may be preferred, especially if the graph is used as a preprocessing step (such as for clustering) rather than an end in and of itself.

6.3 Algorithm

Our maximum likelihood estimation algorithm (Appendix E.4, Algorithm 3) can be described as: calculate each $\mathbf{S}_\ell$, partially eigendecompose them, perform gradient descent on the eigenvalues[2], eigenrecompose to get $\mathbf{\Psi}_\ell$, and then threshold to get the final solution. However, creation of $\mathbf{S}_\ell$ requires $O(d_\ell^2)$ space - the goal is to only require $O(r_\ell d_\ell)$ space. This can be avoided by noting that:

$$\begin{aligned}
\mathbf{S}_\ell &= \sum_\gamma \mathbf{S}^\gamma_\ell \\
&= \sum_\gamma \mathrm{mat}_\ell\left[\mathcal{D}^\gamma\right]\mathrm{mat}_\ell\left[\mathcal{D}^\gamma\right]^T \\
&= \begin{bmatrix}\mathrm{mat}_\ell\left[\mathcal{D}^1\right] & \ldots & \mathrm{mat}_\ell\left[\mathcal{D}^\Gamma\right]\end{bmatrix}\begin{bmatrix}\mathrm{mat}_\ell\left[\mathcal{D}^1\right]^T \\ \ldots \\ \mathrm{mat}_\ell\left[\mathcal{D}^\Gamma\right]^T\end{bmatrix} \\
&\triangleq \mathrm{mat}_\ell\left[\{\mathcal{D}^\gamma\}_\gamma\right]\mathrm{mat}_\ell\left[\{\mathcal{D}^\gamma\}_\gamma\right]^T
\end{aligned}$$

[2]The formula for the gradient of the eigenvalues is derived in Appendix E.7.

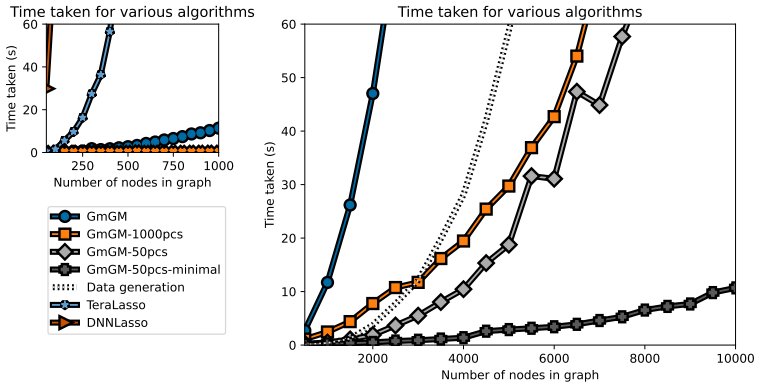


Figure 6.2: The running time of various multi-axis algorithms on synthetic data as the number of nodes in the graph increases. The synthetic datasets were generated from a Kronecker-sum-structured normal distribution The top-left focuses in on the graphs with less than 2000 nodes to be able to show TeraLasso (Greenewald et al., 2019) and DNNLasso (Lin & Zhang, 2024). 'GmGM' refers to the old GmGM (Andrew et al., 2024), whereas GmGM-Xpcs refers to our modifications, using only X principal components. GmGM-50pcs-minimal corresponds to the case in which the number of edges in the graph is proportional to the number of nodes (Assumption 9); for the other models, the full graph was kept.

$\mathbf{S}_\ell$ can be thought of as the covariance matrix for a $d_\ell \times \sum_\gamma d_\ell^\gamma$ matrix "$\text{mat}_\ell \left[\{\mathcal{D}^\gamma\}_\gamma\right]$", and hence its eigenvectors are precisely the singular vectors of $\text{mat}_\ell \left[\{\mathcal{D}^\gamma\}_\gamma\right]$. This allows avoiding ever constructing $\mathbf{S}_\ell$.

Such an approach has one major drawback; the Nonparanormal Skeptic works by supplying a rank-based $\mathbf{S}_\ell$ that, intuitively, represents the correlation matrix 'as if the data had come from a Gaussian'. To both avoid quadratic memory usage *and* weaken the normality assumption, it requires a modification. COCA (Han & Liu, 2014) is used to calculate nonparanormal eigenvectors; it works by replacing $\text{mat}_\ell \left[\{\mathcal{D}^\gamma\}_\gamma\right]$ with a matrix of ranks, mapping these ranks to a normal distribution, and then finding the singular vectors of this rank matrix.

The final practical issue to surmount is that of preserving input sparsity. $\text{mat}_\ell \left[\{\mathcal{D}^\gamma\}_\gamma\right]$ has $d_\forall$ elements, but in many contexts, such as transcriptomics, the dataset will be highly sparse; the majority of the input is full of zeros. Ideally, the matrix would never be 'densified'. Using COCA, however, requires computing the matrix of ranks, which will be fully dense. If unaddressed, this would enforce an $\Omega(d_\forall)$ minimum memory requirement. **Solving this is necessary for scalability**; as will be seen in Section 6.5.1, our algorithm can handle very large sparse datasets; if densified, they would be in excess of 30 GB of memory. In Appendix E.10, we show how to solve this using a rank-one update technique.

6.4 Implementation

The algorithm is implemented in Python, using Numpy, Numba, SciPy, and Dask (Harris et al., 2020; Lam et al., 2015; Rocklin, 2015; Virtanen et al., 2020). The experiments were run on a 13-inch MacBook Pro with an M1 chip and 8GB RAM. 23GB of hard disk storage were available when running the experiments; the million cell experiment (Section 6.5.1) could not fit in 8GB of RAM, and thus much of the free hard disk space was used as swap memory.

6.5 Results

The scalability of our method is demonstrated in Section 6.5.1. In Section 6.5.2, an exploration of the role of long non-coding RNA (lncRNA) in neural progenitor cells is performed. In biology, it can be hard to know the ground truth; in Appendix E.11, there is an additional comparison on a non-biological real-world toy dataset with known ground truth to validate our methodology.

6.5.1 Scalability

In this section, the scalability of our method is demonstrated, comparing it to two other multi-axis methods on in-distribution synthetic data[3]. The synthetic datasets were generated from a Kronecker-sum-structured normal distribution, and the ground truth networks to be learned were generated from a Barabasi-Albert distribution. The Barabasi-Albert distribution was chosen as it is a power law distribution; many real world networks approximately follow a power law. Figure 6.2 shows how much more scalable our modification makes GmGM, with it being the only method that can feasibly run on datasets of several thousand cells.

To emphasize this point, our method was run on a 1,248,980 cell PBMC dataset by Yazar et al. (2022), keeping the top 50 principal components (∼50% of the variance) and only the statistically significant edges (5890 of them). Our method ran in less than two minutes on a personal computer (see Appendix E.11 for more details and a sanity-checking of the results). It should be noted that prior multi-axis work is fundamentally not scalable to problems of this size, *even with access to high performance computing clusters*.

On a theoretical level, the scalability of our method is achievable in large part due to its lower computational complexity. For a dataset of $O(d)$ genes and cells, prior work required an $O(d^3)$ runtime and $O(d^2)$ space use; in contrast, ours requires $O(d^2)$ time and $O(d)$ space. A derivation of the computational complexity, as well as its full statement for tensor-variate and shared-axis datasets, is given in Appendix E.5.

6.5.2 Application to single cell neuronal differentiation data

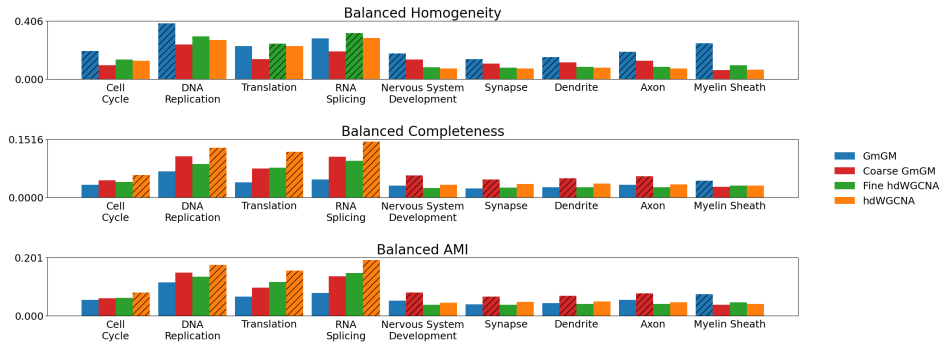


Figure 6.3: A comparison of the homogeneity and completeness of clusterings our method (GmGM) and hdWGCNA across various metrics over different sets of GO terms (see Appendix E.14 for GO ids). Homogeneity measures the tendency for modules to contain only genes of a given set of GO terms, whereas completeness measures the tendency for a set of GO terms to all be contained in a single module. AMI stands for adjusted mutual information. 'Coarse GmGM' refers to a clustering on GmGM with 7 modules (the same as hdWGCNA), and 'fine hdWGCNA' refers to a clustering on hdWGCNA with 119 modules (the same as GmGM), most of which were singletons. These metrics were adjusted for chance following Romano et al. (2016).

The previous section proved the scalability of our method; in this section, we will demonstrate that our method leads to real biological insights, benchmarking against the single-axis hdWGCNA (Morabito et al., 2023) methodology, which identifies gene co-expression networks. Method comparison for biological networks is difficult, as there is no ground truth; however, we will see that GmGM leads to networks with different structure than hdWGCNA, and that these differences result in new biological insights. In particular, GmGM yields gene modules that are more specific to the relevant biology in our dataset. Both methods are applied to a scRNA-seq dataset of induced pluripotent stem cells (iPSCs), measured at 4 different time points as they develop into neural progenitor cells; the dataset consists of 6876 cells and 29,324 genes. See Appendix E.12 for details on how the data was collected.

To generate the GmGM network, the top 100 principal components were kept and the nonparanormal skeptic was used. The top 5 strongest edges per gene/cell were retained. To generate gene clusters ('modules') from the learned gene network, the Leiden (Traag et al., 2019) algorithm was used with a

[3]See Appendix E.11 for precision/recall comparisons on said data.

'resolution' parameter of 3 (chosen to give at least 100 modules). For the hdWGCNA network, 'soft power' of 4 was determined to be optimal by the hdWGCNA package's TestSoftPower's method. We followed hdWGCNA's guidelines to ensure that the network and clustering obtained were the best possible, as defined by the authors of hdWGCNA. The hdWGCNA package's recommended clustering was used for the modules (only 8 modules); a finer clustering was attempted with Leiden, although this resulted in many singletons and very small clusters. GmGM learns conditional dependency and hdWGCNA learns co-expression; these types of networks have fundamentally different topologies which is the likely cause of this difference.

The networks were evaluated by analyzing clusterings on them. We identified 119 gene modules on the GmGM network, labeled m0, m1, m2, and so on. A short description of each is given in Appendix E.15 (Table E.9). The modules range from having 37 genes (m118) to 600 genes (m2), with the exception of m1 with 1756 genes and m0 with 9047. hdWGCNA identified 7 gene modules (ranging from 241 to 6399 genes each) - an eighth 'unnassigned' module contained the 14,405 genes that were not assigned to any module by hdWGCNA. A short description of each is given in Appendix E.15 (Table E.8). To limit the potentially confounding effects of cluster quantity on our analysis, an additional clustering was produced for each method. These were an 8 module clustering on the GmGM network ('Coarse GmGM') and a 119 module clustering on the hdWGCNA network ('Fine hdWGCNA').

The networks and clusterings considered had very different properties. The hdWGCNA network contained many singletons (the genes in the unassigned module) whereas GmGM contained none. The GmGM network also had a higher transitivity (the tendency for connected genes to have neighbors in common) than hdWGCNA (0.055 vs 0.002). The GmGM, coarse GmGM, fine hdWGCNA, and hdWGCNA clusterings achieved modularities (the tendency for a clustering to have more within-cluster than between-cluster edges) of 0.569, 0.600, 0.347, and 0.220, respectively.

To compare methods in terms of their ability to extract biologically relevant information, we evaluated how well their clusterings aligned with a set of curated GO terms. The GO terms were picked to either represent a fundamental biological process (such as the cell cycle or DNA replication) or the development of a key component of neurons (axons, dendrites, synapses, the myelin sheath, and more broadly nervous system development in general). Appendix E.14 (Table E.7) contains the GO terms chosen as well as the rationale behind the choice.

The GO term comparison is given in Figure 6.3. It shows that our method's clusters are better aligned with neuron-specific GO terms, while hdWGCNA better captured non-specific processes, such as the cell cycle. This is unsurprising; when we qualitatively inspect hdWGCNA modules (Appendix E.15's Table E.8), only two of the eight modules are associated with GO terms specific to neurons (the 'blue' and 'red' clusters); GmGM, in contrast, had many (Appendix E.15's Table E.9).

In general, a qualitative analysis shows that every module of hdWGCNA (except the 'unassigned' module) clearly relates to some biological process, although typically a broad process such as RNA binding. In contrast, only 39 of GmGM's 119 modules show something of immediate biological interest. However, those that do tend to be more specific than hdWGCNA; GmGM can capture biological processes that are more relevant to the domain at hand (in this case, neural development). For a module-by-module breakdown, see the tables of Appendix E.15.

Another benefit of GmGM is its ability to simultaneously learn a cell network; this is not available to hdWGCNA, which only learns the gene graph. This cell network was clustered (Leiden, resolution=1) and each cluster was associated with a cell type based on its statistical association with a set of marker genes (Appendix E.13, Table E.6). The resulting clustering is shown in Figure 6.4.

There was often an overlap between cell clusters and the expressions of genes in the gene modules; for example, the cortical neuron cells significantly express the modules m3, m13, and m27. The relation of these clusters to cortical neurons is confirmed by their associated GO terms (including 'forebrain development', 'synapse organization', and 'dendrite') and their dominant genes (the m27 sub-network is dominated by *TBR1*, which is a key transcription factor for early cortical neurogenesis). The overlap between GmGM's gene and cell clusters are given by Table 6.1.

We conclude this section with a brief exploration of selected modules of the GmGM network. The m90 module was especially interesting, as it had GO terms indicating a role in neurogenesis and transcription factors. The most central genes for this module were *NFIA* and *NFIB* (known to be crucial for nervous system development) and the unnamed lncRNA *ENSG00000243620*. *ENSG00000243620* and another unnamed lncRNA (*ENSG00000286757*) were found in the GmGM network to be strongly connected to the *ZIC1*, *ZIC2*, and *ZIC4* transcription factors (also known to be important in neural

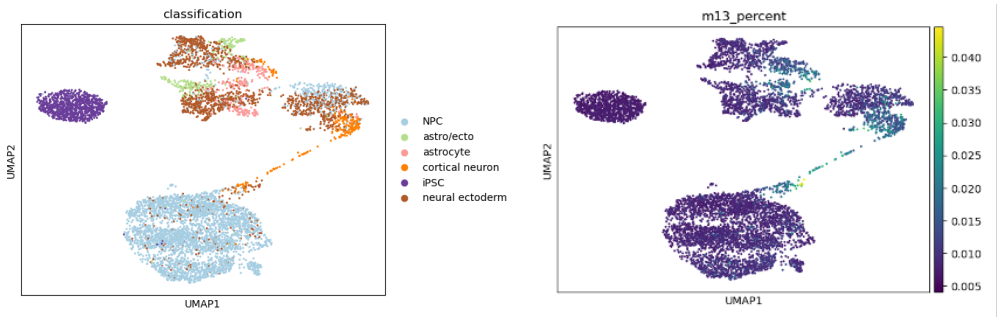


Figure 6.4: (left) The cell type classification produced by GmGM using the marker genes from Appendix E.13. NPC stands for 'neural progenitor cell', and 'astro/ecto' denotes cells which could not be distinguished between astrocyte and neural ectoderm with the markers given. (right) The average gene expression of the genes in GmGM module m13; it tends to express itself only in and around cells that are identified as cortical neurons. (both) UMAP plots have been used for visualization; the overexpression of m13 in the suspected cortical neuron cells has also been validated statistically.

Cell Type	Clusters (Overexpressed gene modules in parentheses)
iPSC	0 (m1), 5 (m1)
Neural ectoderm	1 (m90,m9); 3 (m6,m90,m118,m30); 6 (m14,m11,m90,m30); 15 (m13)
NPC	2 (m25); 4 (m11,m19,m51,m25,m50,m62); 7 (m11,m50,m91); 8 (m11); 9 (m10); 10 (m14,m51); 12 (m11,m91,m50,m25,m51,m62); 14 (m110,m91,m11,m50,m20,m51,m10,m62,m25,m12); 16 (m51,m8); 17 (m16,m90,m62,m14); 19 (m11); 20 (m50); 21 (m11,m91,m62,m25,m51,m50); 23 (m14)
Astrocyte	18 (m90,m3,m13); 22 (m9)
Cortical Neuron	11 (m3,m13,m27)
Astrocyte or neural ectoderm	13 (m90,m2,m6)

Table 6.1: The overexpressed modules for each GmGM cell cluster, using the marker genes from Appendix E.13's Table E.6 to identify cell types. Each cluster had many statistically significantly overexpressed modules; here, we are only showing the most extremely overexpressed modules.

development). They lie on chromosomes 3 and 13, respectively; *ENSG00000243620* is only 50 kilobases from *ZIC1* and *ZIC4*, whereas *ENSG00000286757* is only 20 kilobases from *ZIC2* (and *ZIC5*, not in this module). The closeness on the genome may indicate a regulatory relationship between *ZIC* genes and the unnamed lncRNAs.

To see if this finding could be repeated, the two other neurogenesis-and-transcription-factor-associated modules were also investigated (m13 and m8). m13 associated with the genes *EBF2*, *EBF3*, and *NEUROD4* (all of which are relevant to neural development). The strongest connections in this module often involve the *ELAVL1*, *ELAVL3*, and *ELAVL4* genes, one of which is between *ELAVL2* and an unnamed lncRNA '*ENSG00000283982*' 17 kilobases away. Finally, m8 is not particularly dominated by any one set of genes, but one of the strongest connections in the module is between *SHISA9* and an unnamed lncRNA '*ENSG00000262801*' 5000 base pairs away.

The hdWGCNA network was checked to see if it also identified relationships between the same lncRNAs. The lncRNA results show that the m8 and m13 lncRNAs of interest are unimportant in the hdWGCNA network: each only connected to 5 other genes, on other chromosomes. In contrast, the m90 lncRNAs of interest are both highly central to hdWGCNA's 'brown' module, being in the top 4 most central (behind *DMD* and *MIR100HG*), and each having more than 100 connections to other genes, including the *ZIC* genes.

6.6 Discussion

In this paper, we have developed a novel and general multi-axis methodology able to produce results for large-scale scRNA-seq datasets, and proven this by applying it to a million-cell dataset. Furthermore, we show that our methodology is able to extract novel biological insights by finding promising connections between transcription factors and some poorly-studied lncRNA genes.

We compared our results to hdWGCNA, demonstrating that our method is able to create networks conducive to finer clustering, albeit with a higher proportion of clusters for which no immediate biological meaning is apparent. The results from hdWGCNA lend credence to a potential ZIC-lncRNA relationship found by our method which warrants further study.

Finally, by simultaneously learning both the gene and cell graph, one gains multiple means of learning new biological insights and validating results. This is a benefit our methodology has over more traditional single-axis methodologies like hdWGCNA, which only learn gene graphs.

6.7 Acknowledgements

Bailey Andrew is supported by the UKRI Engineering and Physical Sciences Research Council (EPSRC) [EP/S024336/1]. David Westhead is supported in part by the National Institute for Health and Care Research (NIHR) Leeds Biomedical Research Centre (BRC) (NIHR203331). The views expressed are those of the author(s) and not necessarily those of the NHS, the NIHR or the Department of Health and Social Care. James Poulter is supported by a UKRI Future Leaders Fellowship (MR/Y034325/1).

6.8 Author Contributions

Bailey Andrew (Conceptualization, Formal analysis, Methodology, Software, Writing - original draft), Erica Harris (Data curation, Resources, Investigation, Writing - original draft), James Poulter (Conceptualization, Supervision, Writing - review & editing), David R. Westhead (Conceptualization, Supervision, Writing - review & editing), and Luisa Cutillo (Conceptualization, Supervision, Writing - review & editing).

Bibliography for Paper 5

Andrew, E. B., Westhead, D., & Cutillo, L. (2024). GmGM: A fast multi-axis Gaussian graphical model [ISSN: 2640-3498]. *Proceedings of The 27th International Conference on Artificial Intelligence and Statistics*, 2053–2061. Retrieved June 26, 2024, from https://proceedings.mlr.press/v238/b-andrew24a.html

Cai, T. T., Li, H., Liu, W., & Xie, J. (2016). Joint Estimation of Multiple High-dimensional Precision Matrices. *Statistica Sinica*, *26*(2), 445–464. https://doi.org/10.5705/ss.2014.256

Danaher, P., Wang, P., & Witten, D. M. (2014). The joint graphical lasso for inverse covariance estimation across multiple classes. *Journal of the Royal Statistical Society. Series B, Statistical methodology*, *76*(2), 373–397. https://doi.org/10.1111/rssb.12033

Friedman, J., Hastie, T., & Tibshirani, R. (2008). Sparse inverse covariance estimation with the graphical lasso. *Biostatistics*, *9*(3), 432–441. https://doi.org/10.1093/biostatistics/kxm045

Greenewald, K., Zhou, S., & Hero, A., III. (2019). Tensor Graphical Lasso (TeraLasso). *Journal of the Royal Statistical Society Series B: Statistical Methodology*, *81*(5), 901–931. https://doi.org/10.1111/rssb.12339

Han, F., & Liu, H. (2014). High Dimensional Semiparametric Scale-Invariant Principal Component Analysis. *IEEE transactions on pattern analysis and machine intelligence*, *36*(10), 2016–2032. https://doi.org/10.1109/TPAMI.2014.2307886

Harris, C. R., Millman, K. J., van der Walt, S. J., Gommers, R., Virtanen, P., Cournapeau, D., Wieser, E., Taylor, J., Berg, S., Smith, N. J., Kern, R., Picus, M., Hoyer, S., van Kerkwijk, M. H., Brett, M., Haldane, A., del Río, J. F., Wiebe, M., Peterson, P., ... Oliphant, T. E. (2020). Array programming with NumPy [Number: 7825 Publisher: Nature Publishing Group]. *Nature*, *585*(7825), 357–362. https://doi.org/10.1038/s41586-020-2649-2

Kalaitzis, A., Lafferty, J., Lawrence, N. D., & Zhou, S. (2013). The Bigraphical Lasso [ISSN: 1938-7228]. *Proceedings of the 30th International Conference on Machine Learning*, 1229–1237. Retrieved February 24, 2023, from https://proceedings.mlr.press/v28/kalaitzis13.html

Lam, S. K., Pitrou, A., & Seibert, S. (2015). Numba: A LLVM-based Python JIT compiler. *Proceedings of the Second Workshop on the LLVM Compiler Infrastructure in HPC*, 1–6. https://doi.org/10.1145/2833157.2833162

Langfelder, P., & Horvath, S. (2008). WGCNA: An R package for weighted correlation network analysis. *BMC Bioinformatics*, *9*(1), 559. https://doi.org/10.1186/1471-2105-9-559

Li, S., López-García, M., Lawrence, N. D., & Cutillo, L. (2022, March). Scalable Bigraphical Lasso: Two-way Sparse Network Inference for Count Data [arXiv:2203.07912 [cs, stat]]. Retrieved February 24, 2023, from http://arxiv.org/abs/2203.07912

Lin, M., & Zhang, Y. (2024). DNNLasso: Scalable Graph Learning for Matrix-Variate Data [ISSN: 2640-3498]. *Proceedings of The 27th International Conference on Artificial Intelligence and Statistics*, 316–324. Retrieved January 15, 2026, from https://proceedings.mlr.press/v238/lin24b.html

Liu, H., Han, F., Yuan, M., Lafferty, J., & Wasserman, L. (2012). The nonparanormal SKEPTIC. *Proceedings of the 29th International Conference on Machine Learning, ICML 2012*, 1415–1422. Retrieved October 9, 2023, from https://pure.johnshopkins.edu/en/publications/the-nonparanormal-skeptic-4

McCarter, C., & Kim, S. (2014). On Sparse Gaussian Chain Graph Models. *Advances in Neural Information Processing Systems*, *27*. Retrieved February 22, 2024, from https://papers.nips.cc/paper_files/paper/2014/hash/81c650caac28cdefce4de5ddc18befa0-Abstract.html

Meinshausen, N., & Bühlmann, P. (2006). High-dimensional graphs and variable selection with the Lasso [Publisher: Institute of Mathematical Statistics]. *The Annals of Statistics*, *34*(3), 1436–1462. https://doi.org/10.1214/009053606000000281

Morabito, S., Reese, F., Rahimzadeh, N., Miyoshi, E., & Swarup, V. (2023). hdWGCNA identifies co-expression networks in high-dimensional transcriptomics data. *Cell Reports Methods*, *3*(6), 100498. https://doi.org/10.1016/j.crmeth.2023.100498

Regev, A., Teichmann, S. A., Lander, E. S., Amit, I., Benoist, C., Birney, E., Bodenmiller, B., Campbell, P., Carninci, P., Clatworthy, M., Clevers, H., Deplancke, B., Dunham, I., Eberwine, J., Eils, R., Enard, W., Farmer, A., Fugger, L., Göttgens, B., ... Human Cell Atlas Meeting Participants. (2017). The Human Cell Atlas. *eLife*, *6*, e27041. https://doi.org/10.7554/eLife.27041

Rocklin, M. (2015). Dask: Parallel Computation with Blocked algorithms and Task Scheduling, 126–132. https://doi.org/10.25080/Majora-7b98e3ed-013

Romano, S., Vinh, N. X., Bailey, J., & Verspoor, K. (2016). Adjusting for Chance Clustering Comparison Measures. *Journal of Machine Learning Research*, *17*(134), 1–32. Retrieved February 17, 2026, from http://jmlr.org/papers/v17/15-627.html

Traag, V. A., Waltman, L., & van Eck, N. J. (2019). From Louvain to Leiden: Guaranteeing well-connected communities [Number: 1 Publisher: Nature Publishing Group]. *Scientific Reports*, *9*(1), 5233. https://doi.org/10.1038/s41598-019-41695-z

Virtanen, P., Gommers, R., Oliphant, T. E., Haberland, M., Reddy, T., Cournapeau, D., Burovski, E., Peterson, P., Weckesser, W., Bright, J., van der Walt, S. J., Brett, M., Wilson, J., Millman, K. J., Mayorov, N., Nelson, A. R. J., Jones, E., Kern, R., Larson, E., ... van Mulbregt, P. (2020). SciPy 1.0: Fundamental algorithms for scientific computing in Python [Number: 3 Publisher: Nature Publishing Group]. *Nature Methods*, *17*(3), 261–272. https://doi.org/10.1038/s41592-019-0686-2

Yazar, S., Alquicira-Hernandez, J., Wing, K., Senabouth, A., Gordon, M. G., Andersen, S., Lu, Q., Rowson, A., Taylor, T. R. P., Clarke, L., Maccora, K., Chen, C., Cook, A. L., Ye, C. J., Fairfax, K. A., Hewitt, A. W., & Powell, J. E. (2022). Single-cell eQTL mapping identifies cell type–specific genetic control of autoimmune disease [Publisher: American Association for the Advancement of Science]. *Science*, *376*(6589), eabf3041. https://doi.org/10.1126/science.abf3041

Chapter 7

Discussion and conclusions

In this thesis, we increased the scalability of Kronecker-sum-structured models by multiple orders of magnitude; prior work could barely handle 1000 cell/gene datasets, whereas ours can handle millions of cells and genes. We also generalized the model to work with non-centered and multi-omic data, as well as considered some alternate models (causal graphs and strong products). In this chapter, we will highlight the implications of our work (Section 7.1), the limitations of our work (Section 7.2), and finally give a summary of potential future directions (Section 7.3). We have already begun investigating many of the future directions; if the PhD were one year longer, many of these ideas would have been fleshed out and turned into papers.

7.1 Implications

The primary implication of our work is that it unlocks the use of multi-axis models for multi-omics data. Before this thesis, multi-axis models were rarely ever applied to solve real problems in bioinformatics; this can be attributed to their lack of scalability, and to a lesser extent the lack of familiarity bioinformaticians have with such methods. It is quite telling that a Google Scholar search for "Kronecker structured" scrna-seq yields five results, four of which are from this thesis[1]. We have decisively solved the scalability problem, and have chosen to submit our most recent paper (Chapter 6) to the journal Bioinformatics in part to advertise such methods to the relevant research community.

Greater use of multi-axis models will lead to more insightful results on gene networks and cell heterogeneity. Practically all scRNA-seq studies perform cell clustering (Svensson et al., 2020); by jointly learning the cell and gene networks, our methodology can lead to cell clusterings *informed by the learned gene network*. Currently, cell clustering is usually done by methods naïve to gene dependencies. Such networks will also give a deeper understanding of tumor microenvironment structure. This is needed; for example, tumor heterogeneity is one of the challenges hindering the development of effective CAR-T treatments for solid tumors (Guzman et al., 2023).

There are also implications on the model development side, which we will discuss more in Section 7.3. By expanding multi-axis methodologies to causal networks (Chapter 5) and introducing new graph products (Chapter 4), we have laid the groundwork for future developments in those directions. By generalizing the use case to multi-tensor datasets (including multi-omics), we have increased the number of application domains that may benefit from such models.

7.2 Limitations

Despite these advances, our work still has some limitations. Most prior work used sparsity-inducing regularizers based on the graphical lasso; to achieve scalability, we do not use regularizers. The incompatibility of scalability and regularization is due to the use of eigendecompositions in the most efficient algorithms. In our first paper (Chapter 2) we showed that, without regularization, one only needs one eigendecomposition to fit the model. With regularization, it is much more expensive: one

[1]Search was done March 2nd, 2026. The fifth paper, by Tan and Ma, was published last year; they used a Kronecker-product structure to capture gene and spatial dependencies in spatial transcriptomics.

needs one eigendecomposition *per iteration*. It would be nice to have large-scale regularized models, but this would likely require the development of eigendecomposition-free algorithms.

Additionally, our forays away from Kronecker-sum-structured conditional dependency models (Chapters 4 and 5) are limited. We've proven the concept, but it will require more work to achieve strong performance and scalability. Kalaitzis et al. (2013) introduced BiGLasso in 2013, and it was not until our most recent work (Chapter 6) that Kronecker-sum-structured models became usable on large datasets; thus we might expect another decade until those models mature.

7.3 Future Directions

7.3.1 Applications

As a result of the advances in this thesis, multi-axis models are now a maturing technology, fit for use on real large-scale datasets. As such, the most interesting direction for future work is to start applying them to real problems. While we touched on a real application in Chapter 6, the application was not the focus of the chapter. Going forward, we would like to start doing application-focused research, in which multi-axis models are one of many methodologies used to achieve biologically meaningful results.

7.3.2 Beyond tensor-variate data

One interesting direction to take is to generalize the multi-axis methodology to less cleanly structured data. Currently, our work applies to multi-tensor datasets - what about ragged arrays? Example 24, below, gives a minimal example of such a dataset.

Example Dataset

Example 24 (Longitudinal blood pressure records)**.** *A dataset measuring the blood pressure over time for multiple patients. The timings between measurements were neither consistent nor recorded, and the number of measurements were not consistent either; some patients have many measurements, some have few.*

Form. A set of different-length vectors $\{\mathbf{v}^{(p)}\}$ (in other words, a ragged array), where $\mathbf{v}_t^{(p)}$ represents patient p's blood pressure at time t.

Even though Example 24 cannot be cleanly organized into the shape of a matrix (due to patients having a differing number of measurements), there still exist discernible 'axes' of the data: one 'patients' axis, and then for each patient a distinct 'time' axis. If we make an assumption about how the dependencies of these axes interact, we can formulate the dataset in a Kronecker-structured manner. Here is a plausible assumption one might make;

Assumption 12. $(p_1, t_1) \sim (p_2, t_2) \iff$ *either* $p_1 = p_2$ *and* $t_1 \sim t_2$*, or* $p_1 \sim p_2$

In other words, measurements within the same patient are connected if the timepoints are connected, and measurements between connected patients are always connected regardless of timepoint.

This matrix of conditional dependencies would then manifest as the following Kronecker operation (where $\mathbf{J}$ is the matrix of ones):

$$\begin{bmatrix} \mathbf{I} \otimes \mathbf{\Psi}_{\text{(time, for patient 1)}} & 0 & 0 & \cdots \\ 0 & \mathbf{I} \otimes \mathbf{\Psi}_{\text{(time, for patient 2)}} & 0 & \cdots \\ 0 & 0 & \mathbf{I} \otimes \mathbf{\Psi}_{\text{(time, for patient 3)}} & \vdots \\ \vdots & \vdots & \cdots & \ddots \end{bmatrix} + \mathbf{\Psi}_{\text{patients}} \otimes \mathbf{J}$$

There's nothing that fundamentally prohibits us from setting up a Gaussian model with a precision matrix having this structure, although it would likely require significant thought to be able to optimize it efficiently. The above example is somewhat of a toy example, but it lays the groundwork for the next example.

Example Dataset

Example 25 (Multi-patient scRNA-seq dataset)**.** *scRNA-seq data from multiple patients all with similar but not necessarily identical cancers (such as all having a lymphoma). The genes measured are the same across all patients, but the cells are different.*

Motivation. In addition to the benefits of having scRNA-seq for a single patient, such a dataset would allow us to construct gene networks that average out the heterogeneity of individual patients. Furthermore, if we are able to construct a patient network, it may lead to insights into the relatedness of different cancers, helping pinpoint (for example) the cell-type-of-origin.

Form. A set of different-shape matrices $\{\mathbf{M}^{(p)}\}$, where $\mathbf{M}^{(p)}_{cg}$ represents patient p's cell c's expression of gene g.

Example 25 can already be partially handled by our methodology - by picking the genes as a 'shared axis' and ignoring patient-patient conditional dependencies, it can be expressed as a set of matrices under our 'shared axis' framework introduced in Chapter 2. However, doing so would not capture the patient-patient relations that would potentially yield novel insights. It would also assume independence of patients, which would not be ideal when the sample contains multiple patients with the same cancer. To ameliorate these issues, we could make the following assumption:

Assumption 13. $(p_1, c_1, g_1) \sim (p_2, c_2, g_2) \iff$ *either* $p_1 = p_2$ *and* $(c_1, g_1) \sim_{\boxplus} (c_2, g_2)$*, or* $p_1 \sim p_2$

In other words, measurements within the same patient can be connected according to the Cartesian product (like most of the work in the thesis), and measurements between connected patients are always connected, regardless of gene/cell.

This would yield the following conditional dependency structure:

$$\begin{bmatrix} \mathbf{I} \otimes \left(\mathbf{\Psi}_{(\text{cells in patient 1})} \oplus \mathbf{\Psi}_{(\text{genes})} \right) & 0 & \cdots \\ 0 & \mathbf{I} \otimes \left(\mathbf{\Psi}_{(\text{cells in patient 2})} \oplus \mathbf{\Psi}_{(\text{genes})} \right) & \cdots \\ \vdots & \vdots & \ddots \end{bmatrix} + \mathbf{\Psi}_{\text{patients}} \otimes \mathbf{J}$$

We could alternatively make a different assumption (there are in fact many *other* assumptions one could make here, depending on how exactly one believes dependencies should be connected):

Assumption 14. $(p_1, c_1, g_1) \sim (p_2, c_2, g_2) \iff (p_1 = p_2$ *and* $(c_1, g_1) \sim_{\boxplus} (c_2, g_2))$ *or* $(p_1 \sim p_2$ *and* $g_1 = g_2)$

In other words, measurements within the same patient can be connected according to the Cartesian product (like most of the work in the thesis), and measurements between connected patients are always connected if they concern the same gene.

This would yield the following conditional dependency structure:

$$\begin{bmatrix} \mathbf{I} \otimes \left(\mathbf{\Psi}_{(\text{cells in patient 1})} \oplus \mathbf{\Psi}_{(\text{genes})} \right) & 0 & \cdots \\ 0 & \mathbf{I} \otimes \left(\mathbf{\Psi}_{(\text{cells in patient 2})} \oplus \mathbf{\Psi}_{(\text{genes})} \right) & \cdots \\ \vdots & \vdots & \ddots \end{bmatrix} + \mathbf{\Psi}_{\text{patients}} \otimes \left(\mathbf{J}_{(\text{all cells})} \otimes \mathbf{I}_{(\text{genes})} \right)$$

This framework can easily be extended to multi-omic and longitudinal contexts. It would likely be hard to optimize; whether the insights given would be worth the difficulty will only be able to be answered if actually implemented. This would be a very interesting direction for future work; if complex structures such as this can be efficiently optimized, the multi-axis methodology would be able to be applied to virtually any loosely structured dataset.

7.3.3 Beyond the Kronecker sum

In this thesis we have mostly focused on the Kronecker sum, which along with the Kronecker product are the only products commonly considered in prior work. Only a few other operations have been considered, such as the squared Kronecker sum (Wang & Hero, 2021; Wang et al., 2020) and our work in Chapter 4. Depending on whether the operation is inspired by a natural graph operation or a generative model, all existing single-modality models can be expressed in one of the following forms:

$$\boldsymbol{\Omega} = \sum_{r=0}^{R} \bigotimes_{\ell=0}^{L} \boldsymbol{\Psi}^{(r)}_{\text{axis } \ell} \qquad \text{(Based on a graph operation)}$$

$$\boldsymbol{\Omega} = \left(\sum_{r=0}^{R} \bigotimes_{\ell=0}^{L} \boldsymbol{\Psi}^{(r)}_{\text{axis } \ell} \right)^2 \qquad \text{(Based on a generative model)}$$

The generative-model-based approach includes the squared Kronecker sum, and arises from the following model for a dataset $\mathcal{X}$ and white noise $\mathcal{E}$:

$$\sum_{r=0}^{R} \mathcal{X} \times_1 \boldsymbol{\Psi}^{(r)}_{\text{axis } 1} \times_2 \cdots \times_L \boldsymbol{\Psi}^{(r)}_{\text{axis } L} = \mathcal{E}$$

It would be nice to have the capability to fit any one of these models efficiently, as they can have natural interpretations. Suppose one has R gene regulatory pathways, each encoded as $\boldsymbol{\Psi}^{(r)}_{\text{genes}}$. In each summand $\boldsymbol{\Psi}^{(r)}_{\text{genes}} \otimes \boldsymbol{\Psi}^{(r)}_{\text{cells}}$, $\boldsymbol{\Psi}^{(r)}_{\text{cells}}$ can be interpreted as capturing the way cells relate to each other *through regulatory pathway* r. In a sample of stem cells, there are several regulatory networks that one might expect to be captured, such as the process of differentiation or the cell cycle. In our learned cell networks, the cell network linked with the differentiation gene network might then have strong connections between cells differentiating into the same end cell type, and the cell network linked with the cell cycle gene network may have strong connections between cells at the same stage in the cell cycle.

The size of the decomposition, R, is called the 'Kronecker rank'. The Kronecker sum approach has Kronecker rank 2, as it needs two Kronecker products to be expressed ($\boldsymbol{\Psi}_1 \otimes \mathbf{I}$ and $\mathbf{I} \otimes \boldsymbol{\Psi}_2$); Kronecker product models have rank 1, and our strong product model in Chapter 4 has rank 2. For covariance models based on graph products, the Kronecker PCA methodology can learn decompositions of arbitrary rank (Tsiligkaridis & Hero, 2013); however, no such methodology has been applied to conditional dependency networks.

Sadly, there is good reason to believe that it would be exceedingly difficult to make any such algorithm efficient when $R \geq 3$. Performing maximum likelihood estimation on a normal distribution with a graph-operation-based precision matrix yields the following term $\mathbf{A}$ in the gradient (with $\mathbf{J}^{(ij)}$ being the matrix of zeros except for a 1 in position (i, j)):

$$\mathbf{A}^{(r)}_{ij} = \operatorname{tr}\left[\boldsymbol{\Omega}^{-1} \left(\mathbf{J}^{ij} \otimes \bigotimes_{\ell=1}^{L} \boldsymbol{\Psi}^{(r)}_{\ell} \right) \right]$$

Naïvely, $\mathbf{A}$ is incredibly expensive to compute; for $d \times d$ matrix data, it requires the inversion of a $d^2 \times d^2$ matrix, which takes $O(d^6)$ time (and $O(d^4)$ memory just to hold the result!). This is strongly intractable; in our tests, it became infeasible to compute this on our personal computer by $d = 100$. Most methods avoid this by using eigendecomposition tricks (such as Li et al. (2022) and our own work in Chapters 2, 6, and 4). Essentially, when $R \leq 2$ the summands can be neatly simultaneously diagonalized, requiring $O(d^3)$ time and reducing the problem to (where $\boldsymbol{\Lambda}$ is diagonal):

$$\mathbf{A}^{(r)}_{ij} = \operatorname{tr}\left[\left(\bigotimes_{\ell=0}^{L} \mathbf{V}^{(r)}_{\ell} \right) \boldsymbol{\Lambda}^{-1} \left(\bigotimes_{\ell=0}^{L} \mathbf{V}^{(r)}_{\ell} \right)^T \left(\mathbf{J}^{ij} \otimes \bigotimes_{\ell=1}^{L} \boldsymbol{\Psi}^{(r)}_{\ell} \right) \right]$$

Further simplifications are available from this form depending on the exact choice of graph operation, but even as-written this is much easier to compute. For $d \times d$ matrix data, inversion of $\mathbf{\Lambda}$ is $O(d^2)$, dwarfed by the $O(d^3)$ cost of simultaneous diagonalization and the cost of the remaining matrix multiplications. Thus, the gradient can be computed at cost asymptotically equivalent to the cost of $d \times d$ matrix multiplication - expensive, but acceptable.

However, this trick is completely unavailable when the Kronecker rank is three or more; under-the-hood, this is due to the fact that one cannot simultaneously diagonalize an arbitrary set of three or more matrices. In fact, this is likely harder than another well-known problem: suppose we consider the sub-problem of finding $\mathbf{\Omega}^{-1}\mathbf{x}$, where $\mathbf{\Omega}$ is Kronecker-structured and $\mathbf{x}$ is any vector (evaluating $\mathbf{\Omega}^{-1}\mathbf{M}$ for a matrix $\mathbf{M}$, like in our case, is necessarily harder than this, as each column of the result is the answer to $\mathbf{\Omega}^{-1}\mathbf{m}$ for columns $\mathbf{m}$ of $\mathbf{M}$). In other words, we want to solve for $\mathbf{y}$ as follows:

$$
\begin{aligned}
& & \mathbf{\Omega}^{-1}\mathbf{x} &= \mathbf{y} \\
\iff & & \mathbf{x} &= \mathbf{\Omega}\mathbf{y} \\
\iff & & \mathcal{X} &= \sum_r \mathcal{Y} \times_1 \mathbf{\Psi}_1^{(r)} \times_2 \cdots \times_L \mathbf{\Psi}_L^{(r)} \quad \text{(where } \mathcal{X}, \mathcal{Y} \text{ are sufficiently reshaped versions of } \mathbf{x}, \mathbf{y}\text{)}
\end{aligned}
$$

The last equation is known as a 'generalized Sylvester equation'; such equations are difficult to solve efficiently (see for example Dmytryshyn et al. (2022)). Barring significant (and unlikely) advances in solving generalized Sylvester equations, any efficient algorithm to evaluate $\operatorname{tr}\left[\mathbf{\Omega}^{-1}\left(\mathbf{J}^{ij} \otimes \bigotimes_{\ell=1}^{L} \mathbf{\Psi}_\ell^{(r)}\right)\right]$ would have to take advantage of some trick to directly evaluate the trace, rather than solving for a representation of $\mathbf{\Omega}^{-1}\left(\mathbf{J}^{ij} \otimes \bigotimes_{\ell=1}^{L} \mathbf{\Psi}_\ell^{(r)}\right)$ as a first step; it is not clear if such a trick exists.

Instead of solving for the maximum likelihood, one could choose a different loss function - and this is what we think the most fruitful direction for future work would be. During the PhD, we did attempt to use the D-Trace estimator (Zhang & Zou, 2014), which sidesteps the need to compute such a difficult term. Our preliminary results suggested that the resulting gradients were sufficiently fast enough to be computed many times at scale. However, we had difficulty getting the estimator to converge; eventually, we shifted focus to other areas. It is likely that, with more focused effort on the task, this problem would be overcome.

7.3.4 Becoming robust to noise

Not every cell is created equal; when collecting scRNA-seq data, some cells have higher quality gene counts than others. Typically, a scRNA-seq analysis starts by identifying 'low-quality' cells and discarding them. This is a binary choice; some cells will just barely pass the quality threshold (and thus be more corrupted than cells well above the threshold), and the information from all cells below the threshold is completely disregarded. While such information is corrupted, it still contains some useful signal. Likewise, some genes are more likely to be measured than others, due to technical biases such as GC-bias (Benjamini & Speed, 2012; Dohm et al., 2008).

A reasonable assumption is that such noise is multiplicative: due to GC-bias, a gene is p times less likely to be measured, regardless of which cell it appears in, and due to low quality it has q times less gene expression across all cells. Notably, we assume that cell-level effects affect all genes within the cell equally, and gene-level effects affect all cells equally. We can model this type of noise on the data as follows:

$$
\begin{aligned}
\mathbf{X} &= \mathbf{R}_{row}\mathbf{Z}\mathbf{R}_{col} && \text{(Corrupted Gaussian)} \\
\mathbf{R}_\ell &= \begin{bmatrix} r_1^{(\ell)} & 0 & \cdots \\ 0 & r_2^{(\ell)} & \cdots \\ \vdots & \vdots & \ddots \end{bmatrix} \succ 0 \\
\operatorname{vec}[\mathbf{Z}] &\sim \mathcal{N}\left(\mathbf{0}, \mathbf{\Omega}^{-1}\right) \\
r_i^{(\ell)} &\sim \text{any strictly positive distribution}
\end{aligned}
$$

We can then impose structure, such as the Kronecker sum, on $\mathbf{\Omega}$. If we think in terms of the vectorization of our data matrices, $\mathbf{x}$ and $\mathbf{z}$, this model can be rewritten much more concisely as:

$$\begin{aligned}\mathbf{x} &= (\mathbf{r}_{col} \otimes \mathbf{r}_{row}) \circ \mathbf{z} \\ \mathbf{z} &\sim \mathcal{N}\left(\mathbf{0}, \mathbf{\Omega}^{-1}\right) \\ r_i^{(\ell)} &\sim \text{any strictly positive distribution}\end{aligned}$$

This type of model would capture the type of noise we mentioned earlier, where cells/genes are measured disproportionately. This distribution is very general; all we require on the noise terms is that they are never zero or negative - they may have any distribution that fits that criteria. Such a model bears a strong resemblance to the class of elliptical distributions:

$$\begin{aligned}\mathbf{x} &= r\mathbf{z} && \text{(Elliptical distribution)} \\ \mathbf{z} &\sim \mathcal{N}\left(\mathbf{0}, \mathbf{\Psi}^{-1}\right) \\ r &\sim \text{any strictly positive distribution}\end{aligned}$$

To make the relation clearer, note that we can express elliptical distributions as so:

$$\begin{aligned}\text{vec}\left[\mathbf{X}\right] &= (\mathbf{1} \otimes \mathbf{r}) \circ \text{vec}\left[\mathbf{Z}\right] && \text{(Elliptical distribution, rewritten)} \\ \text{vec}\left[\mathbf{Z}\right] &\sim \mathcal{N}\left(\mathbf{0}, (\mathbf{I} \otimes \mathbf{\Psi})^{-1}\right) \\ r_i &\sim \text{any strictly positive distribution}\end{aligned}$$

During the write-up period of the thesis, I elaborated on these ideas, and developed a draft paper implementing them. This is attached as Appendix F, and is intended for submission at NeurIPS 2026. The work as attached is the rough draft, comprising wholly my own work; the version that will ultimately by submitted will likely be an expanded and improved version of what is presented here, based on supervisor feedback.

7.3.5 Multi-axis mixture models

One of the benefits of multi-axis models is their ability to create cell networks as well as gene networks. Cell networks can then be used to identify cell types via clustering. However, it is reasonable to expect that the gene networks governing a cell type is unique to the cell type - in other words, a B cell and a T cell should have different gene regulatory networks. What multi-axis models learn, then, is the *average* network over all the cell types in the sample. This could potentially be addressed by further advancing the future direction in Section 7.3.3 (each summand in the rank-R Kronecker decomposition may correspond to one of the R cell types in the sample).

One alternative way of addressing this is to merge multi-axis technology with Gaussian mixture models. A Gaussian mixture model is a type of model that simultaneously clusters a dataset and fits a Gaussian to each cluster.

$$\text{pdf}(\mathbf{x}) = \sum_{\text{cluster } c} \pi_c \text{pdf}_{\mathcal{N}}\left(\mathbf{x}|\boldsymbol{\mu}_c, \mathbf{\Omega}_c^{-1}\right)$$

(probability density function for Gaussian mixtur model)

where:

π_c is probability of belonging to cluster c

$\text{pdf}_{\mathcal{N}}\left(\mathbf{x}|\boldsymbol{\mu}_c, \mathbf{\Omega}_c^{-1}\right)$ is probability density function for normal distribution

There is no fundamental reason why one cannot apply a Kronecker-sum-structure to a Gaussian mixture model, although it will require a bit more care than a naive application as we typically only have one 'sample' in a multi-axis framework.

7.4 Conclusion

The work presented during this thesis has successfully made Kronecker-sum-structured conditional dependency models scalable, as well as introduced new structures to handle various scenarios and deficiencies. While future improvements will always be possible, we have outlined the main research directions we think are the most important, and in may cases have demonstrated how one would go about making such improvements. We believe we have improved the scalability of Kronecker-sum-structured models to such a point that they are usable in practice, and are excited to see what the future holds.

Bibliography for the discussion

Benjamini, Y., & Speed, T. P. (2012). Summarizing and correcting the GC content bias in high-throughput sequencing. *Nucleic Acids Research*, *40*(10), e72. https://doi.org/10.1093/nar/gks001

Dmytryshyn, A., Fasi, M., & Gulliksson, M. (2022). The dynamical functional particle method for multi-term linear matrix equations. *Applied Mathematics and Computation*, *435*, 127458. https://doi.org/10.1016/j.amc.2022.127458

Dohm, J. C., Lottaz, C., Borodina, T., & Himmelbauer, H. (2008). Substantial biases in ultra-short read data sets from high-throughput DNA sequencing. *Nucleic Acids Research*, *36*(16), e105. https://doi.org/10.1093/nar/gkn425

Guzman, G., Reed, M. R., Bielamowicz, K., Koss, B., & Rodriguez, A. (2023). CAR-T Therapies in Solid Tumors: Opportunities and Challenges. *Current Oncology Reports*, *25*(5), 479–489. https://doi.org/10.1007/s11912-023-01380-x

Kalaitzis, A., Lafferty, J., Lawrence, N. D., & Zhou, S. (2013). The Bigraphical Lasso. *Proceedings of the 30th International Conference on Machine Learning*, 1229–1237. Retrieved February 25, 2026, from https://proceedings.mlr.press/v28/kalaitzis13.html

Li, S., López-García, M., Lawrence, N. D., & Cutillo, L. (2022). Two-way Sparse Network Inference for Count Data. *Proceedings of The 25th International Conference on Artificial Intelligence and Statistics*, 10924–10938. Retrieved February 26, 2026, from https://proceedings.mlr.press/v151/li22g.html

Svensson, V., da Veiga Beltrame, E., & Pachter, L. (2020). A curated database reveals trends in single-cell transcriptomics. *Database*, *2020*, baaa073. https://doi.org/10.1093/database/baaa073

Tan, C., & Ma, Y. (2025). A spatially informed matrix normal model for gene co-expression analysis in spatial transcriptomics studies. *Nucleic Acids Research*, *53*(22), gkaf1264. https://doi.org/10.1093/nar/gkaf1264

Tsiligkaridis, T., & Hero, A. O. (2013). Covariance Estimation in High Dimensions Via Kronecker Product Expansions. *IEEE Transactions on Signal Processing*, *61*(21), 5347–5360. https://doi.org/10.1109/TSP.2013.2279355

Wang, Y., & Hero, A. (2021). SG-PALM: A Fast Physically Interpretable Tensor Graphical Model. *Proceedings of the 38th International Conference on Machine Learning*, 10783–10793. Retrieved February 26, 2026, from https://proceedings.mlr.press/v139/wang21k.html

Wang, Y., Jang, B., & Hero, A. (2020). The Sylvester Graphical Lasso (SyGlasso). *Proceedings of the Twenty Third International Conference on Artificial Intelligence and Statistics*, 1943–1953. Retrieved February 26, 2026, from https://proceedings.mlr.press/v108/wang20d.html

Zhang, T., & Zou, H. (2014). Sparse precision matrix estimation via lasso penalized D-trace loss. *Biometrika*, *101*(1), 103–120. https://doi.org/10.1093/biomet/ast059

Appendix

This page intentionally left blank.

Appendix A

GmGM: a fast multi-axis Gaussian graphical model

Supplementary Materials for:
GmGM: a Fast Multi-Axis Gaussian Graphical Model

Outline of the supplementary material

1. Notation
2. Proofs
    2.1. Permutations
    2.2. Derivation of the probability density function
    2.3. Gradient
    2.4. Maximum likelihood estimate for the eigenvectors
    2.5. Maximum likelihood estimate for the eigenvalues
    2.6. Incorporation of priors
    2.7. Covariance thresholding
3. Dependencies
4. Experiments
    4.1. Synthetic data
    4.2. COIL video
    4.3. EchoNet-Dynamic ECGs
    4.4. Mouse embryo stem cell transcriptomics
    4.5. LifeLines-DEEP metagenomics + metabolomics
    4.6. 10x Genomics flash frozen lymph nodes
5. Regularization
6. Asymptotic complexity
7. Centering the data

In the supplementary material, we give proofs of all theorems (Section A.2), provide code, pseudocode, and computer specs (Section A.3), give more informations on our experiments (Section A.4), describe how we incorporated regularization (Section A.5), and finally give an overview of the asymptotic complexity of ours and other algorithms (Section A.6).

Lemmas 1 and 2 are used to derive the Kronecker sum normal distributions's pdf, which has been known in prior work; we have included it for completeness. All other lemmas and theorems are novel.

A.1 Notation

In addition to the notation used in the main paper, we also introduce further notation to aid in the proofs. For working with tensors, Kolda and Bader (2009) proved to be an invaluable resource; we have borrowed their notation in most cases. The only exception is that we have chosen to denote the ℓ-mode matricization of a tensor $\mathcal{T}$ as $\text{mat}_\ell[\mathcal{T}]$ rather than $\mathcal{T}_{(\ell)}$, to highlight its similarity to $\text{vec}[\mathcal{T}]$ and free up the subscript for other purposes.

To keep track of lengths of axes, we define the following notation:

- d_ℓ^γ is the length of axis ℓ
- $d_{>\ell}^\gamma$ is the product of lengths of all axes after ℓ
- $d_{<\ell}^\gamma$ is the product of lengths of all axes before ℓ
- $d_{\setminus\ell}^\gamma$ is the product of lengths of all axes except for ℓ
- $d_\forall^\gamma$ is the product of lengths of all axes (i.e. the number of elements in $\mathcal{D}^\gamma$)
- $d_\forall = \sum_\gamma d_\forall^\gamma$ is the total number of elements across all datasets

In prior work, d_ℓ has been used to represent the lengths of axes but m_ℓ was used where we write $d_{\setminus\ell}$ (such as in Greenewald et al. (2019)). As prior work also used $\setminus\ell$ to represent leaving out the ℓth axis in other contexts (such as in Kalaitzis et al. (2013)), and the analogous definitions of $d_{>\ell}$ and $d_{<\ell}$ were convenient for use in proofs, we chose to introduce $d_{\setminus\ell}$ as the variable to represent leave-one-out length products. By representing all of these related concepts with similar symbols, we hope the maths will be easier to parse.

We will let $\mathbf{I}_a$ be the $a \times a$ identity matrix, which allows a concise definition of the Kronecker sum: $\bigoplus_\ell \mathbf{\Psi}_\ell = \sum_\ell \mathbf{I}_{d_{<\ell}} \otimes \mathbf{\Psi}_\ell \otimes \mathbf{I}_{d_{>\ell}}$.

We make frequent use of the vectorization $\text{vec}[\mathbf{M}]$ of a matrix $\mathbf{M}$, and more generally of a tensor $\text{vec}[\mathcal{T}]$. We adopt the rows-first convention of vectorization, such that:

$$\text{vec}\begin{bmatrix}1 & 2\\ 3 & 4\end{bmatrix} = \begin{bmatrix}1 & 2 & 3 & 4\end{bmatrix} \tag{A.1}$$

While columns-first is more common, rows-first is more natural when we adopt the convention that rows are the first axis of tensor; this is the convention that matricization uses, and matricization is much more important for our work due to its role in defining the Gram matrices. Note that, for matrices, a rows-first vectorization of $\mathbf{M}$ is equivalent to a columns-first vectorization of $\mathbf{M}^T$, so there is no fundamental difference between the two. For vectorizing a tensor, we proceed by stacking the rest of the axes in order, such that an element $(i_1, ..., i_K)$ in $\mathcal{T}$ gets mapped to the element $\sum_\ell i_\ell d_{<\ell}$ in $\text{vec}[\mathcal{T}]$.

We define the Gram matrices as $\mathbf{S}_\ell^\gamma = \text{mat}_\ell[\mathcal{D}^\gamma]\,\text{mat}_\ell[\mathcal{D}^\gamma]^T$. Typically we consider only the one-sample case. If you have multiple samples, indexed by a subscript i, then the Gram matrix becomes an average: $\mathbf{S}_\ell^\gamma = \frac{1}{n}\sum_i^n \text{mat}_\ell\left[\mathcal{D}_i^\gamma\right]\text{mat}_\ell\left[\mathcal{D}_i^\gamma\right]^T$.

An essential concept is that of the 'stridewise-blockwise trace', defined as:

$$\text{tr}_b^a[\mathbf{M}] = \left[\text{tr}\left[\mathbf{M}\left(\mathbf{I}_a \otimes \mathbf{J}^{ij} \otimes \mathbf{I}_b\right)\right]\right]_{ij} \tag{A.2}$$

Where $\mathbf{J}^{ij}$ is the matrix of zeros except at (i,j) where it has a 1. It is a generalization of the blockwise trace considered by Kalaitzis et al. (2013), and is related to the $\text{proj}_{\mathcal{K}}$ operation defined by Greenewald et al. (2019). Specifically, $\text{proj}_{\mathcal{K}}[\mathbf{M}]$ is equivalent to $\bigoplus_\ell \text{tr}_{d_{>\ell}}^{d_{<\ell}}[\mathbf{M}]$ up to an additive diagonal factor (Lemma 33 from Greenewald et al. (2019)). $\text{proj}_{\mathcal{K}}[\mathbf{M}]$ was defined to be the matrix that best approximates $\mathbf{M}$ (in terms of the Frobenius norm) while being Kronecker-sum-decomposable. This matrix is not unique; the choice by Greenewald et al. (2019) to include an additive factor was to enforce $\text{tr}\left[\text{proj}_{\mathcal{K}}[\mathbf{M}]\right] = 0$. We do not wish to enforce this constraint as it would be difficult to preserve in the multi-tensor case.

The parameter b of the stridewise-blockwise trace partitions the $m \times m$ matrix $\mathbf{M}$ into a block matrix with $b \times b$ blocks of size $(\frac{m}{b} \times \frac{m}{b})$. The parameter a then partitions these blocks into a 'strided' matrix with $a \times a$ strides containing $\frac{m}{ab} \times \frac{m}{ab}$ blocks. We take the trace of each stride, and the final matrix is the matrix of these traces. As this is conceptually complicated, we provide an example.

$$\text{tr}_2^2 \begin{bmatrix} 1&2&3&4&5&6&7&8\\ 1&2&3&4&5&6&7&8\\ 1&2&3&4&5&6&7&8\\ 1&2&3&4&5&6&7&8\\ 1&2&3&4&5&6&7&8\\ 1&2&3&4&5&6&7&8\\ 1&2&3&4&5&6&7&8\\ 1&2&3&4&5&6&7&8 \end{bmatrix} \tag{A.3}$$

$$= \text{tr}^2 \begin{bmatrix} \text{tr}\begin{bmatrix}1&2\\1&2\end{bmatrix} & \text{tr}\begin{bmatrix}3&4\\3&4\end{bmatrix} & \text{tr}\begin{bmatrix}5&6\\5&6\end{bmatrix} & \text{tr}\begin{bmatrix}7&8\\7&8\end{bmatrix} \\ \text{tr}\begin{bmatrix}1&2\\1&2\end{bmatrix} & \text{tr}\begin{bmatrix}3&4\\3&4\end{bmatrix} & \text{tr}\begin{bmatrix}5&6\\5&6\end{bmatrix} & \text{tr}\begin{bmatrix}7&8\\7&8\end{bmatrix} \\ \text{tr}\begin{bmatrix}1&2\\1&2\end{bmatrix} & \text{tr}\begin{bmatrix}3&4\\3&4\end{bmatrix} & \text{tr}\begin{bmatrix}5&6\\5&6\end{bmatrix} & \text{tr}\begin{bmatrix}7&8\\7&8\end{bmatrix} \\ \text{tr}\begin{bmatrix}1&2\\1&2\end{bmatrix} & \text{tr}\begin{bmatrix}3&4\\3&4\end{bmatrix} & \text{tr}\begin{bmatrix}5&6\\5&6\end{bmatrix} & \text{tr}\begin{bmatrix}7&8\\7&8\end{bmatrix} \end{bmatrix} \tag{A.4}$$

$$= \text{tr}^2 \begin{bmatrix} 3&7&11&15\\ 3&7&11&15\\ 3&7&11&15\\ 3&7&11&15 \end{bmatrix} \tag{A.5}$$

$$= \begin{bmatrix} \text{tr}\begin{bmatrix}3&11\\3&11\end{bmatrix} & \text{tr}\begin{bmatrix}7&15\\7&15\end{bmatrix} \\ \text{tr}\begin{bmatrix}3&11\\3&11\end{bmatrix} & \text{tr}\begin{bmatrix}7&15\\7&15\end{bmatrix} \end{bmatrix} \tag{A.6}$$

$$= \begin{bmatrix} 14&22\\14&22 \end{bmatrix} \tag{A.7}$$

Notice the construction of the 'strides' in Line A.6 - the parameter of 2 told us to take the trace of the most 'spread out' evenly spaced 2×2 strided submatrix.

A.2 Proofs

We will assume that no dataset contains repeated axes (i.e. no single tensor has two axes represented by the same graph), as this greatly affects the derived gradients. Shared axes - two tensors having one or more axes in common - are allowed. The case of shared axes is, after all, the whole point of developing this extension to prior work.

A.2.1 Permutations

Note that both $\text{vec}\left[\text{mat}_1\left[\mathcal{D}^\gamma\right]\right]$ and $\text{vec}\left[\text{mat}_\ell\left[\mathcal{D}^\gamma\right]\right]$ are row vectors containing the same elements, just in a different order. This means that there is a permutation matrix $\mathbf{P}_{\ell\to 1}$ such that $\text{vec}\left[\text{mat}_1\left[\mathcal{D}^\gamma\right]\right]^T \mathbf{P}_{\ell\to 1} =$

$\text{vec}\left[\text{mat}_\ell\left[\mathcal{D}^\gamma\right]\right]^T$.

Lemma 1 (Rearrangement lemma)**.** $\mathbf{P}_{\ell\to 1}\left(\mathbf{I}_{d_{<\ell}}\otimes\mathbf{\Psi}_\ell\otimes\mathbf{I}_{d_{>\ell}}\right)\mathbf{P}_{\ell\to 1}^T=\mathbf{\Psi}_\ell\otimes\mathbf{I}_{d_{\setminus\ell}}$

Proof. While vec, mat_ℓ and $\bigotimes$ are defined as operations on matrices, for the purposes of permutations we can consider them as operations on indices. We can express them as follows:

$$\text{vec}:(i_1,...,i_K)\to\left(\sum_\ell i_\ell d_{<\ell}\right) \tag{A.8}$$

$$\text{mat}_\ell:(i_1,...,i_K)\to\left(i_\ell,\sum_{\ell'<\ell}i_{\ell'}d_{<\ell'}+\sum_{\ell'>\ell}i_{\ell'}\frac{d_{<\ell'}}{d_\ell}\right) \tag{A.9}$$

$$\bigotimes:\left((i_1^1,i_1^2),...,(i_K^1,i_K^2)\right)\to\left(\sum_\ell i_\ell^1 d_{<\ell},\sum_\ell i_\ell^2 d_{<\ell}\right) \tag{A.10}$$

We'll consider just the rows of $\bigotimes$, $\bigotimes_{rows}$ - although the same argument applies with columns:

$$\bigotimes_{rows}:\left(i_1^1,...,i_K^1\right)\to\left(\sum_\ell i_\ell^1 d_{<\ell}\right) \tag{A.11}$$

Finally, we'll introduce the permutation operation $\sigma_{\ell\to 1}$ that will change the order of our Kronecker product:

$$\sigma_{\ell\to 1}:\left((i_1^1,i_1^2),...,(i_K^1,i_K^2)\right)\to\left(((i_\ell^1,i_\ell^2),(i_1^1,i_1^2),...,(i_{\ell-1}^1,i_{\ell-1}^2),(i_{\ell+1}^1,i_{\ell+1}^2),...,(i_K^1,i_K^2)\right) \tag{A.12}$$

And again without loss of generality we restrict ourself to $\sigma_{\ell\to 1}^{rows}$:

$$\sigma_{\ell\to 1}^{rows}:\left(i_1^1,...,i_K^1\right)\to\left(i_\ell^1,i_1^1,...,i_{\ell-1}^1,i_{\ell+1}^1,...,i_K^1\right) \tag{A.13}$$

After a Kronecker product our indices are in the form $\sum_\ell i_\ell d_{<\ell}$, and if we were to reorder it with $\sigma_{\ell\to 1}$ they would be in the form $i_\ell+\sum_{\ell'<\ell}i_{\ell'}d_{<\ell'}d_\ell+\sum_{\ell'>\ell}i_{\ell'}d_{<\ell'}$. Likewise, if we had matricized it we would have $\left(i_\ell,\sum_{\ell'<\ell}i_{\ell'}d_{<\ell'}+\sum_{\ell'>\ell}i_{\ell'}\frac{d_{<\ell'}}{d_\ell}\right)$, which is vectorized to $i_\ell+\sum_{\ell'<\ell}i_{\ell'}d_{<\ell'}d_\ell+\sum_{\ell'>\ell}i_{\ell'}d_{<\ell'}$. These reorderings are the same, and hence the matrix that represents it is $\mathbf{P}_{\ell\to 1}$.

□

A.2.2 Derivation of the probability density function

Recall that the Kronecker-sum-structured normal distribution for a single tensor is defined as follows:

$$\operatorname{vec}\left[\mathcal{D}^{\gamma}\right] \sim \mathcal{N}\left(\mathbf{0}, \left(\bigoplus_{\ell \in \gamma} \mathbf{\Psi}_{\ell}\right)^{-1}\right) \iff \mathcal{D}^{\gamma} \sim \mathcal{N}_{KS}\left(\left\{\Psi_{\ell}\right\}_{\ell \in \gamma}\right) \tag{A.14}$$

The log-likelihood for this distribution is given in Kalaitzis et al. (2013) for the matrix case and Greenewald et al. (2019) for the general tensor case. However, neither of these papers provide a derivation. As the full derivation will motivate the construction of lemmas useful for the proofs of Theorems G, we will give it here. First, observe that the density function is that of a normal distribution.

$$p\left(\mathcal{D}^{\gamma}\right) = \frac{\sqrt{\left|\bigoplus_{\ell \in \gamma} \mathbf{\Psi}_{\ell}\right|}}{(2\pi)^{\frac{d_{\forall}^{\gamma}}{2}}} e^{\frac{-1}{2}\operatorname{vec}\left[\mathcal{D}^{\gamma}\right]^{T}\left(\bigoplus_{\ell} \mathbf{\Psi}_{\ell}\right)\operatorname{vec}\left[\mathcal{D}^{\gamma}\right]} \tag{A.15}$$

Lemma 2 ($\oplus$-vec lemma)**.** $\operatorname{vec}\left[\mathcal{D}^{\gamma}\right]^{T}\left(\bigoplus_{\ell} \mathbf{\Psi}_{\ell}\right)\operatorname{vec}\left[\mathcal{D}^{\gamma}\right] = \sum_{\ell} \operatorname{tr}\left[\mathbf{S}_{\ell}^{\gamma}\mathbf{\Psi}_{\ell}\right]$

Proof. This proof relies on the following two properties of vec: $(\mathbf{A} \otimes \mathbf{B})\operatorname{vec}\left[\mathbf{C}\right] = \operatorname{vec}\left[\mathbf{B}\mathbf{C}^{T}\mathbf{A}^{T}\right]$ and $\operatorname{tr}\left[\mathbf{A}^{T}\mathbf{B}\right] = \operatorname{vec}\left[\mathbf{A}\right]^{T}\operatorname{vec}\left[\mathbf{B}\right]$. The $\mathbf{C}$ term picks up a transpose due to our use of the rows-first vectorization; when using columns-first notation the right hand side becomes $\operatorname{vec}\left[\mathbf{B}\mathbf{C}\mathbf{A}^{T}\right]$.

$$\begin{aligned}
\operatorname{vec}\left[\mathcal{D}^{\gamma}\right]\left(\bigoplus_{\ell} \mathbf{\Psi}_{\ell}\right)\operatorname{vec}\left[\mathcal{D}^{\gamma}\right] &= \sum_{\ell} \operatorname{vec}\left[\mathcal{D}^{\gamma}\right]^{T}\left(\mathbf{I}_{d_{<\ell}} \otimes \mathbf{\Psi}_{\ell} \otimes \mathbf{I}_{d_{>\ell}}\right)\operatorname{vec}\left[\mathcal{D}^{\gamma}\right] && \text{(Definition of } \bigoplus\text{)} \\
&= \sum_{\ell} \operatorname{vec}\left[\operatorname{mat}_{1}\left[\mathcal{D}^{\gamma}\right]\right]^{T}\left(\mathbf{I}_{d_{<\ell}} \otimes \mathbf{\Psi}_{\ell} \otimes \mathbf{I}_{d_{>\ell}}\right)\operatorname{vec}\left[\operatorname{mat}_{1}\left[\mathcal{D}^{\gamma}\right]\right] && \text{(A.16)} \\
&= \sum_{\ell} \operatorname{vec}\left[\operatorname{mat}_{\ell}\left[\mathcal{D}^{\gamma}\right]\right]^{T}\mathbf{P}_{\ell\to 1}^{T}\left(\mathbf{I}_{d_{<\ell}} \otimes \mathbf{\Psi}_{\ell} \otimes \mathbf{I}_{d_{>\ell}}\right)\mathbf{P}_{\ell\to 1}\operatorname{vec}\left[\operatorname{mat}_{\ell}\left[\mathcal{D}^{\gamma}\right]\right] && \text{(A.17)} \\
&= \sum_{\ell} \operatorname{vec}\left[\operatorname{mat}_{\ell}\left[\mathcal{D}^{\gamma}\right]\right]^{T}\left(\mathbf{\Psi}_{\ell} \otimes \mathbf{I}_{d_{\setminus\ell}}\right)\operatorname{vec}\left[\operatorname{mat}_{\ell}\left[\mathcal{D}^{\gamma}\right]\right] && \text{(Rearrangement Lemma)} \\
&= \sum_{\ell} \operatorname{vec}\left[\operatorname{mat}_{\ell}\left[\mathcal{D}^{\gamma}\right]\right]^{T}\operatorname{vec}\left[\operatorname{mat}_{\ell}\left[\mathcal{D}^{\gamma}\right]\mathbf{\Psi}_{\ell}^{T}\right] && \text{(A.18)} \\
&= \sum_{\ell} \operatorname{tr}\left[\mathbf{S}_{\ell}^{\gamma}\mathbf{\Psi}_{\ell}\right] && \text{(A.19)}
\end{aligned}$$

□

With this lemma, the probability density function in the single-tensor case can be expressed in the form:

$$p\left(\mathcal{D}^{\gamma}\right) = \frac{\sqrt{\left|\bigoplus_{\ell \in \gamma} \mathbf{\Psi}_{\ell}\right|}}{(2\pi)^{\frac{d_{\forall}^{\gamma}}{2}}} e^{\frac{-1}{2}\sum_{\ell} \operatorname{tr}\left[\mathbf{S}_{\ell}^{\gamma}\mathbf{\Psi}_{\ell}\right]} \tag{A.20}$$

Leading to the probability density function for the multi-tensor case as:

$$p\left(\{\mathcal{D}^\gamma\}\right) = \prod_\gamma \frac{\sqrt{\left|\bigoplus_{\ell\in\gamma} \mathbf{\Psi}_\ell\right|}}{(2\pi)^{\frac{d_\forall^\gamma}{2}}} e^{\frac{-1}{2}\sum_\ell \mathrm{tr}\left[\mathbf{S}_\ell^\gamma \mathbf{\Psi}_\ell\right]} \tag{A.21}$$

$$= \frac{\prod_\gamma \sqrt{\left|\bigoplus_{\ell\in\gamma} \mathbf{\Psi}_\ell\right|}}{(2\pi)^{\frac{d_\forall}{2}}} e^{\frac{-1}{2}\sum_\gamma\sum_\ell \mathrm{tr}\left[\mathbf{S}_\ell^\gamma \mathbf{\Psi}_\ell\right]} \tag{A.22}$$

$$= \frac{\prod_\gamma \sqrt{\left|\bigoplus_{\ell\in\gamma} \mathbf{\Psi}_\ell\right|}}{(2\pi)^{\frac{d_\forall}{2}}} e^{\frac{-1}{2}\sum_\ell \mathrm{tr}[\mathbf{S}_\ell \mathbf{\Psi}_\ell]} \tag{A.23}$$

The negative log-likelihood is thus:

$$\mathrm{NLL}\left(\{\mathcal{D}^\gamma\}\right) = \frac{d_\forall}{2}\log(2\pi) + \frac{1}{2}\sum_\ell \mathrm{tr}\left[\mathbf{S}_\ell \mathbf{\Psi}_\ell\right] - \frac{1}{2}\sum_\gamma \log\left|\bigoplus_{\ell\in\gamma} \mathbf{\Psi}_\ell\right| \tag{A.24}$$

A.2.3 Gradient

The derivation of the gradient of the negative log-likelihood is essentially the same as the derivation given by Kalaitzis et al. (2013) for the original Bi-Graphical Lasso. Our derivation is complicated by the fact that we are considering general tensors rather than matrices. We'll let sym be the symmetricizing operator that must be applied as we are taking the derivative with respect to a symmetric matrix: $\mathrm{sym}\left[\mathbf{M}\right] = \mathbf{K} \circ \mathbf{M}$, where $\mathbf{K}$ is a matrix with 1s on the diagonal and 2s everywhere else. We'll also define $\mathbf{J}^{ij}$ to be the matrix of zeros except for a 1 at position (i, j).

$$\frac{d}{d\mathbf{\Psi}_\ell}\text{NLL}\left(\{\mathcal{D}^\gamma\}\right) = \frac{1}{2}\text{sym}\left[\mathbf{S}_\ell\right] - \frac{1}{2}\sum_\gamma \frac{d}{d\mathbf{\Psi}_\ell}\log\left|\bigoplus_{\ell'\in\gamma}\mathbf{\Psi}_{\ell'}\right| \tag{A.25}$$

$$= \frac{1}{2}\text{sym}\left[\mathbf{S}_\ell\right] - \frac{1}{2}\sum_\gamma \text{tr}\left[\left(\bigoplus_{\ell'\in\gamma}\mathbf{\Psi}_{\ell'}\right)^{-1}\frac{d}{d\psi_\ell^{ij}}\bigoplus_{\ell'\in\gamma}\mathbf{\Psi}_{\ell'}\right]_{ij} \tag{A.26}$$

$$= \frac{1}{2}\text{sym}\left[\mathbf{S}_\ell\right] - \frac{1}{2}\sum_\gamma \text{tr}\left[\left(\bigoplus_{\ell'\in\gamma}\mathbf{\Psi}_{\ell'}\right)^{-1}\left(\mathbf{I}_{d_{<\ell}}\otimes\frac{d}{d\psi_\ell^{ij}}\mathbf{\Psi}_\ell\otimes\mathbf{I}_{d_{>\ell}}\right)\right]_{ij} \tag{A.27}$$

$$= \frac{1}{2}\text{sym}\left[\mathbf{S}_\ell\right] - \frac{1}{2}\sum_{\gamma|\ell\in\gamma} \text{tr}\left[\left(\bigoplus_{\ell'\in\gamma}\mathbf{\Psi}_{\ell'}\right)^{-1}\left(\mathbf{I}_{d_{<\ell}}\otimes\left(\mathbf{J}^{ij}+\mathbf{J}^{ji}-\delta_{ij}\mathbf{J}^{ij}\right)\otimes\mathbf{I}_{d_{>\ell}}\right)\right]_{ij} \tag{A.28}$$

$$= \frac{1}{2}\text{sym}\left[\mathbf{S}_\ell\right] - \frac{1}{2}\sum_{\gamma|\ell\in\gamma}\left[(2-\delta_{ij})\text{tr}\left[\left(\bigoplus_{\ell'\in\gamma}\mathbf{\Psi}_{\ell'}\right)^{-1}\left(\mathbf{I}_{d_{<\ell}}\otimes\mathbf{J}^{ij}\otimes\mathbf{I}_{d_{>\ell}}\right)\right]\right]_{ij} \tag{A.29}$$

$$= \frac{1}{2}\text{sym}\left[\mathbf{S}_\ell\right] - \frac{1}{2}\sum_{\gamma|\ell\in\gamma}(2\mathbf{J}-\mathbf{I})\circ\text{tr}\left[\left(\bigoplus_{\ell'\in\gamma}\mathbf{\Psi}_{\ell'}\right)^{-1}\left(\mathbf{I}_{d_{<\ell}}\otimes\mathbf{J}^{ij}\otimes\mathbf{I}_{d_{>\ell}}\right)\right]_{ij} \tag{A.30}$$

$$= \frac{1}{2}\text{sym}\left[\mathbf{S}_\ell\right] - \frac{1}{2}\sum_{\gamma|\ell\in\gamma}\text{sym}\left[\text{tr}_{d_{>\ell}^\gamma}^{d_{<\ell}^\gamma}\left[\left(\bigoplus_{\ell'\in\gamma}\mathbf{\Psi}_{\ell'}\right)^{-1}\right]\right] \tag{A.31}$$

The MLE occurs when this gradient is zero, i.e. when the following equation is satisfied:

$$\mathbf{S}_\ell = \sum_{\gamma|\ell\in\gamma}\text{tr}_{d_{>\ell}^\gamma}^{d_{<\ell}^\gamma}\left[\left(\bigoplus_{\ell'\in\gamma}\mathbf{\Psi}_{\ell'}\right)^{-1}\right] \tag{A.32}$$

In other words, our effective Gram matrices are the best Kronecker-sum decomposition of the covariance matrix of the maximum likelihood estimate. Unfortunately, the Kronecker-sum decomposition does not interact well with matrix inverses, so this does not directly yield an analytic solution. It does, however, yield a solution for the eigenvectors.

A.2.4 Maximum likelihood estimate for the eigenvectors

We first produce two lemmas to aid in the derivation.

Lemma 3 (Cyclic property of the stridewise-blockwise trace)**.** *For any matrices* $\mathbf{M}, \mathbf{A}_{a\times a}, \mathbf{B}_{b\times b}$*, we have that* $\mathbf{tr}_b^a\left[\left(\mathbf{A}\otimes\mathbf{I}\otimes\mathbf{B}\right)\mathbf{M}\right] = \mathbf{tr}_b^a\left[\mathbf{M}\left(\mathbf{A}\otimes\mathbf{I}\otimes\mathbf{B}\right)\right]$

Proof. This follows directly from the cyclic property of the (normal) trace operator and the definition of the stridewise-blockwise trace. □

Lemma 4 (Conjugacy extraction of the stridewise-blockwise trace)**.** *For any matrices* $\mathbf{M}$ *and* $\mathbf{V}$*, we have that* $\text{tr}_b^a\left[\left(\mathbf{I}_a\otimes\mathbf{V}\otimes\mathbf{I}_b\right)\mathbf{M}\left(\mathbf{I}_a\otimes\mathbf{V}\otimes\mathbf{I}_b\right)^T\right] = \mathbf{V}\text{tr}_b^a\left[\mathbf{M}\right]\mathbf{V}^T$.

Proof.

$$\mathrm{tr}_b^a\left[(\mathbf{I}_a\otimes\mathbf{V}\otimes\mathbf{I}_b)\,\mathbf{M}\,(\mathbf{I}_a\otimes\mathbf{V}\otimes\mathbf{I}_b)^T\right]=\left[\mathrm{tr}\left[(\mathbf{I}_a\otimes\mathbf{V}\otimes\mathbf{I}_b)\,\mathbf{M}\,(\mathbf{I}_a\otimes\mathbf{V}\otimes\mathbf{I}_b)^T\left(\mathbf{I}_a\otimes\mathbf{J}^{ij}\otimes\mathbf{I}_b\right)\right]\right]_{ij}$$
(Definition of tr_b^a)

Thanks to the Rearrangement Lemma, we can get this just in terms of the standard blockwise trace, for which there exists a convenient lemma from Dahl et al. (2013) that does the heavy lifting for us. Unfortunately, this requires inserting permutation matrices into every nook and cranny.

$$=\left[\mathrm{tr}\left[\mathbf{P}\,(\mathbf{I}_a\otimes\mathbf{V}\otimes\mathbf{I}_b)\,\mathbf{P}^T\mathbf{P}\mathbf{M}\mathbf{P}^T\mathbf{P}\,(\mathbf{I}_a\otimes\mathbf{V}\otimes\mathbf{I}_b)^T\,\mathbf{P}^T\mathbf{P}\left(\mathbf{I}_a\otimes\mathbf{J}^{ij}\otimes\mathbf{I}_b\right)\mathbf{P}^T\right]\right]_{ij} \tag{A.33}$$

$$=\left[\mathrm{tr}\left[(\mathbf{V}\otimes\mathbf{I}_{ab})^T\,\mathbf{P}\mathbf{M}\mathbf{P}^T\,(\mathbf{V}\otimes\mathbf{I}_{ab})\left(\mathbf{J}^{ij}\otimes\mathbf{I}_{ab}\right)\right]\right]_{ij} \tag{A.34}$$

$$=\mathrm{tr}_{ab}\left[(\mathbf{V}\otimes\mathbf{I}_{ab})\,\mathbf{P}\mathbf{M}\mathbf{P}^T\,(\mathbf{V}\otimes\mathbf{I}_{ab})^T\right] \quad \text{(Definition of } \mathrm{tr}_{ab}\text{)}$$

$$=\mathbf{V}\mathrm{tr}_{ab}\left[\mathbf{P}\mathbf{M}\mathbf{P}^T\right]\mathbf{V}^T \quad \text{(Lemma 2 of Dahl et al. (2013))}$$

We then can see analogously that $\mathrm{tr}_{ab}\left[\mathbf{P}\mathbf{M}\mathbf{P}^T\right]=\mathrm{tr}_b^a\left[\mathbf{M}\right]$, completing the proof.

□

Theorem (Theorem G)**.** *Let $\mathbf{V}_\ell\mathbf{e}_\ell\mathbf{V}_\ell^T$ be the eigendecomposition of $\mathbf{S}_\ell$. Then $\mathbf{V}_\ell$ are the eigenvectors of the maximum likelihood estimate of $\mathbf{\Psi}_\ell$.*

Proof.

$$\mathbf{S}_\ell=\sum_{\gamma|\ell\in\gamma}\mathrm{tr}_{d_{>\ell}^\gamma}^{d_{<\ell}^\gamma}\left[\left(\bigoplus_{\ell'\in\gamma}\mathbf{\Psi}_{\ell'}\right)^{-1}\right] \tag{A.35}$$

$$=\sum_{\gamma|\ell\in\gamma}\mathrm{tr}_{d_{>\ell}^\gamma}^{d_{<\ell}^\gamma}\left[\left(\bigoplus_{\ell'\in\gamma}\mathbf{V}_{\ell'}\mathbf{\Lambda}_{\ell'}\mathbf{V}_{\ell'}^T\right)^{-1}\right] \tag{A.36}$$

$$=\sum_{\gamma|\ell\in\gamma}\mathrm{tr}_{d_{>\ell}^\gamma}^{d_{<\ell}^\gamma}\left[\left(\bigotimes_{\ell'}\mathbf{V}_{\ell'}\right)\left(\bigoplus_{\ell'\in\gamma}\mathbf{\Lambda}_{\ell'}\right)^{-1}\left(\bigotimes_{ell'}\mathbf{V}_{\ell'}\right)^T\right] \tag{A.37}$$

$$=\sum_{\gamma|\ell\in\gamma}\mathrm{tr}_{d_{>\ell}^\gamma}^{d_{<\ell}^\gamma}\left[\left(\mathbf{I}_{d_{<\ell}}\otimes\mathbf{V}_\ell\otimes\mathbf{I}_{d_{>\ell}}\right)\left(\bigoplus_{\ell'\in\gamma}\mathbf{\Lambda}_{\ell'}\right)^{-1}\left(\mathbf{I}_{d_{<\ell}}\otimes\mathbf{V}_\ell\otimes\mathbf{I}_{d_{>\ell}}\right)^T\right] \quad \text{(Cyclic Property)}$$

$$=\sum_{\gamma|\ell\in\gamma}\mathbf{V}_\ell\mathrm{tr}_{d_{>\ell}^\gamma}^{d_{<\ell}^\gamma}\left[\left(\bigoplus_{\ell'\in\gamma}\mathbf{\Lambda}_{\ell'}\right)^{-1}\right]\mathbf{V}_\ell^T \quad \text{(Conjugacy Extraction)}$$

$$=\mathbf{V}_\ell\left(\sum_{\gamma|\ell\in\gamma}\mathrm{tr}_{d_{>\ell}^\gamma}^{d_{<\ell}^\gamma}\left[\left(\bigoplus_{\ell'\in\gamma}\mathbf{\Lambda}_{\ell'}\right)^{-1}\right]\right)\mathbf{V}_\ell^T \tag{A.38}$$

We conclude the proof by observing that the central matrix is diagonal, and thus the right hand side constitutes an eigendecomposition of $\mathbf{S}_\ell$. Thus $\mathbf{S}_\ell$ and $\mathbf{\Psi}_\ell$ share eigenvectors. □

A.2.5 Maximum likelihood estimate for the eigenvalues

In the previous section, we derived the eigenvectors of the maximum likelihood estimate. While interesting (they correspond to the principal components of our data), we need the eigenvalues to reconstruct $\mathbf{\Psi}_\ell$. In the main paper, we derived the gradient with respect to the Negative Log-Likelihood. Then, for a learning rate μ_t, gradient descent can be performed with the update equation $\mathbf{\Lambda}_\ell^{t+1} = \mathbf{\Lambda}_\ell^t - \mu_t \left[\mathbf{e}_\ell - \sum_{\gamma|\ell\in\gamma} \mathrm{tr}_{d_{>\ell}^\gamma}^{d_{<\ell}^\gamma}\left[\left(\bigoplus_{\ell'\in\gamma} \mathbf{\Lambda}_{\ell'}\right)^{-1}\right]\right]$. A reasonable restriction is to make $\mathbf{\Psi}_\ell$ is positive definite, in which case μ_t must be chosen to prevent $\mathbf{\Lambda}_\ell^t$ from becoming negative. Iterating over the eigenvalues reduces our optimization task from one with $\sum_\ell d_\ell^2$ parameters to one with $\sum_\ell d_\ell$ parameters.

A.2.6 Incorporation of priors

Theorem (Theorem 2: GmGM Eigenvector Estimator with Priors)**.** *Suppose we have the same setup as in Theorem G, and that we have a prior of the form:*

$$\prod_\ell g_\ell(\mathbf{\Theta}_\ell) e^{\mathrm{tr}\left[\eta_\ell(\mathbf{\Theta}_\ell)^T \mathbf{\Psi}_\ell\right] + h_\ell(\mathbf{\Psi}_\ell)} \tag{A.39}$$

In other words, our prior is an exponential distribution for $\mathbf{\Psi}_\ell$, in which $\mathbf{\Psi}_\ell$ is the sufficient statistic. Then, if h_ℓ depends only on the eigenvalues (i.e. it is 'unitarily invariant'), the eigenvectors of $\frac{1}{2}\mathbf{S}_\ell - \eta_\ell(\mathbf{\Theta}_\ell)$ are the eigenvectors $\mathbf{V}_\ell$ of the MAP estimate for $\mathbf{\Psi}_\ell$.

Proof. Observe that our log-prior is:

$$\sum_\ell \log g_\ell(\mathbf{\Theta}_\ell) + \mathrm{tr}\left[\eta_\ell(\mathbf{\Theta}_\ell)^T \mathbf{\Psi}_\ell\right] + h_\ell(\mathbf{\Psi}_\ell) \tag{A.40}$$

With gradient:

$$\eta_\ell(\mathbf{\Theta}_\ell) + \frac{\partial}{\partial \mathbf{\Psi}_\ell} h_\ell(\mathbf{\Psi}_\ell) \tag{A.41}$$

Theorem 1.1 from Lewis (1996) states that the eigenvectors of the derivative of any spectral function are the same as the eigenvectors of its input. In our case, this means that the eigenvectors of $\frac{\partial}{\partial \mathbf{\Psi}_\ell} h_\ell(\mathbf{\Psi}_\ell)$ are $\mathbf{V}_\ell$, the eigenvectors of $\mathbf{\Psi}_\ell$.

Recall the gradient of the log-likelihood of the Kronecker-sum-structured normal distribution was:

$$\frac{1}{2}\sum_{\gamma|\ell\in\gamma} \mathrm{tr}_{d_{>\ell}^\gamma}^{d_{<\ell}^\gamma}\left[\left(\bigoplus_{\ell'\in\gamma} \mathbf{\Psi}_{\ell'}\right)^{-1}\right] - \frac{1}{2}\mathbf{S}_\ell \tag{A.42}$$

The gradient of the log-likelihood after including the prior is the sum of these two gradients:

$$\left(\frac{1}{2}\sum_{\gamma|\ell\in\gamma} \mathrm{tr}_{d_{>\ell}^\gamma}^{d_{<\ell}^\gamma}\left[\left(\bigoplus_{\ell'\in\gamma} \mathbf{\Psi}_{\ell'}\right)^{-1}\right] + \frac{\partial}{\partial \mathbf{\Psi}_\ell} h_\ell(\mathbf{\Psi}_\ell)\right) - \frac{1}{2}\mathbf{S}_\ell + \eta_\ell(\mathbf{\Theta}_\ell) \tag{A.43}$$

And the optimum is achieved at:

$$\left(\frac{1}{2}\sum_{\gamma|\ell\in\gamma} \mathrm{tr}_{d_{>\ell}^\gamma}^{d_{<\ell}^\gamma}\left[\left(\bigoplus_{\ell'\in\gamma} \mathbf{\Psi}_{\ell'}\right)^{-1}\right] + \frac{\partial}{\partial \mathbf{\Psi}_\ell} h_\ell(\mathbf{\Psi}_\ell)\right) = \frac{1}{2}\mathbf{S}_\ell - \eta_\ell(\mathbf{\Theta}_\ell) \tag{A.44}$$

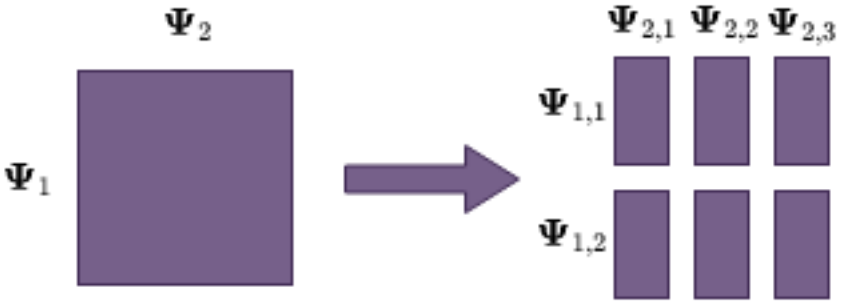


Figure A.1: Graphical representation of the benefits of Theorem 3. Provided the thresholded covariance matrix can be partitioned, so can the precision matrix. This allows us to split the dataset into distinct parts, and estimate the covariance matrices separately. Note that, in the diagram, we have split the dataset into 6 chunks but there are only 5 precision matrices to estimate - this is because there are shared axes, necessitating the use of our shared axis framework to take advantage of this partitioning.

The right hand side is known, and hence has known eigenvectors. The left hand side is unknown, but has eigenvectors $\mathbf{V}_\ell$.

□

The change to the iterative portion of the GmGM algorithm is minimal; by virtue of being unitarily invariant and convex, Lewis (1996) guarantees that $\frac{\partial}{\partial \mathbf{\Psi}_\ell} h_\ell(\mathbf{\Psi}_\ell)$ is some function of just the eigenvalues $\mathbf{\Lambda}_\ell$ of $\mathbf{\Psi}_\ell$. This function is added to our gradient each step (Line A.41). This fact is significant, as it means the algorithm does not need major changes to incorporate priors - we only need to consider $\frac{\partial}{\partial \mathbf{\Lambda}_\ell} h_\ell(\mathbf{\Lambda}_\ell)$. $\frac{1}{2}\mathbf{S}_\ell - \eta_\ell(\mathbf{\Theta}_\ell)$ can be thought of as our 'priorized' effective Gram matrix, its eigenvalues fill the role exactly of $\mathbf{e}_\ell$ in the original algorithm. Thus, the only change needed to incorporate priors in the algorithm, other than use of the priorized Gram matrices, is inclusion of the $\frac{\partial}{\partial \mathbf{\Lambda}_\ell} h_\ell(\mathbf{\Lambda}_\ell)$ term.

Corollary 3. *The Wishart and matrix gamma distributions all satisfy Theorem 2.*

A.2.7 Covariance thresholding

In this section, we restate and prove Theorem 4 from the main paper in more general terms; see Figure A.1. When we say that two matrices $\mathbf{A}$ and $\mathbf{B}$ have the same block diagonal structure, we mean that both have the following form:

$$\mathbf{B} = \begin{bmatrix} \mathbf{B}_1 & 0 & \dots \\ 0 & \mathbf{B}_2 & \\ \vdots & & \ddots \end{bmatrix} \tag{A.45}$$

$$\mathbf{A} = \begin{bmatrix} \mathbf{A}_1 & 0 & \dots \\ 0 & \mathbf{A}_2 & \\ \vdots & & \ddots \end{bmatrix} \tag{A.46}$$

Where there is no restrictions placed on the submatrices except that they are square and that the shape of $\mathbf{A}_i$ is the same as the shape of $\mathbf{B}_i$ for all i. The structure of our proof of Lemma 6 follows the same structure as that of Theorem 1 of Mazumder and Hastie (2012); we give it in a more general form in such that both their result and our result arise as corollaries in conjunction with Lemma 6.

Lemma 5. *Let* $\mathbf{F}\left[\mathbf{\Psi}_\ell; \{\mathbf{\Psi}_{\ell'}\}_{\ell' \neq \ell}\right]$ *be a function that preserves the block diagonal structure of* $\mathbf{\Psi}_\ell$*, which as a shorthand we will denote* $\mathbf{F}_\ell$*. Suppose we had an objective whose gradient is* $\mathbf{S}_\ell - \mathbf{F}_\ell$*. If we incorporate an L1 penalty on* $\mathbf{\Psi}_\ell$ *with strength* ρ *into the objective, then the subgradient of the function contains zero for an estimate* $\hat{\mathbf{\Psi}}_\ell$ *with the following property:*

If we replace every element of $\mathbf{S}_\ell$ whose magnitude is less than ρ with 0, we can partition the encoded graph into disconnected components. This partition is the same as the partition into disconnected components for $\hat{\mathbf{\Psi}}_\ell$.

Proof. The incorporation of an L1 penalty leads to a subgradient of:

$$\mathbf{S}_\ell - \mathbf{F}_\ell + \rho \text{sign} \mathbf{\Psi}_\ell \tag{A.47}$$

Where sign is the element-wise sign function:

$$\text{sign} \triangleq [\mathbf{M}]_{ij} \begin{cases} \in [-\rho, \rho] & \text{if } \mathbf{M}_{ij} = 0 \\ = -\rho & \text{if } \mathbf{M}_{ij} > 0 \\ = \rho & \text{if } \mathbf{M}_{ij} < 0 \end{cases} \tag{A.48}$$

We have critical points when the subgradient contains zero; this leads to the KKT conditions, which must be satisfied for all ij.

$$\left|\mathbf{S}_\ell^{ij} - \mathbf{F}_\ell^{ij}\right| \leq \rho \qquad \text{if } \mathbf{\Psi}_\ell^{ij} = 0 \tag{A.49}$$

$$\mathbf{F}_\ell^{ij} = \mathbf{S}_\ell^{ij} + \rho \qquad \text{if } \mathbf{\Psi}_\ell^{ij} > 0 \tag{A.50}$$

$$\mathbf{F}_\ell^{ij} = \mathbf{S}_\ell^{ij} - \rho \qquad \text{if } \mathbf{\Psi}_\ell^{ij} < 0 \tag{A.51}$$

Define $\text{thresh}_\rho[\mathbf{S}_\ell]$ to be a matrix with 0s everywhere that $\left|\mathbf{S}_\ell^{ij}\right| < \rho$ and otherwise is $\mathbf{S}_\ell^{ij}$. Without loss of generality, we can assume our data has been ordered such that $\text{thresh}_\rho[\mathbf{S}_\ell]$ is block diagonal. We will construct a $\hat{\mathbf{\Psi}}_\ell$ that satisfies the KKT conditions while also having a block-diagonal structure. The construction of such a $\hat{\mathbf{\Psi}}_\ell$ would show that the graph partition in our L1-penalized solution is at least as fine as the graph partition arising from thresholding $\mathbf{S}_\ell$.

Let us construct $\hat{\mathbf{\Psi}}_\ell$ to be block-diagonal with the same block structure as $\text{thresh}_\rho[\mathbf{S}_\ell]$. Denote the blocks to be indexed by a ($\hat{\mathbf{\Psi}}_{\ell,a}$ for $\mathbf{\Psi}$, $\mathbf{S}_{\ell,a}$ for $\mathbf{S}$), and define them to be solutions to the following problem:

$$0 \in \mathbf{S}_{\ell,a} - \mathbf{F}\left[\mathbf{\Psi}_{\ell,a}; \{\mathbf{\Psi}_{\ell'}\}_{\ell' \neq \ell}\right] + \rho \text{sign} \mathbf{\Psi}_{\ell,a} \tag{A.52}$$

To see why $\hat{\mathbf{\Psi}}_\ell$ satisfies the KKT conditions, note that for any index ij lying off the block diagonal, $\mathbf{S}_\ell^{ij} < \rho$ by construction. As $\mathbf{F}_{ij}$ preserves the sparsity structure of $\hat{\mathbf{\Psi}}_\ell$, and $\hat{\mathbf{\Psi}}_\ell^{ij} = 0$, $|\mathbf{S}_\ell^{ij} - \mathbf{F}_{ij}| = 0 < \rho$ is satisfied. For the indices ij lying on the block diagonal, note that the KKT conditions are satisfied via each block's constriction in Equation A.52.

To conclude the proof, it is necessary to show that if our solution $\hat{\mathbf{\Psi}}_\ell$ has a block diagonal structure, then $\mathbf{S}_\ell$ has the same block diagonal structure. In other words, the L1-penalized solution is not a more fine-grained partition than that which can be attained through covariance thresholding. Suppose that we have some $\mathbf{\Psi}_\ell$ with block-diagonal structure, and thus that $\mathbf{F}_\ell$ has that same structure. Then, $\mathbf{F}_\ell^{ij} = 0$ implies $\left|\mathbf{S}_\ell^{ij}\right| \leq \rho$, as by the KKT we know that $\left|\mathbf{S}_\ell^{ij} - \mathbf{F}_\ell^{ij}\right| \leq \rho$. Thus, $\text{thresh}_\rho[\mathbf{S}_\ell]$ has the same block diagonal structure as $\mathbf{\Psi}_\ell$.

We have now shown that both graph partitions are equally fine-grained, and are thus equivalent. This completes the proof. □

Corollary 4. *As the matrix inverse preserves block diagonal structure, Theorem 1 of Mazumder and Hastie (2012) follows immediately.*

Lemma 6. $\sum_{\gamma | \ell \in \gamma} \text{tr}_{d_{>\ell}^\gamma}^{d_{<\ell}^\gamma}\left[\left(\bigoplus_{\ell' \in \gamma} \mathbf{\Psi}_{\ell'}\right)^{-1}\right]$ *preserves the block diagonal structure of $\mathbf{\Psi}_\ell$.*

Proof. For notational convenience, and without loss of generality, assume that ℓ is the first axis in each of the modalities it appears in. Furthermore, observe that if for all γ we have that $\text{tr}_{d_{>\ell}^\gamma}^{d_{<\ell}^\gamma}\left[\left(\bigoplus_{\ell' \in \gamma} \mathbf{\Psi}_{\ell'}\right)^{-1}\right]$

preserves the block diagonal structure, then the sum $\sum_{\gamma|\ell\in\gamma} \mathrm{tr}_{d_{<\ell}^{\gamma} d_{>\ell}^{\gamma}} \left[\left(\bigoplus_{\ell'\in\gamma} \mathbf{\Psi}_{\ell'} \right)^{-1} \right]$ also preserves this structure. Thus, we consider only one term in the sum and omit reference to γ. Doing this, our problem reduces to showing that $\mathrm{tr}_{d_{\setminus\ell}} \left[\left(\mathbf{\Psi}_\ell \otimes \mathbf{I}_{d_{\setminus\ell}} + \mathbf{I}_{d_\ell} \otimes \bigoplus_{\ell'\neq\ell} \mathbf{\Psi}_{\ell'} \right)^{-1} \right]$ preserves the block diagonal structure.

While $\mathbf{\Psi}_\ell \otimes \mathbf{I}_{d_{\setminus\ell}}$ does not preserve the block diagonal structure in the strict sense (as its blocks have different sizes than those of $\mathbf{\Psi}_\ell$), it is still block diagonal and there is an obvious mapping from the blocks of one to the blocks of the other:

$$\begin{bmatrix} \mathbf{\Psi}_\ell^1 & 0 & \dots \\ 0 & \mathbf{\Psi}_\ell^2 & \\ \vdots & & \ddots \end{bmatrix} \otimes \mathbf{I}_{d_{\setminus\ell}} = \begin{bmatrix} \mathbf{\Psi}_\ell^1 \otimes \mathbf{I}_{d_{\setminus\ell}} & 0 & \dots \\ 0 & \mathbf{\Psi}_\ell^2 \otimes \mathbf{I}_{d_{\setminus\ell}} & \\ \vdots & & \ddots \end{bmatrix} \tag{A.53}$$

Suppose block $\mathbf{\Psi}_i$ has size s_i, then the size of the blocks in $\mathbf{\Psi}_\ell \otimes \mathbf{I}_{d_{\setminus\ell}}$ is $s_i \times d_{\setminus\ell}$. The second term in the sum is also block diagonal:

$$\begin{bmatrix} 1 & 0 & \dots \\ 0 & 1 & \\ \vdots & & \ddots \end{bmatrix} \otimes \bigoplus_{\ell'\neq\ell} \mathbf{\Psi}_{\ell'} = \begin{bmatrix} \bigoplus_{\ell'\neq\ell} \mathbf{\Psi}_{\ell'} & 0 & \dots \\ 0 & \bigoplus_{\ell'\neq\ell} \mathbf{\Psi}_{\ell'} & \\ \vdots & & \ddots \end{bmatrix} \tag{A.54}$$

But these blocks are of size $1 \times d_{\setminus\ell}$; no larger than the blocks of $\mathbf{\Psi}_\ell \otimes \mathbf{I}_{d_{\setminus\ell}}$. Thus, overall, $\mathbf{\Psi}_\ell \otimes \mathbf{I}_{d_{\setminus\ell}} + \mathbf{I}_{d_\ell} \otimes \bigoplus_{\ell'\neq\ell} \mathbf{\Psi}_{\ell'}$ has block diagonal structure with block sizes $s_i \times d_{\setminus\ell}$. It follows that its inverse does as well. The blockwise trace maps blocks of size $s_i \times d_{\setminus\ell}$ back to size s_i, completing the proof. □

Theorem 13. *Set all elements of $\mathbf{S}_\ell$ whose absolute value is less than ρ to 0. This encodes a potentially disconnected graph. Likewise, consider $\mathbf{\Psi}_\ell$ to be the estimated precision matrix for our model equipped with an L1 penalty of strength ρ. This also encodes a potentially disconnected graph. If we label the vertices by which disconnected component they are part of, then this labeling is the same in both procedures (the procedure with $\mathbf{S}_\ell$ and the procedure with $\mathbf{\Psi}_\ell$).*

Proof. This follows directly from the convexity of our objective as well as Lemmas 5 and 6. □

A.3 Dependencies

All tests and figures were generated on a Mac with an M1 chip and 8GB of RAM. Along with our code, we provide an environment file (environment.yml) that contains full details of all the dependencies used. In our GitHub repository https://github.com/AIStats-GmGM/GmGM/), we give precise and simple instructions on how to create a conda environment with the same setup as ours. Most of the packages used were specific to the experiments we ran. The dependencies necessary for our algorithm were Python 3.9 and NumPy 1.23.5.

A.4 Experiments

A.4.1 Synthetic data

For various precision-recall performances, see Figures A.2, A.3, and A.4. For runtimes, see Figure A.5. We can see that our algorithm performs comparably to prior work; it does slightly worse without the restricted L1 penalty. We focused on experimenting with the 1-sample case, as that is typically the case with real-world data (all of the five real-world experiments in Section A.4 are an example of this).

For the runtimes, we cut off our tests once GmGM reached the one-minute runtime mark, or if the size of the synthetic data exceeded a gigabyte; see Table A.1 for an overview, and Section A.6 for the computational complexity. Note that the runtime is almost linear in the total size of

The GmGM algorithm

Input: $\{\mathcal{D}_i^\gamma\}$, tolerance
Output: $\{\mathbf{\Psi}_\ell\}$

for $1 \leq \ell \leq K$
  $\mathbf{S}_\ell \leftarrow \sum_{\gamma|\ell\in\gamma} \frac{1}{n^\gamma} \sum_i^{n^\gamma} \mathrm{mat}_\ell \left[\mathcal{D}_i^\gamma\right] \mathrm{mat}_\ell \left[\mathcal{D}_i^\gamma\right]^T$
  $\mathbf{V}_\ell \leftarrow \mathrm{eigenvectors}[\mathbf{S}_\ell]$
  $\mathbf{e}_\ell \leftarrow \mathrm{eigenvalues}[\mathbf{S}_\ell]$
end for
$\mathbf{\Lambda} \leftarrow \begin{bmatrix}1 & ... & 1\end{bmatrix}^T$
$\mu \leftarrow 1$
while not converged
  for $1 \leq \ell \leq K$
    $\mathbf{G}_\ell^\gamma \leftarrow \mathrm{proj}_{KS}\left[\left(\bigoplus_{\ell'\in\gamma} \mathbf{\Lambda}_\ell\right)^{-1}\right]$
    $\mathbf{\Lambda}'_\ell \leftarrow \mathbf{\Lambda}_\ell - \mu\left[\mathbf{e}_\ell - \sum_{\gamma|\ell\in\gamma} \mathbf{G}_\ell^\gamma\right]$
  end for
  for $1 \leq \ell \leq K$
    $\mathbf{\Lambda}_\ell \leftarrow \mathbf{\Lambda}'_\ell$
  end for
  for $\gamma \in$ modalities
    if $\sum_{\ell\in\gamma} \min\mathbf{\Lambda}_\ell <$ tolerance **then**
      decrease μ to make sum far from zero
    end if
  end for
end while
for $1 \leq \ell \leq K$
  $\mathbf{\Psi}_\ell \leftarrow \mathbf{V}_\ell\mathbf{\Lambda}_\ell\mathbf{V}_\ell^T$
end for

the dataset $d_\forall$ ($O(Kd_\forall^{1+\frac{1}{K}})$. As K increases, $\frac{1}{K}$ vanishes. The cost of Gram matrix computation and eigendecomposition compared to the iterations also levels out; this is likely what explains the higher-order tensors all taking roughly the same amount of time to compute 1GB-sized datasets (10 seconds). These results clearly show that for high-axis-data ($K > 3$), our algorithm can handle all smaller-than-RAM datasets in reasonable time on a personal computer. For low-axis data ($K < 4$), it may take more than a minute but, as we can see from our algorithm's performance on large real world datasets such as the 10x Genomics dataset, its runtime will still be measured in minutes, not hours, before reaching the limits of RAM. When we consider the covariance-thresholding trick (Figure 7b in the main paper), we can push these limits even further.

Amount of Axes (K)	d_{GB}	$d_{\min}$	Runtime at d_{GB}
2	11,000	4,000	???
3	500	400	???
4	100	???	7.1 seconds
5	40	???	9.5 seconds
6	20	???	7.4 seconds

Table A.1: Comparing the point at which a K-axis tensor of double precision floats surpasses a gigabyte of data ($\sqrt[K]{\frac{1000000000}{64}} = d_{\text{GB}}$) to the point at which GmGM's runtime surpasses a minute ($d_{\min}$). Values are rounded.

In all tests, our regularized algorithm took about the same time as the unregularized algorithm. This is not unexpected, as our restricted regularization only affects the iterations, not the eigendecomposition. In the real world, where the data may be 'harder' than our synthetic data (i.e. take more iterations to converge), we would expect our regularized algorithm to be slower.

A.4.2 COIL video

The dataset

We downloaded the processed COIL-20 (Nene et al., n.d.) dataset, and tested our model on it. It is available at https://cave.cs.columbia.edu/repository/COIL-20; we used the 'processed' data. They do not formally license it, but state that it is available for academic purposes and provide the download links with no restrictions.

Experiment justification

In the original paper introducing Kronecker-sum-structured normal distributions for graphical models, Kalaitzis et al. (2013) showed that their BiGraphical Lasso (BiGLasso) could reconstruct the ordering of the frames of the video. Thus, it seemed natural to choose this as an experiment, to show that our lack of L1 regularization was not a significant inhibitor for the model in the real world setting.

BiGLasso was a significant algorithm in that it was the first of this type of model and proved that it could work, but it was also incredibly slow (taking around 15 minutes on 100x100 matrix data) and limited to two axes. Thus, they reduced the resolution of the image from 72 frames and 128x128 pixels to 36 frames and 9x9 pixels and flattened the rows and columns axis into one axis. We wanted to show that our algorithm could handle the full resolution in negligible time while still reconstructing the video.

Results

We wanted to see if our model could understand the structure of a video, which we expected to consist of two linear graphs (for the rows and columns, i.e. each row is connected only to its neighbor rows) and a circular graph (for the frames, because the video is of a 360° rotation). To generate these graphs, we ran our algorithm on the duck video from the dataset, and then greedily kept the largest edge from each vertex such that vertices in the final graph had at most two edges. If we shuffled our data (shuffle rows, columns, and frames) and try to reconstruct it with these graphs, we get decent without the nonparanormal skeptic (Figure A.6a) and great results with it (Figure A.6b). While in both cases the algorithm has clearly reconstructed most of the duck, when using the nonparanormal skeptic our

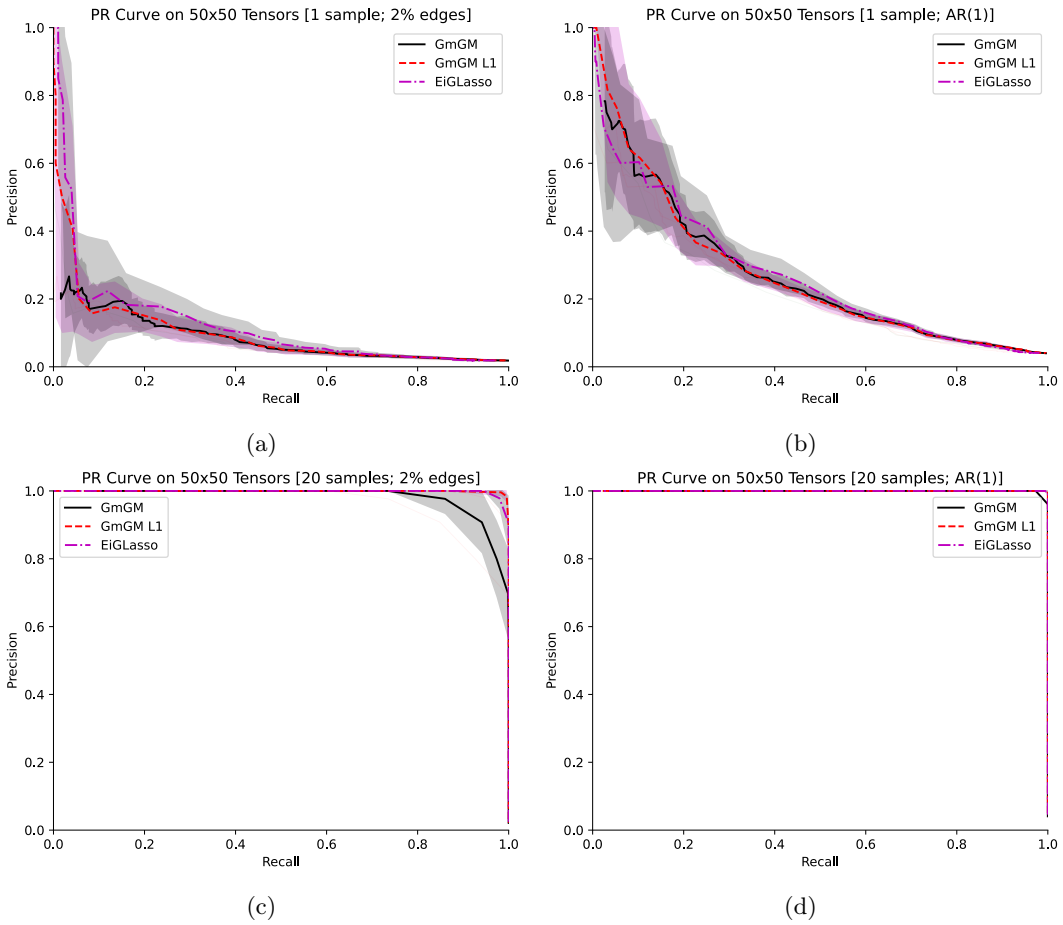


Figure A.2: Various precision-recall curves on synthetic matrix-variate data, for both an Erdos-Renyi with a 2% per-edge probability and AR(1) distribution.

algorithm reconstructs it nearly perfectly. The only human-noticeable issue is that it does not know which row/column the duck should start on - if we tessellate the image, the reconstruction would be indistinguishable from the original image.

We can put a numeric value to the reconstruction, by measuring the percentage of the time that our reconstructed edges connect two adjacent rows/columns/frames. Without the nonparanormal skeptic, we get an accuracy of 88% for the rows, 94% for the columns, and 93% for the frames. With it, the accuracy becomes 99% for all three. These values are without regularization; we did experiment with regularization, but it did not seem to help. TeraLasso had the same results.

On this dataset, our algorithm runs in 0.044 seconds whereas TeraLasso takes 33 seconds, showing that our algorithm is highly performant on real world data. For data of this size, the nonparanormal skeptic is expensive compared to our algorithm - but our algorithm is still an order of magnitude faster than TeraLasso. If we do not include the nonparanormal skeptic in our runtime calculation, the speed of GmGM is comparable regardless of the input. However, convergence is much quicker for TeraLasso when it is fed the nonparanormal skeptic (Table A.2).

A.4.3 EchoNet-Dynamic ECGs

The dataset

We downloaded all of the EchoNet-Dynamic (Ouyang et al., 2020) data. The dataset is available at https://echonet.github.io/dynamic/index.html; one has to sign a Research Use Agreement to access it. It is available for personal, non-commercial research purposes only. It prohibits distribution of portions of the data, which is why we do not show any frames from the echocardiograms in this paper. The dataset has labels (for example, end systolic volume), but these labels are not relevant to the task of

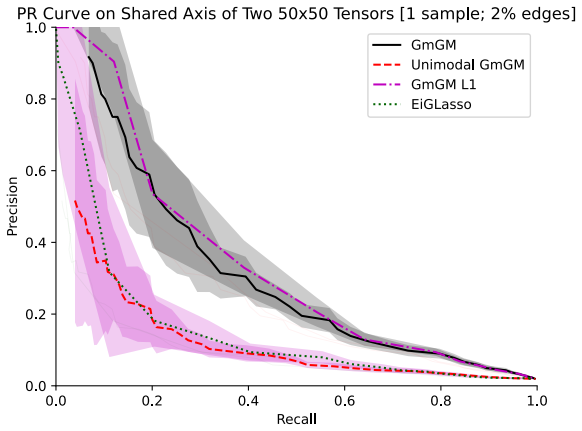
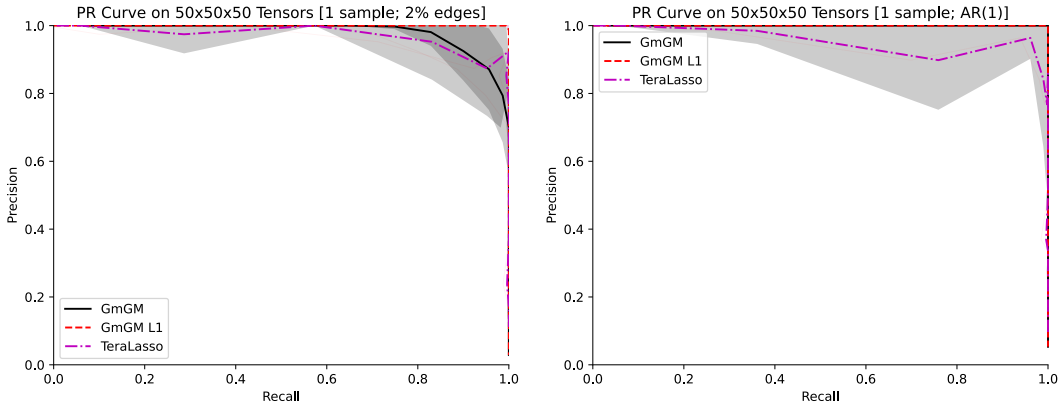


Figure A.3: Performance on two 50x50 datasets with one shared axis, whose true graphs were drawn from an Erdos-Renyi distribution (2% probability for each edge to exist, independently). Unimodal GmGM and EiGLasso only consider one of the two datasets.



(a) (b)

Figure A.4: Performance on tensor-variate (50x50x50) data whose true graphs were drawn from an Erdos-Renyi distribution (2% edge probability) and an AR(1) distribution.

predicting the frames the heartbeats occur on. For this, we picked the opening of the mitral valve as an arbitrary but easily visible occurrence, and manually recorded the frames of these openings for 5 videos in the dataset. These videos were picked at random.

Experiment justification

We chose the COIL duck video because of its use in prior work, but these methods were already known to perform quite well on it. Thus, we wanted to try a similar but harder task. Echocardiograms have a periodic structure (the heartbeat), and hence we would expect the learned graph of an echocardiogram to be similar to that of the duck video, with the addition of extra connections corresponding to the same phase of the heartbeat across different heartbeats. We saw that our algorithm was able to find this more complicated structure, and prove it by using that structure to detect future heartbeats.

Results

We downloaded all of the EchoNet-Dynamic (Ouyang et al., 2020) data. This dataset did not have heartbeats labeled, so we picked a few videos at random and labeled them ourselves as a proof of concept. Specifically, we labeled every frame in which the mitral valve opened. Our goal was to see if the graphs produced by our algorithm could predict this opening. Table A.4 contains the videos we picked, the labels we gave, and the labels we predicted. When we used the nonparanormal skeptic

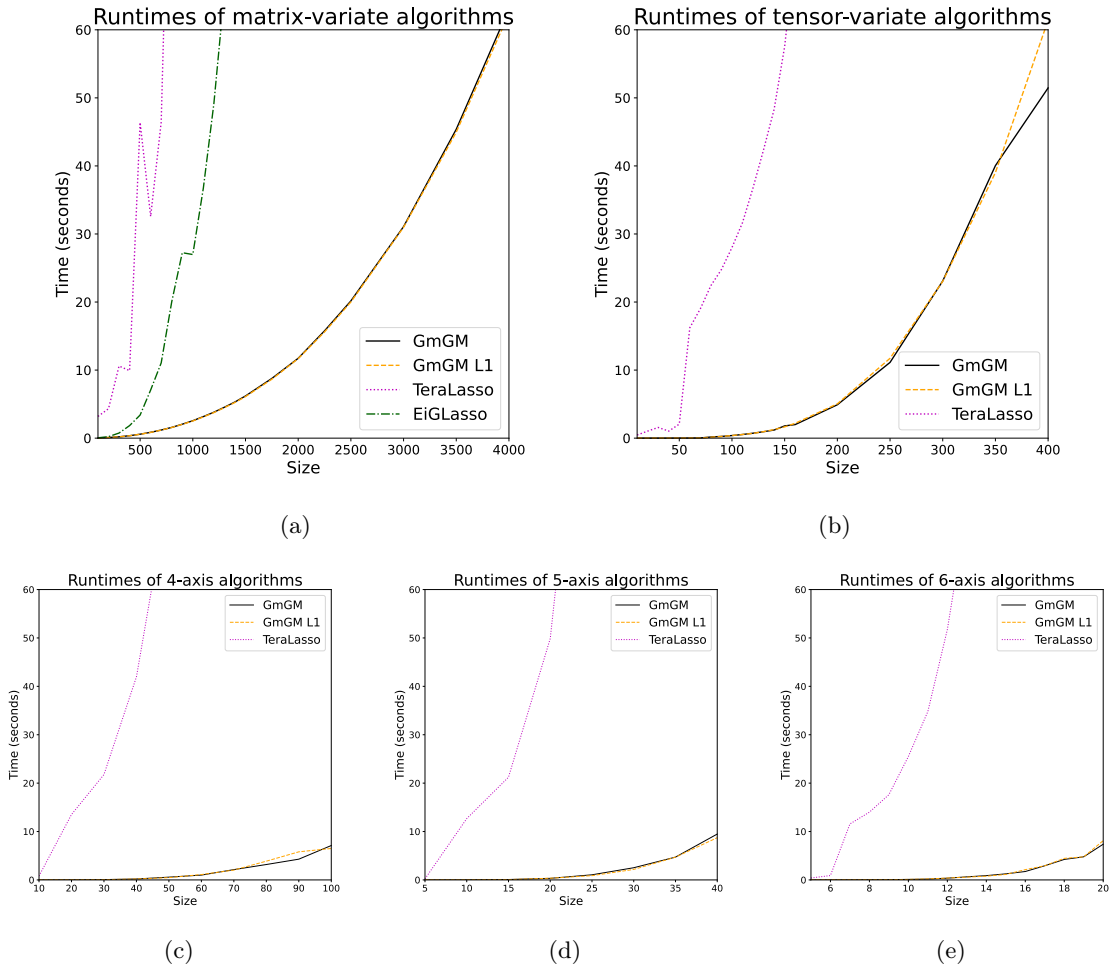


Figure A.5: Runtimes for 2-axis (a) 3-axis (b) 4-axis (c) 5-axis (d) and 6-axis (e) data, whose true graphs come from the Erdos-Renyi distribution with 2% edge weights.

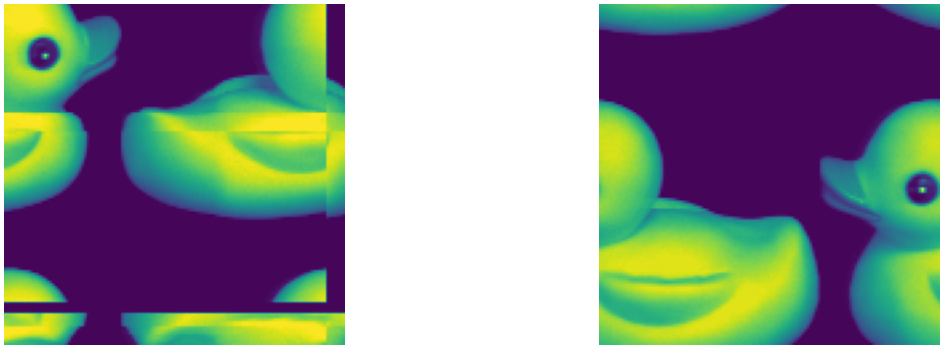
as a preprocessing step, we got broadly similar empirical results (Table A.5), although the precision matrices arguably more clearly show the periodic structure.

Mitral valve predictions were done by taking GmGM's output frames graph in precision matrix form, and measuring the mass along the diagonals. We treated this as a time series (since each diagonal corresponds to an increasing time offset from all frames simultaneously). We applied gaussian blur and then a continuous wavelet transform peak detection algorithm (Du et al., 2006) to find which diagonals had the most mass (Figure A.7). These represent the offsets corresponding with a heartbeat. Given the first mitral valve opening and these offsets, we predict the remaining openings. From Tables A.4 and A.5, we can see that our algorithm performs well. Our algorithm is much quicker than TeraLasso; see Table A.3.

A.4.4 Mouse embryo stem cell transcriptomics

The dataset

We used the mouse embryo stem cell dataset E-MTAB-2805 (Buettner et al., 2015), avaliable at https://www.ebi.ac.uk/biostudies/arrayexpress/studies/E-MTAB-2805, under a Creative Commons Zero license. This dataset is by what stage of the cell cycle each cell was in (G1, S, and G2M). We consider a subset of the genes available in this dataset, as that was the case in the scBiGLasso paper (Li et al., 2022); this subset is given in the text file on their github page for the algorithm.



(a) (b)

Figure A.6: A reconstruction of the COIL-20 duck video after shuffling the rows, columns, and frames, using GmGM. (a) without the nonparanormal skeptic. (b) with the nonparanormal skeptic.

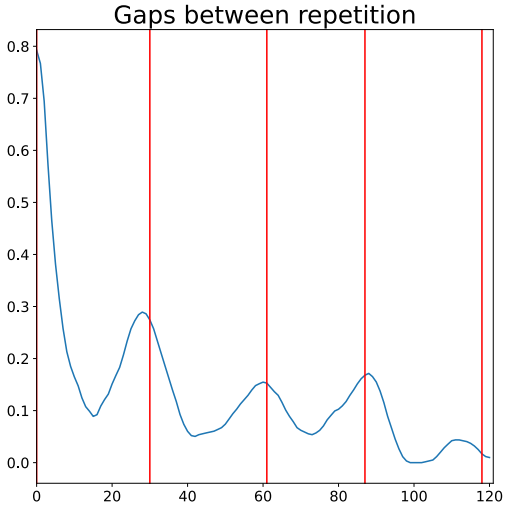


Figure A.7: An example heartbeat offset detection, from EchoNet-Dynamic video 0XFE6E32991136338. The blue curve represents our Gaussian-blurred diagonal mass (if x=10, it represents the blurred mass of the 10th diagonal to the right of the main diagonal). The red lines represent the predicted peaks via a continuous wavelet transform peak detection algorithm. These represent offsets from the first mitral valve opening. For this video, the mitral valve opened on frame 17 and our first offset was on the 30th diagonal. Hence, we would predict the second mitral valve opening to occur at frame 47 (which, in this case, was correct).

Algorithm	Preprocessing	Runtime
GmGM	Center Data	0.044 seconds
GmGM	Nonparanormal Skeptic	0.23 seconds
TeraLasso	Center Data	33 seconds
TeraLasso	Nonparanormal Skeptic	5.2 seconds

Table A.2: A comparison of the runtimes our algorithm and TeraLasso. Runtimes were averaged over 10 runs, and include the preprocessing.

Algorithm	Preprocessing	Runtime
GmGM	None	0.067 seconds
GmGM	Nonparanormal Skeptic	0.33 seconds
TeraLasso	None	40 seconds
TeraLasso	Nonparanormal Skeptic	12 seconds

Table A.3: Runtimes of our algorithm and TeraLasso, with various preprocessing methods, on the EchoNet-Dynamic dataset. Runtimes given are the average runtime over the five videos considered.

Experiment justification

We chose this dataset as it had been considered by multi-axis methods before (albeit in a non-quantitative way) and had a relatively clear task associated with it (cell cycle stage clustering).

Results - thresholding methods

We experimented with various thresholding methods on this dataset. One type of thresholding is global, in which one knocks out all edges whose magnitude is below a certain value. We found that this meant that clusters that tended to have high edge values were preserved relative to clusters who, while distinct, had a lower average edge value. One attempt to fix this was by thresholding per row; this has the interpretation of keeping the top n edges per vertex. Per-row thresholding mitigated this problem for some datasets (such as the 10x Genomics one), but not this dataset. The main problem was that some vertices tended to have higher edges than others - this meant that low-value vertices that were connected to a high-value vertex would always keep their connections to it, even if those connections were not very important from the perspective of the high-value vertex. To address this, we used a strategy of normalizing each column to sum to one before thresholding per row.

From Figure A.8, we can see that, with uncentered data, the third thresholding method is clearly better. In fact, without the nonparanormal skeptic, the other methods are practically worthless. However, if we centered the data as described in Section A.7, all methods perform similarly; the third is still best. This makes sense, as the model assumes zero-mean data. Performance doesn't change under the nonparanormal transformation if we center the data, as centering is a monotonic transformation. Notably, this means that the nonparanormal skeptic does not always yield the best performance. We can see this in Figure A.9.

Results - regularization

We wanted to check whether our restricted L1 regularizer would improve performance. To choose the best regularization parameter, we devised the following test:

1. Run GmGM with regularization parameter ρ.
2. Threshold according to the third method (normalize the columns, then pick the top n cells from each row). We let $n = 1$ as we wanted to optimize the parameter for the very sparse case.
3. Repeat for a range of values of ρ.
4. Measure the percentage of within-group connections. I.e, if 30% of the connections from cells in S stage were to other cells in S stage, we would report 30% for this metric.

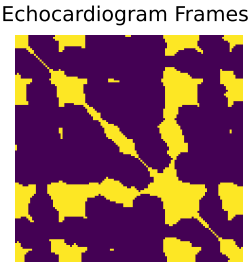
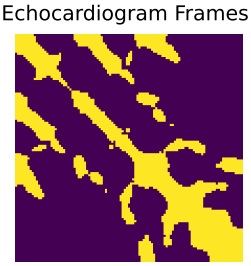
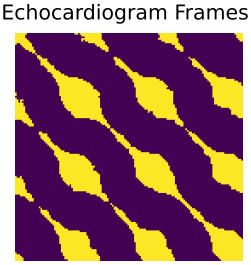
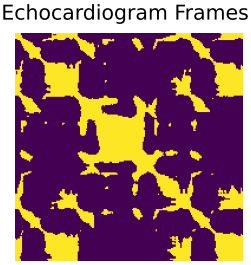
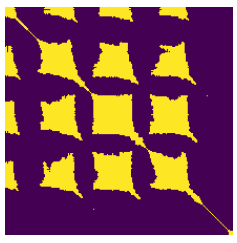
Video ID	Label	Predicted	Precision Matrix
0XFE6E32991136338	[17, 47, 77, 106]	[17, 47, 78, 104]	
0XF072F7A9791B060	[24, 56, 100]	[24, 59, 90]	
0XF70A3F712E03D87	[22, 66, 110]	[22, 67, 111]	
0XF60BBEC9C303C98	[19, 67, 114, 162]	[19, 66, 115, 162]	
0XF46CF63A2A1FA90	[25, 79, 134, 188]	[25, 80, 133, 184]	

Table A.4: Mitral valve labelings and precision matrices for the EchoNet-Dynamic dataset. The precision matrices, for the most part, seem to have clear off-diagonal structures, as expected, and the mitral valve prediction is generally quite good; it is only significantly off for the last opening in 0XF072F7A9791B060. The precision matrices are shown with the top 20% of the edges kept.

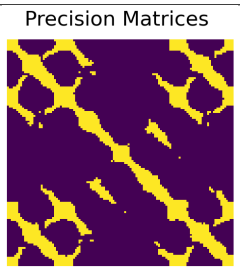
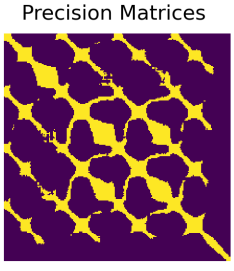
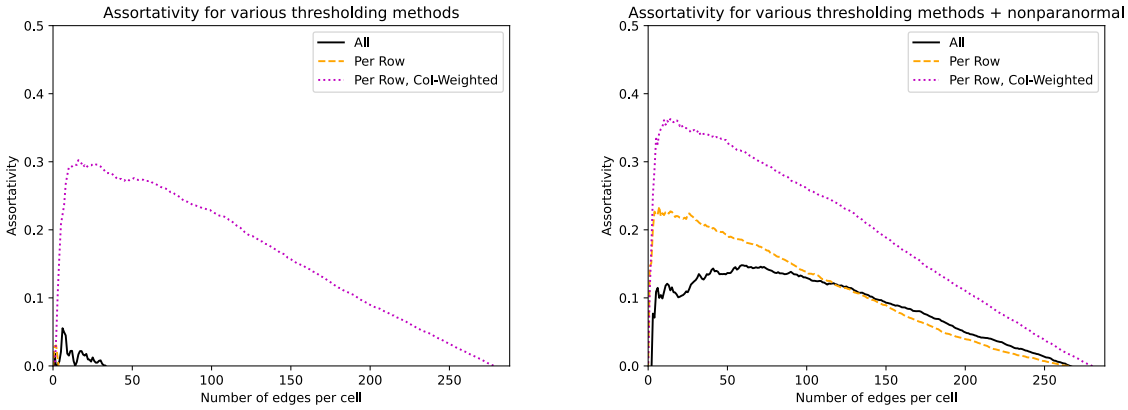
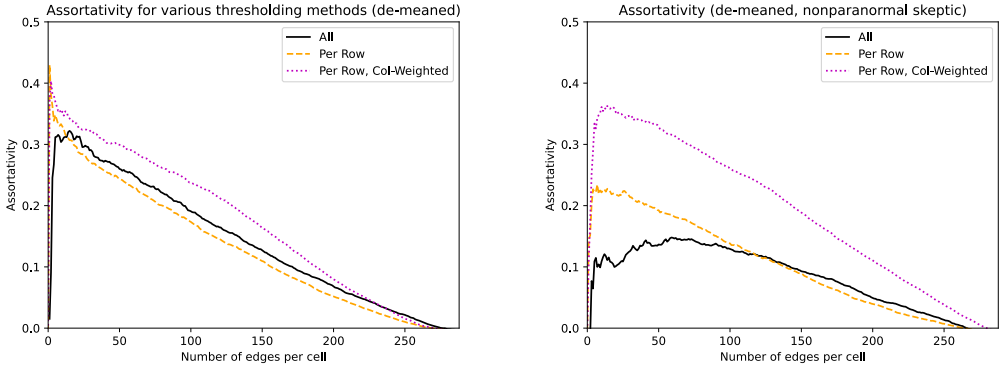
Video ID	Label	Predicted	Precision Matrix
0XFE6E32991136338	[17, 47, 77, 106]	[17, 48, 77, 106]	
0XF072F7A9791B060	[24, 56, 100]	[24, 58, 91]	
0XF70A3F712E03D87	[22, 66, 110]	[22, 67, 112]	
0XF60BBEC9C303C98	[19, 67, 114, 162]	[19, 66, 116, 162]	
0XF46CF63A2A1FA90	[25, 79, 134, 188]	[25, 80, 134, 188]	

Table A.5: Mitral valve labelings and precision matrices for the EchoNet-Dynamic dataset, using the nonparanormal skeptic to preprocess. The precision matrices, for the most part, seem to have clear off-diagonal structures, as expected, and the mitral valve prediction is generally quite good; it is only significantly off for the last opening in 0XF072F7A9791B060. The precision matrices are shown with the top 20% of the edges kept.



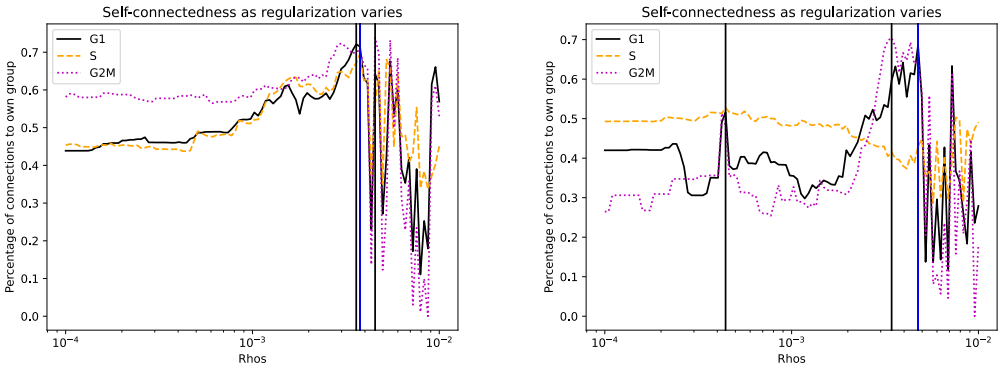
(a) (b)

Figure A.8: (a) Various thresholding methods with log-transformed data. (b) Various thresholding methods with nonparanormal-skeptic-transformed data. In both cases, we did not center the data.



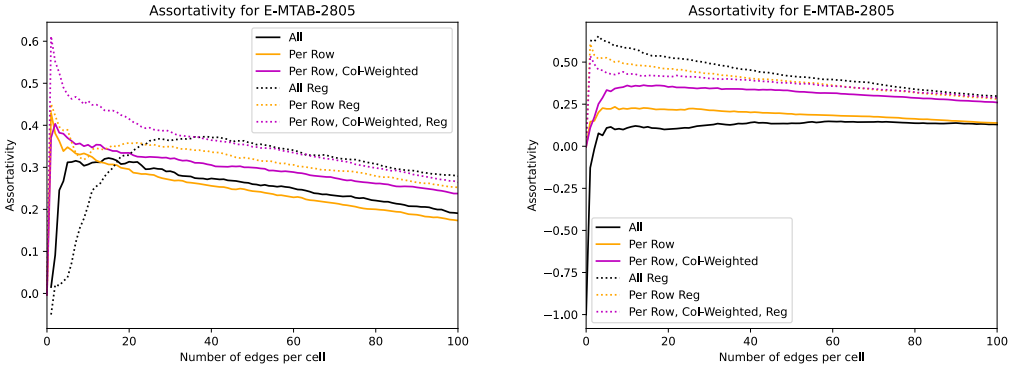
(a) (b)

Figure A.9: Assortativity of the various methods once we center the data. (a) log-transformed (b) nonparanormal skeptic.



(a) (b)

Figure A.10: The percentage of within-group connections, varied as the regularization parameter increases. (a) with a log transform (b) with the nonparanormal skeptic. Vertical lines represent the best values for each class; the blue vertical line represents the best value overall.



(a) (b)

Figure A.11: Assortativity for GmGM L1, (a) with a log transform and (b) with the nonparanormal skeptic. Note that the scales between the two plots are different.

5. Pick the ρ that optimizes the average percentage of within-group connections over all groups. Run GmGM with that ρ.

6. Calculate the assortativities over all thresholding methods.

We chose this setup as it is a distinct but related task to assortativity. Its results can be seen in Figure A.10. The best parameters were around 0.0038 for the log-transformed data, and 0.0048 when using the nonparanormal skeptic. Figure A.11 demonstrates the advantage of regularization.

Results - comparison with EiGLasso

We compare the assortativities of the best GmGM L1 parameters we discovered in the last section with the assortativities of EiGLasso as we varied its regularization parameter. We found a surprisingly clear advantage when using our method. This was unexpected, as our method and EiGLasso are quite similar, differing only in how we enforce sparsity. Judging by the 'chink' in EiGLasso's curve (Figure A.12, it could be that EiGLasso simply needed more time to converge to a good result. However, generating this curve for EiGLasso already took a long time, so we did not investigate further. To compare runtimes, see Table A.6. We did not perform a UMAP consistency plot, as cells in UMAP-space formed a homogenous blob.

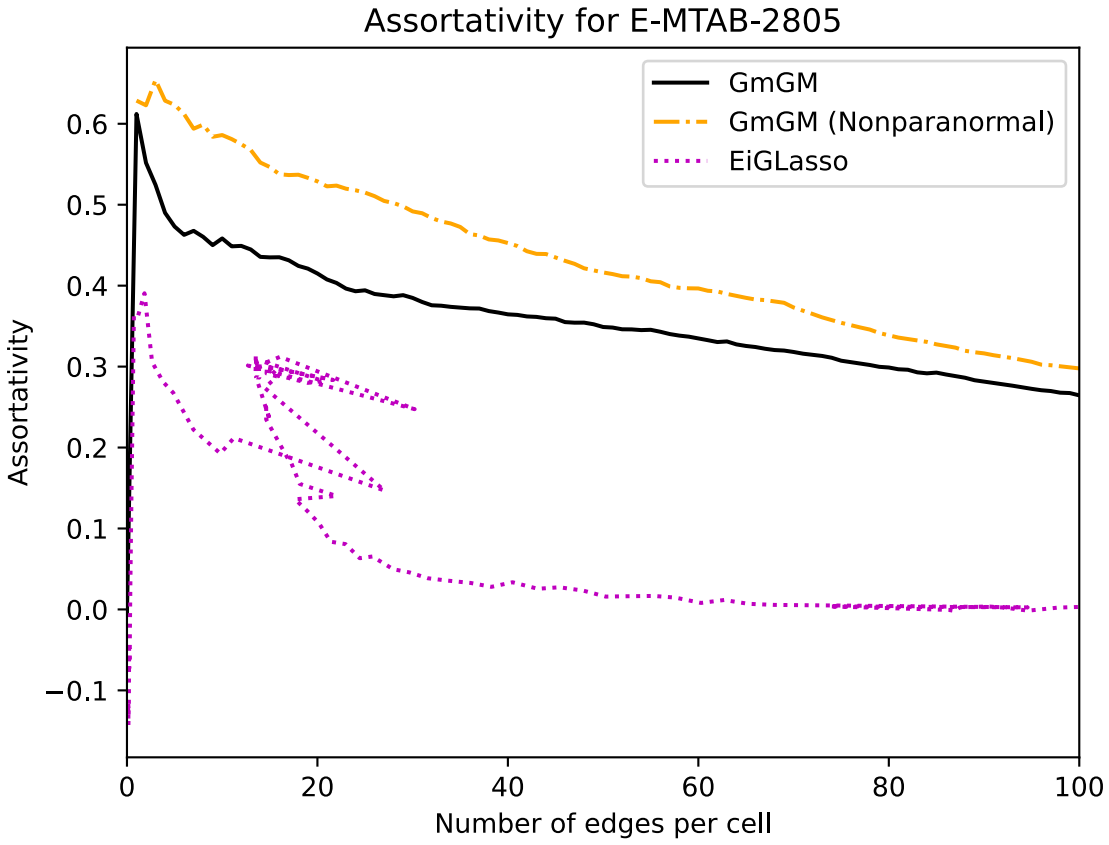


Figure A.12: A comparison of GmGM with and without the nonparanormal skeptic, and EiGLasso. As EiGLasso was much slower than GmGM, we did not rerun the calculation for it with the nonparanormal skeptic.

Algorithm	Preprocessing	Runtime
GmGM	Log Transform, Centered	0.0079 seconds
GmGM	Nonparanormal Skeptic	0.043 seconds
EiGLasso	Log Transform, Centered	30 seconds
EiGLasso	Nonparanormal Skeptic	108 seconds

Table A.6: Runtimes of our algorithm and EiGLasso, with various preprocessing techniques, on the E-MTAB-2805 dataset. Runtimes given are an average over 10 runs.

A.4.5 LifeLines-DEEP metagenomics + metabolomics

The dataset

We used the LifeLines-DEEP metagenomics and metabolomics datasets (Tigchelaar et al., 2015). We did not do any pre-processing to the metabolomics, and we used the already pre-processed version of the metagenomics data from Prost et al. (2021). This dataset is available on the European Genome-Phenome Archive (EGA) under Study ID EGAS00001001704; to access it, one has to agree to an LL-DEEP-specific data access agreement.

Experiment justification

We chose this dataset as it had been considered by prior single-axis work (ZiLN; Prost et al. (2021)), as well as being multi-omic. It had a more complicated class structure (taxonomy) than the E-MTAB-2805 dataset, so it was a natural next step in our investigations of the performance of this algorithm.

Results

We kept only patients that appeared in both datasets. We compared our model's results to the model given by Prost et al. (2021) in the main paper. We report runtimes in Table A.7, and give a UMAP consistency plot as in Figure A.13. We can see that our algorithm is the fastest on the metagenomics dataset, even outperforming a single-axis method. Furthermore, our results seem quite sensible in UMAP-space.

Finally, we compared our assortativity to that of the Zero-Inflated Log Normal model by Prost et al. (2021). We found that our method performed similarly to theirs; see Figure A.14.

Finally, we experimented with incorporating priors in our model. It is not unreasonable to find oneself in a situation in which one knows the highest-level taxonomic categorization of an organism (its phylum), but not know its lower-level categorizations. Thus, we chose a Wishart prior whose covariance matrix encoded the adjacency matrix of the following graph:

1. If two species are in the same phylum, they are connected
2. If two species are in a different phylum, they are not connected.

This *vastly* improves performance (Figure A.15a). One might worry that the algorithm is just 'memorizing' the prior, and that the lower taxonomic ranks are improving because no genuses are cross-phylum, for example. To show that this is not the case, we conduct a quick experiment with synthetic data, in which we feed the true graph in as the prior. If our algorithm were memorizing priors, then we would expect performance with the prior to be perfect. However, what we instead see is a huge but imperfect gain in performance (Figure A.15b).

Algorithm	Axes	Speed	Per-Axis Speed
ZiLN	Species	3.2	3.2
GmGM	Species, People	2.59	1.30
GmGM	Species, People, Metabolites	22.18	7.39
TeraLasso	Species, People	1299.33	649.67

Table A.7: Runtimes of various algorithms on the LifeLines-DEEP dataset. Speed is measured in seconds.

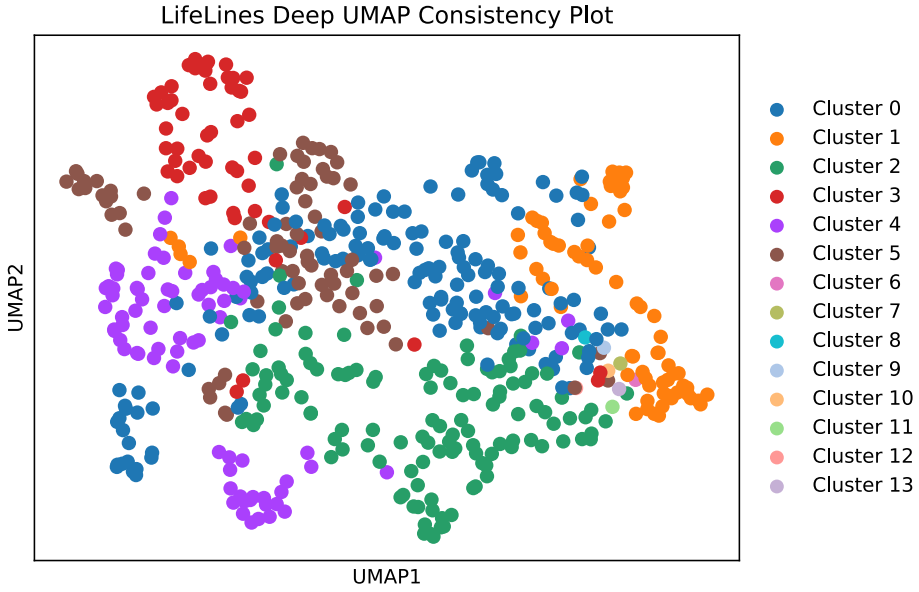


Figure A.13: UMAP consistency plot for our algorithm applied to the LifeLines-DEEP dataset. The UMAP was constructed from log-transformed metagenomics data; the coloring represents a Louvain clustering of the output of GmGM performed on the centered log-transformed metagenomics+metabolomics data, thresholded by keeping the top 7% of edges overall.

Algorithm	Preprocessing	Runtime
GmGM	Log transform, just RNA	52 seconds
GmGM	Log transform, just ATAC	607 seconds
GmGM	Log transform, both modalities	590 seconds
EiGLasso	Log transform, just RNA	>60,000 seconds

Table A.8: Runtimes of our algorithm and EiGLasso, with various preprocessing techniques, on the 10x Genomics data.

A.4.6 10x Genomics flash frozen lymph nodes

The dataset

For this experiment, we looked at a single-cell RNA-sequencing+ATAC-sequencing dataset from 10x Genomics (10x Genomics, 2021). It is available here. The dataset is publicly available, but one does need to fill in a brief form for data collection purposes. They do not specify a license.

Experiment justification

It is a realistically-sized multi-omics dataset, and by far the largest considered in this paper. It is so large that the use of prior work becomes infeasible, so demonstrating that our algorithm works on it would represent a significant leap forward. The dataset is unlabeled, so we evaluated our algorithm by means of a UMAP consistency plot.

Results

Before performing the experiment, we removed cells whose library size was three median absolute deviations from the median, and similarly removed genes and peaks if the the total amount of times they were expressed was three median absolute deviations from the median. In our output graphs, we kept the top 5 edges per vertex. We can see the UMAP consistency plot in Figure 2.8b and the runtimes in Table A.8.

In Figure 2.8b, we can see that clusters 0, 1, 3, and 4 all inhabit the large contiguous region in the lower left. Clusters 2 and 7 inhabit the contiguous region in the lower right. In the top right, there are

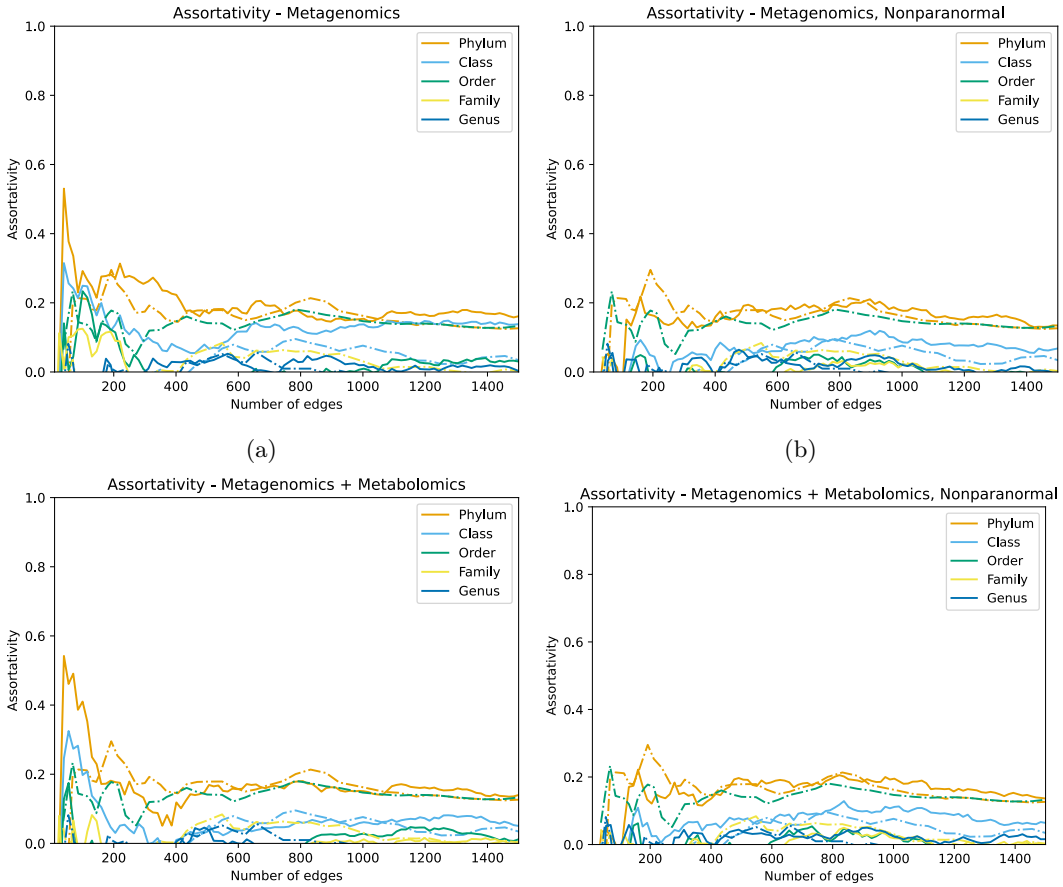


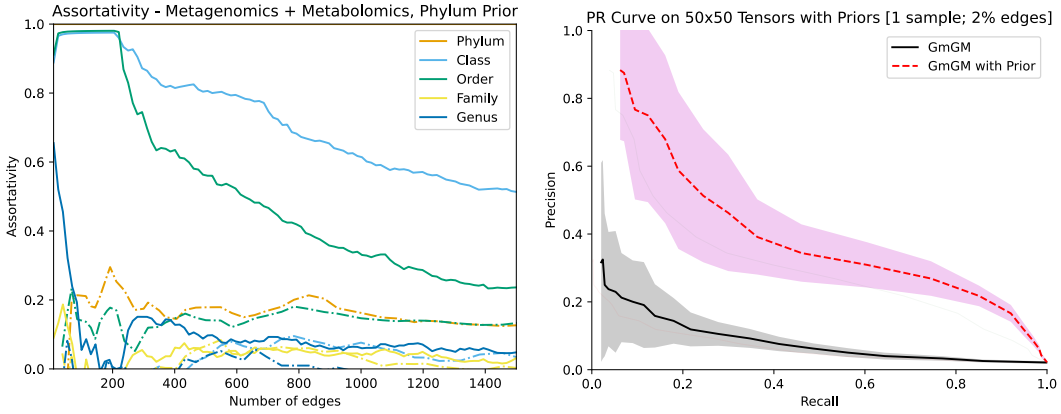
Figure A.14: Comparison of our method (solid lines) to that of Prost et al. (2021) (dashed lines). (a) Just metagenomics, log-transformed. (b) Metagenomics using the nonparanormal skeptic. (c) The full multi-omic dataset, log-transformed. (d) The full multi-omic dataset using the nonparanormal skeptic.

three regions; 5, 6, and 8 (in order of left to right). Figure A.20 implies that clusters 5 and 6 have some similarities, whereas cluster 8 is largely different from all the other clusters. A GO term analysis shows cluster 8 is associated with blood coagulation and the integrin signalling pathway, whereas none of the other clusters have such an association. All other clusters, except 2 and 7, are highly associated with B cell activation-related genes. Cluster 2 is distinct in that it is related to the CCKR signalling map and the apoptosis signalling pathways. Cluster 7 was too small to perform the analysis. Full details of the significant go terms are available on the github repository.

A.5 Regularization

As remarked in the main paper, our algorithm by default includes no regularization. This is because our algorithm leverages the fact that we have a closed-form expression for the eigenvectors of the maximum likelihood estimate to avoid costly eigendecompositions every iteration. We do not have a closed-form expression for the eigenvectors in the regularized case.

Nevertheless, we can add regularization to the eigenvalue estimation by performing an eigenrecomposition and regularizing that. Eigenrecomposition requires a matrix multiplication, which is quite costly compared to the cost of an unregularized iteration - both in practice, and asymptotically in the matrix-variate case (matrix multiplication is $O(\sum_\ell d_\ell^3)$ whereas an unregularized iteration is $O(\prod_\ell d_\ell)$). Thus, to regularize we first let our algorithm converge to the MLE before considering the penalty term. This allows us to avoid a major increase in runtime; our regularized algorithm runs in roughly the same time as the unregularized one (Figure A.21a).



(a) (b)

Figure A.15: (a) The assortativity when incorporating prior knowledge. (b) A test on 50x50 1-sample synthetic data with 2% edges connected, with the true graphs fed in as the parameter to a Wishart prior to our algorithm.

It is important to note that this estimator is slightly different than the standard Lasso estimator, as the standard estimator would minimize $\|\mathbf{\Psi}_\ell\|_1$ and our estimator minimizes $\|\hat{\mathbf{V}}_\ell \mathbf{\Lambda}_\ell \hat{\mathbf{V}}_\ell^T\|_1$ (where only the eigenvalues $\mathbf{\Lambda}_\ell$ are free to vary). This restriction prevents it from being able to drive elements exactly to zero, so thresholding is still needed afterwards. It can be derived as follows:

$$\frac{\partial}{\partial \lambda_i}\|\mathbf{V}\mathbf{\Lambda}\mathbf{V}^T\|_1 = \frac{\partial}{\partial \lambda_i}\|\sum_j \lambda_j v_{ja} v_{bj}\|_1 \tag{A.55}$$

$$= \left[\frac{\partial}{\partial \lambda_i}\left|\sum_j \lambda_j v_{ja} v_{bj}\right|\right]_{ab} \tag{A.56}$$

$$= \left[\frac{\partial}{\partial \lambda_i}\text{sign}\left[\sum_j \lambda_j v_{ja} v_{bj}\right] v_{ia} v_{bi}\right]_{ab} \tag{A.57}$$

$$= \left[\text{sign}\left[\mathbf{V}\mathbf{\Lambda}\mathbf{V}^T\right]_{ab} v_{ia} v_{bi}\right]_{ab} \tag{A.58}$$

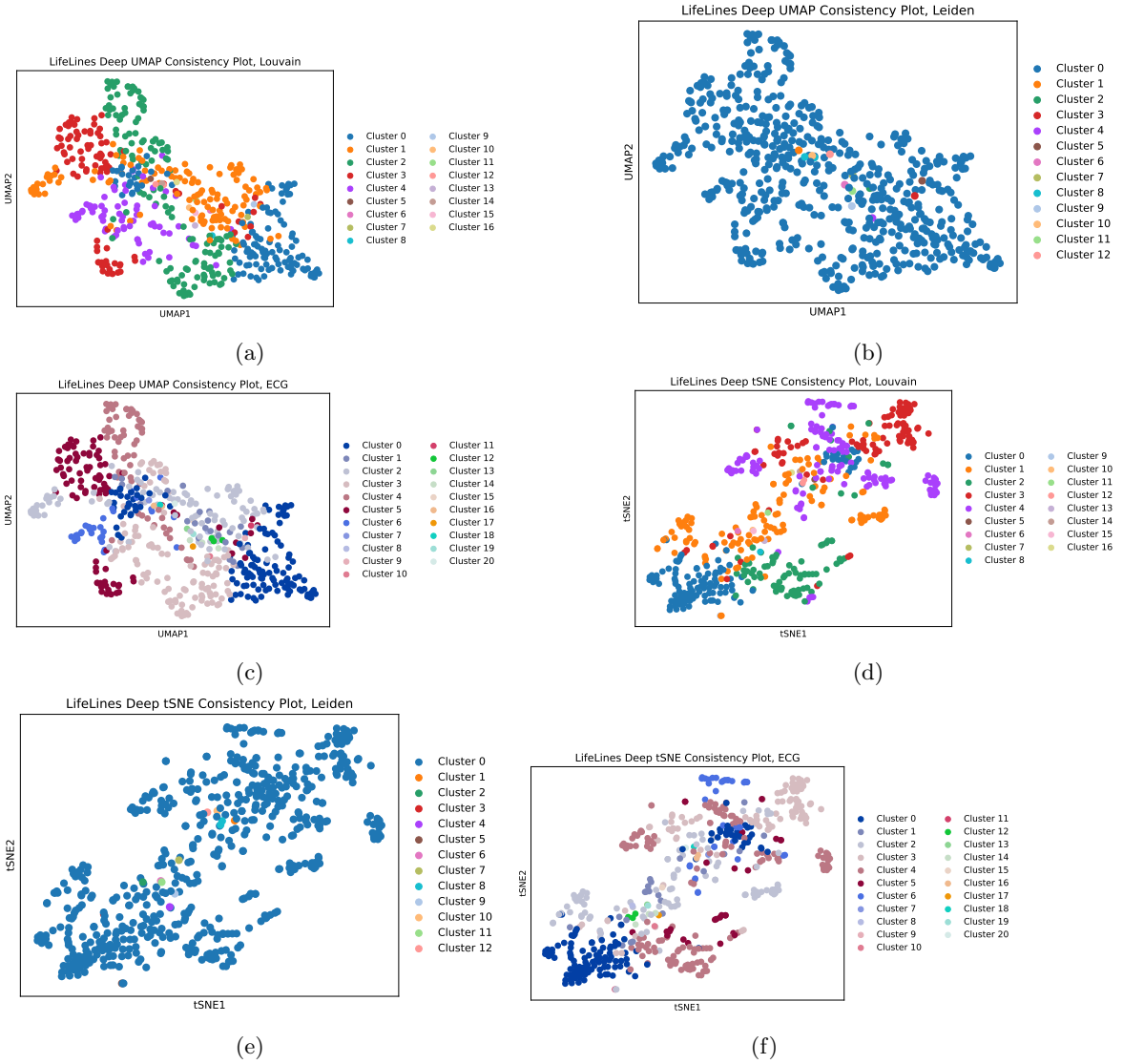
$$= \mathbf{v}_i^T \text{sign}\left[\mathbf{V}\mathbf{\Lambda}\mathbf{V}^T\right]\mathbf{v}_i \tag{A.59}$$

Despite this difference, it performs comparably to prior work. We show in Figure A.21b the precision-recall curves for the 3-axis case, and observe that it performs almost perfectly. This is notable as it was the case that the unregularized algorithm performed worse than prior work.

To test whether regularization improved the algorithm in the real world, we performed another test on the E-MTAB-2805 dataset. We first looked at the metric of 'percentage of connections to own group'. We varied the regularization parameter, and for each estimated graph we thresholded the graph to keep only the largest edge per cell. We plotted the results in Figure A.22a.

The optimal parameter arises right near the end of the range of considered regularization parameters; when we considered larger parameters, we entered a region of instability in which convergence took far longer and results became sporadic (Figure A.22b). We are not sure what causes this, but hypothesize that our restricted regularizer no longer has the degrees of freedom to give useful information to the problem, rather making it harder to find a good solution. The fact that the optimal parameters occur specifically at the end of the stable region implies that the algorithm would likely have benefited from a non-restricted regularizer - although our restricted regularizer is also definitively better than no regularizer at all.

Our second test, shown in Figure A.23, was to take the optimal regularing parameter found in the



(a) (b) (c) (d) (e) (f)

Figure A.16: UMAP and tSNE consistency plots with Louvain, Leiden, and ECG clustering algorithms on the LifeLines-Deep data.

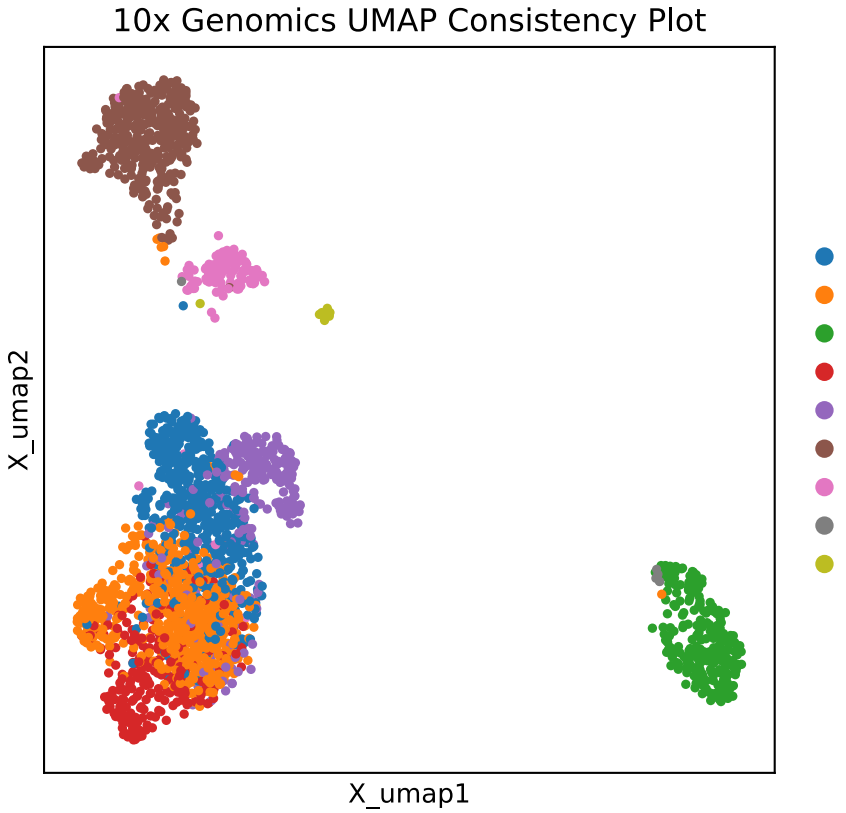
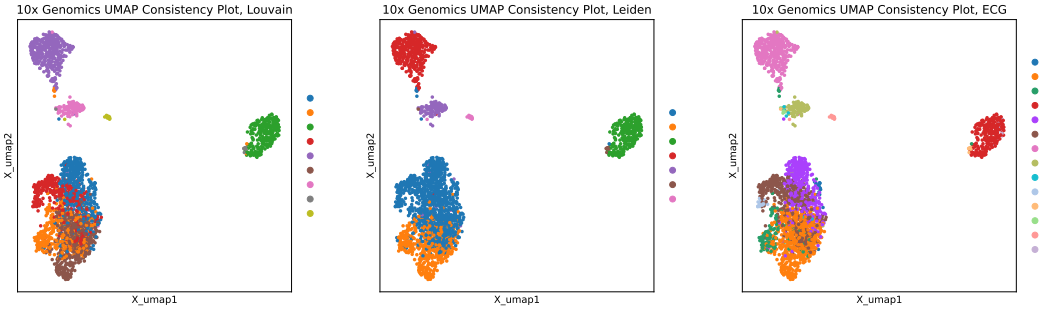
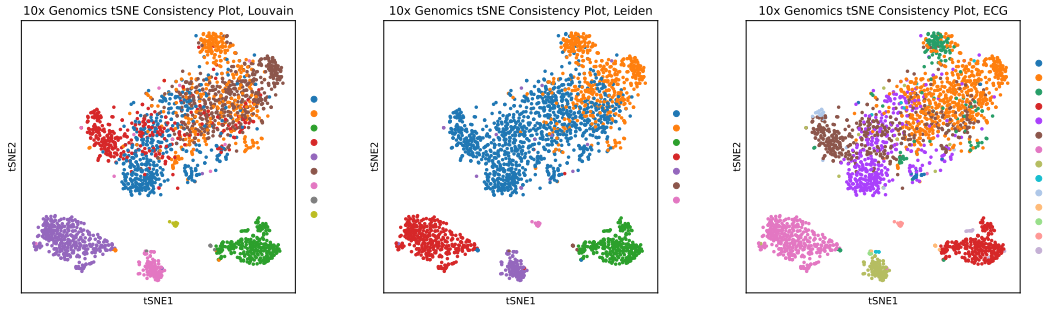


Figure A.17: A UMAP consistency plot for the 10x Genomics dataset. Both UMAP and GmGM were used on the full multi-omic log-transformed dataset. The coloring represents a Louvain clustering on the output graph of GmGM, with the top 5 edges per vertex being kept after column normalization (the third thresholding method from Section A.4.4).



(a) (b) (c)

Figure A.18: UMAP consistency plots with Louvain, Leiden, and ECG clustering algorithms on the 10x Genomics data.



(a) (b) (c)

Figure A.19: tSNE consistency plots with Louvain, Leiden, and ECG clustering algorithms on the 10x Genomics data.

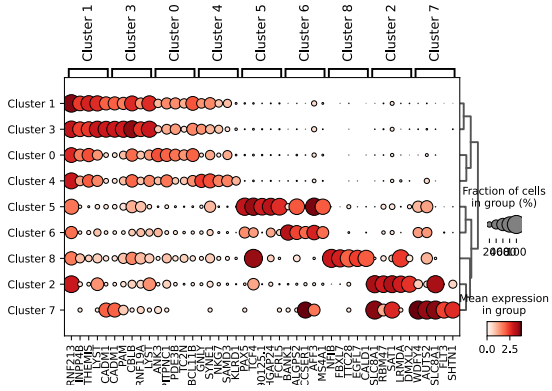
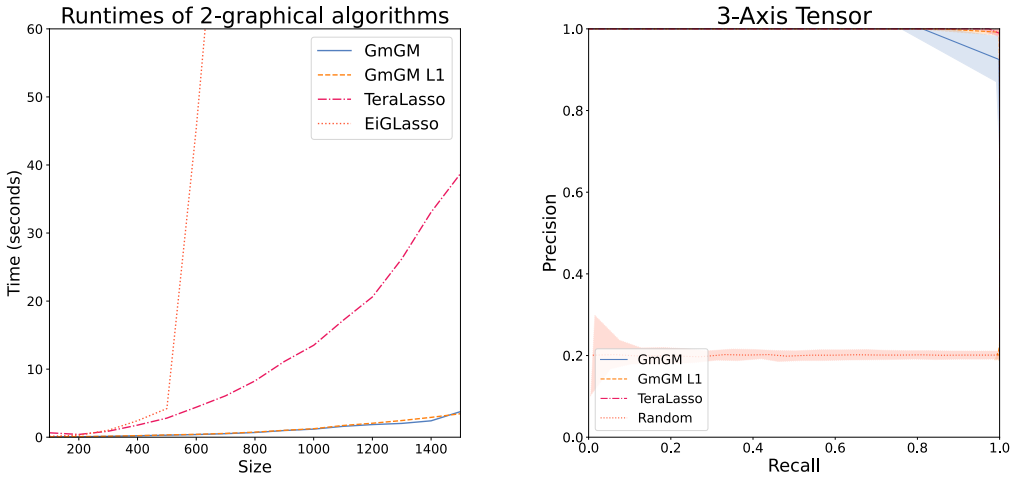
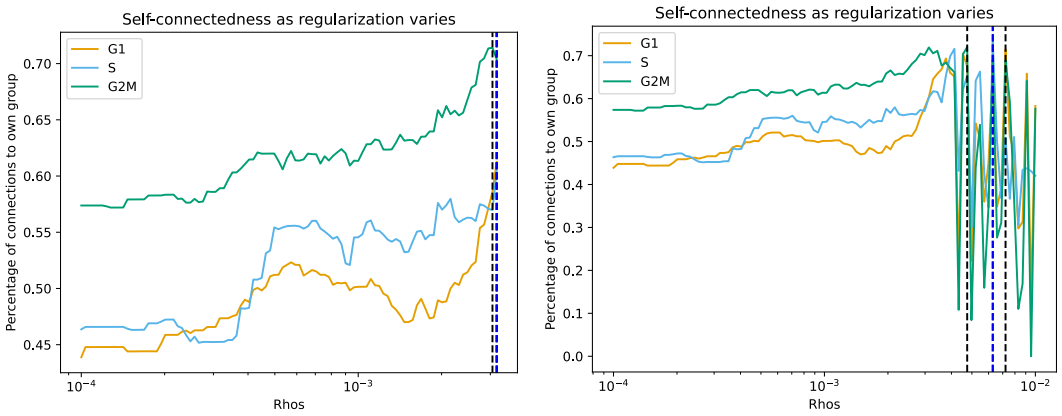


Figure A.20: Expression of top differentially-expressed genes by cluster.



(a) (b)

Figure A.21: (a) Runtimes of our algorithm and prior work on matrix-variate data. Our regularized algorithm is denoted "GmGM L1", and takes about the same time as the unregularized "GmGM". (b) Precision-recall curves for tensor-variate data. TeraLasso and our regularized "GmGM L1" perform almost perfectly.



(a) (b)

Figure A.22: (a) Self-connectedness of each of the three cell cycle stages as regularization parameter varies. Vertical bars represent the optimal parameters; one for each of the parameters (there is overlap). (b) The same plot as on the left, with the range extended to show the region of instability.

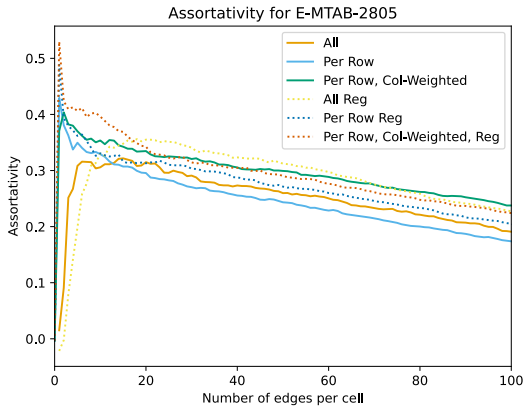


Figure A.23: Assortativity as the number of kept edges per cell varies.

first test, and perform the same assortativity test as was performed in Section A.4.4. We can see that the regularized methods tend to outperform the unregularized methods.

A.6 Asymptotic complexity

Our space complexity is the optimal $O(\sum_\ell d_\ell^2)$, as was all prior work except the original BiGLasso. For computational complexity, refer to Table A.9. In the reported values, we treat matrix multiplication as having cubic complexity. Note that Gram matrix computation is an $O(Kd^{K+1})$ operation, and that our algorithm has the asymptotically fastest iterations in the matrix-variate case ($O(d^2)$). If we ignore the per-axis sizes d and instead consider the size of the whole dataset $d_\forall = d^K$, we can note that the runtime of the tensor-variate algorithms is almost linear in the size of the dataset.

A.7 Centering the data

Multi-axis models such as ours typically assume a mean of 0 for the data. When one has multiple samples, this is not a problem - one can just center the data. However, with multi-axis methods we typically only have one sample. Because the nonparanormal skeptic is invariant to monotone transformations, this is a viable method to circumvent this problem. When not using that, we would

Algorithm	Range of K	Overall Complexity	Per-Iteration Complexity
BiGLasso	$K = 2$	$O(Kd^4)$	$O(Kd^4)$
EiGLasso	$K = 2$	$O(Kd^3)$	$O(Kd^3)$
TeraLasso	Any	$O(Kd^{K+1})$	$O(Kd^3 + d^K)$
GmGM	Any	$O(Kd^{K+1})$	$O(d^K)$
GmGM L1	Any	$O(Kd^{K+1})$	$O(Kd^3 + d^K)$

Table A.9: Computational complexities of selected algorithms. 'GmGM L1' refers to GmGM equipped with our restricted L1 penalty; note that GmGM L1 first converges to the unregularized solution (using the cheap iterations of GmGM) before it adds in the regularizer. For simplicity, all dimensions d_ℓ are considered to be the same size d.

take the average value of the tensor (averaged across all elements in the tensor) and subtract this value from ever element in the tensor.

Bibliography for Paper 1 (appendix)

10x Genomics. (2021, May). Flash-Frozen Lymph Node with B Cell Lymphoma (14k sorted nuclei). Retrieved May 10, 2023, from https://www.10xgenomics.com/resources/datasets/fresh-frozen-lymph-node-with-b-cell-lymphoma-14-k-sorted-nuclei-1-standard-2-0-0

Buettner, F., Natarajan, K. N., Casale, F. P., Proserpio, V., Scialdone, A., Theis, F. J., Teichmann, S. A., Marioni, J. C., & Stegle, O. (2015). Computational analysis of cell-to-cell heterogeneity in single-cell RNA-sequencing data reveals hidden subpopulations of cells [Number: 2 Publisher: Nature Publishing Group]. *Nature Biotechnology*, *33*(2), 155–160. https://doi.org/10.1038/nbt.3102

Dahl, A., Hore, V., Iotchkova, V., & Marchini, J. (2013, December). Network inference in matrix-variate Gaussian models with non-independent noise [arXiv:1312.1622 [stat]]. https://doi.org/10.48550/arXiv.1312.1622

Du, P., Kibbe, W. A., & Lin, S. M. (2006). Improved peak detection in mass spectrum by incorporating continuous wavelet transform-based pattern matching. *Bioinformatics*, *22*(17), 2059–2065. https://doi.org/10.1093/bioinformatics/btl355

Greenewald, K., Zhou, S., & Hero III, A. (2019, September). Tensor Graphical Lasso (TeraLasso) [arXiv:1705.03983 [stat]]. Retrieved February 24, 2023, from http://arxiv.org/abs/1705.03983

Kalaitzis, A., Lafferty, J., Lawrence, N. D., & Zhou, S. (2013). The Bigraphical Lasso [ISSN: 1938-7228]. *Proceedings of the 30th International Conference on Machine Learning*, 1229–1237. Retrieved February 24, 2023, from https://proceedings.mlr.press/v28/kalaitzis13.html

Kolda, T. G., & Bader, B. W. (2009). Tensor Decompositions and Applications. *SIAM Review*, *51*(3), 455–500. https://doi.org/10.1137/07070111X

Lewis, A. S. (1996). Derivatives of Spectral Functions [Publisher: INFORMS]. *Mathematics of Operations Research*, *21*(3), 576–588. https://doi.org/10.1287/moor.21.3.576

Li, S., López-García, M., Lawrence, N. D., & Cutillo, L. (2022, March). Scalable Bigraphical Lasso: Two-way Sparse Network Inference for Count Data [arXiv:2203.07912 [cs, stat]]. Retrieved February 24, 2023, from http://arxiv.org/abs/2203.07912

Mazumder, R., & Hastie, T. (2012). Exact Covariance Thresholding into Connected Components for Large-Scale Graphical Lasso. *Journal of machine learning research : JMLR*, *13*, 781–794. Retrieved October 9, 2023, from https://www.ncbi.nlm.nih.gov/pmc/articles/PMC4225650/

Nene, S. A., Nayar, S. K., & Murase, H. (n.d.). Columbia Object Image Library (COIL-20).

Ouyang, D., He, B., Ghorbani, A., Yuan, N., Ebinger, J., Langlotz, C. P., Heidenreich, P. A., Harrington, R. A., Liang, D. H., Ashley, E. A., & Zou, J. Y. (2020). Video-based AI for beat-to-beat assessment of cardiac function [Number: 7802 Publisher: Nature Publishing Group]. *Nature*, *580*(7802), 252–256. https://doi.org/10.1038/s41586-020-2145-8

Prost, V., Gazut, S., & Brüls, T. (2021). A zero inflated log-normal model for inference of sparse microbial association networks [Publisher: Public Library of Science]. *PLOS Computational Biology*, *17*(6), e1009089. https://doi.org/10.1371/journal.pcbi.1009089

Tigchelaar, E. F., Zhernakova, A., Dekens, J. A. M., Hermes, G., Baranska, A., Mujagic, Z., Swertz, M. A., Muñoz, A. M., Deelen, P., Cénit, M. C., Franke, L., Scholtens, S., Stolk, R. P., Wijmenga, C., & Feskens, E. J. M. (2015). Cohort profile: LifeLines DEEP, a prospective, general population cohort study in the northern Netherlands: Study design and baseline characteristics [Publisher: British Medical Journal Publishing Group Section: Epidemiology]. *BMJ Open*, *5*(8), e006772. https://doi.org/10.1136/bmjopen-2014-006772

Appendix B

Gaussian graphical modelling without independence assumptions for uncentered data

Graphical Modelling without Independence Assumptions for Uncentered Data

Supplementary material

B.1 Deferred proofs

In this portion of the appendix, we will formally prove the claims presented in the paper, namely we will prove that the locations of coordinate-wise minima reported in the paper are indeed accurate, and furthermore that the full optimization problem has a unique solution.

B.1.1 Derivation of gradients

First, we will derive gradients with respect to all parameters. Recall that the negative log-likelihood is, up to an additive constant:

$$\mathrm{NLL}\left(\mathcal{D}\right) \propto \frac{-1}{2}\log\left|\boldsymbol{\Omega}\right| + \frac{1}{2}\left(\boldsymbol{x}-\boldsymbol{\omega}\right)^T\boldsymbol{\Omega}\left(\boldsymbol{x}-\boldsymbol{\omega}\right)$$

We do not need to work out the gradient w.r.t. $\boldsymbol{\Psi}_\ell$, as we defer that optimization task to algorithms such as TeraLasso. Thus, we are left with the following quadratic function:

$$\mathrm{NLL}\left(\mathcal{D}\right) \propto \frac{1}{2}\left(\boldsymbol{x}-\boldsymbol{\omega}\right)^T\boldsymbol{\Omega}\left(\boldsymbol{x}-\boldsymbol{\omega}\right)$$

We will differentiate with respect to the identifiable parameters $\widetilde{\boldsymbol{\mu}}_\ell, m$. Generically, for any parameter p, the derivative will look like:

$$\frac{\partial}{\partial p}\mathrm{NLL}\left(\mathcal{D}\right) = -\left(\boldsymbol{x}-\boldsymbol{\omega}\right)^T\boldsymbol{\Omega}\left(\frac{\partial}{\partial p}\boldsymbol{\omega}\right)$$

Oberve that:

$$\begin{aligned}
\frac{\partial}{\partial m}\boldsymbol{\omega} &= \frac{\partial}{\partial m}\left(m\mathbf{1}_{d_\forall} + \widetilde{\bigoplus}_\ell\widetilde{\boldsymbol{\mu}}_\ell\right)\\
&= \mathbf{1}_{d_\forall}\\
\frac{\partial}{\partial\widetilde{\boldsymbol{\mu}}_\ell}\boldsymbol{\omega} &= \frac{\partial}{\partial\widetilde{\boldsymbol{\mu}}_\ell}\widetilde{\bigoplus}_\ell\widetilde{\boldsymbol{\mu}}_\ell\\
&= \frac{\partial}{\partial\widetilde{\boldsymbol{\mu}}_\ell}\left(\widetilde{\boldsymbol{\mu}}_\ell\widetilde{\oplus}\widetilde{\boldsymbol{\mu}}_{\setminus\ell}\right) && \text{(up to permutation)}\\
&= \frac{\partial}{\partial\widetilde{\boldsymbol{\mu}}_\ell}\left(\widetilde{\boldsymbol{\mu}}_\ell\otimes\mathbf{1}_{d_{\setminus\ell}} + \mathbf{1}_{d_\ell}\otimes\widetilde{\boldsymbol{\mu}}_{\setminus\ell}\right)\\
&= \mathbf{I}_{d_\ell}\otimes\mathbf{1}_{d_{\setminus\ell}}
\end{aligned}$$

The latter derivation is also obvious from our ability to rewrite $\widetilde{\boldsymbol{\mu}}_\ell\otimes\mathbf{1}_{d_{\setminus\ell}}$ as $\left(\mathbf{I}_{d_\ell}\otimes\mathbf{1}_{d_{\setminus\ell}}\right)\widetilde{\boldsymbol{\mu}}_\ell$. We gave the derivative up to permutation, for notational simplicity (since we are only interested in when it equals zero). Without the permutation, it is easy to see that it is $\mathbf{1}_{d_{<\ell}}\otimes\mathbf{I}_{d_\ell}\otimes\mathbf{1}_{d_{>\ell}}$. Finally, note that the latter derivative would be analogous if done w.r.t. the unconstrained parameters $\boldsymbol{\mu}_\ell$.

B.1.2 Derivation of the coordinate-wise minimum of m

To derive the coordinate-wise optimum of m, we will find when the gradient is zero.

$$\begin{aligned}
\frac{\partial}{\partial m}\mathrm{NLL}\left(\mathcal{D}\right) &= -\left(\boldsymbol{x}-\boldsymbol{\omega}\right)^T\boldsymbol{\Omega}\mathbf{1}_{d_\forall}\\
&= -\left(\boldsymbol{x}-\boldsymbol{\omega}\right)^T\left(\bigoplus_\ell\boldsymbol{\Psi}_\ell\right)\left(\bigotimes_\ell\mathbf{1}_{d_\ell}\right)\\
&= -\left(\boldsymbol{x}-\boldsymbol{\omega}\right)^T\left(\widetilde{\bigoplus}_\ell\boldsymbol{\Psi}_\ell\mathbf{1}_\ell\right)\\
&= \left(m\mathbf{1}_{d_\forall} - \boldsymbol{x} + \widetilde{\bigoplus}_\ell\widetilde{\boldsymbol{\mu}}_\ell\right)^T\left(\widetilde{\bigoplus}_\ell\boldsymbol{\Psi}_\ell\mathbf{1}_\ell\right)
\end{aligned}$$

Before proceeding, let's prove a helpful lemma:

Lemma 7. *For any set of $d_\ell \times d_\ell$ matrices $\mathbf{M}_\ell$:*

$$\mathbf{1}_{d_\forall}^T \left(\bigoplus_\ell \mathbf{M}_\ell \right) \mathbf{1}_{d_\forall} = \sum_\ell d_{\setminus \ell} \mathbf{1}_{d_\ell}^T \mathbf{M}_\ell \mathbf{1}_{d_\ell}$$

Proof.

$$\begin{aligned}
\mathbf{1}_{d_\forall}^T \left(\bigoplus_\ell \mathbf{M}_\ell \right) \mathbf{1}_{d_\forall} &= \mathbf{1}_{d_\forall}^T \left(\sum_\ell \mathbf{I}_{d_{<\ell}} \otimes \mathbf{M}_\ell \otimes \mathbf{I}_{d_{>\ell}} \right) \mathbf{1}_{d_\forall} \\
&= \sum_\ell \mathbf{1}_{d_{<\ell}}^T \mathbf{1}_{d_{<\ell}} \otimes \mathbf{1}_{d_\ell}^T \mathbf{M}_\ell \mathbf{1}_{d_\ell} \otimes \mathbf{1}_{d_{>\ell}}^T \mathbf{1}_{d_{>\ell}} \\
&= \sum_\ell d_{<\ell} d_{>\ell} \mathbf{1}_{d_\ell}^T \mathbf{M}_\ell \mathbf{1}_{d_\ell} \\
&= \sum_\ell d_{\setminus \ell} \mathbf{1}_{d_\ell}^T \mathbf{M}_\ell \mathbf{1}_{d_\ell}
\end{aligned}$$

□

Recall that $\mathbf{1}_{d_\ell}^T \mathbf{\Psi}_\ell \mathbf{1}_{d_\ell} \overset{\text{def}}{=} \theta_\ell$. At zero, the equality can be reshaped into the following:

$$\begin{aligned}
m \left(\bigotimes_\ell \mathbf{1}_{d_\ell}^T \right) \widetilde{\bigoplus}_\ell \mathbf{\Psi}_\ell \mathbf{1}_\ell &= \left(\mathbf{x} - \widetilde{\bigoplus}_\ell \widetilde{\boldsymbol{\mu}}_\ell \right)^T \widetilde{\bigoplus}_\ell \mathbf{\Psi}_\ell \mathbf{1}_\ell \\
\implies m \sum_\ell d_{\setminus \ell} \theta_\ell &= \left(\mathbf{x} - \widetilde{\bigoplus}_\ell \widetilde{\boldsymbol{\mu}}_\ell \right)^T \widetilde{\bigoplus}_\ell \mathbf{\Psi}_\ell \mathbf{1}_\ell \qquad \text{(Lemma 7)} \\
\implies m &= \frac{\left(\mathbf{x} - \widetilde{\bigoplus}_\ell \widetilde{\boldsymbol{\mu}}_\ell \right)^T \widetilde{\bigoplus}_\ell \mathbf{\Psi}_\ell \mathbf{1}_\ell}{\sum_\ell d_{\setminus \ell} \theta_\ell}
\end{aligned}$$

B.1.3 Derivation of the coordinate-wise minima of $\widetilde{\boldsymbol{\mu}}_\ell$

Rather than considering the gradients, we will frame this problem as an equality-constrained QP problem, which has closed-form minima. Here, we will let $\mathbf{x}_\ell$ be our vectorized dataset, vectorized in an order such that axis ℓ was considered the first axis. We'll further write $\widetilde{\mathbf{x}}_\ell = \mathbf{x}_\ell - m\mathbf{1}_{d_\forall}$ to keep our equations compact, and represent $\widetilde{\bigoplus}_{\ell'} \widetilde{\boldsymbol{\mu}}_{\ell'}$ as $\widetilde{\boldsymbol{\mu}}_\ell \widetilde{\oplus} \widetilde{\boldsymbol{\mu}}_{\setminus \ell}$.

$$\begin{aligned}
&\underset{\widetilde{\boldsymbol{\mu}}_\ell}{\operatorname{argmin}} \ \frac{1}{2} \left(\widetilde{\mathbf{x}}_\ell - \widetilde{\bigoplus}_{\ell'} \widetilde{\boldsymbol{\mu}}_{\ell'} \right)^T \mathbf{\Omega} \left(\widetilde{\mathbf{x}}_\ell - \widetilde{\bigoplus}_{\ell'} \widetilde{\boldsymbol{\mu}}_{\ell'} \right) \\
&\text{where } \ \widetilde{\boldsymbol{\mu}}_\ell^T \mathbf{1}_{d_\ell} = \mathbf{0}
\end{aligned}$$

We then expand the equation, removing terms that do not involve $\widetilde{\boldsymbol{\mu}}_\ell$ and recalling that $\mathbf{u}^T \mathbf{M} \mathbf{v} = \mathbf{v}^T \mathbf{M} \mathbf{u}$.

$$\begin{aligned}
&\underset{\widetilde{\boldsymbol{\mu}}_\ell}{\operatorname{argmin}} \frac{1}{2} \left(\widetilde{\bigoplus}_{\ell'} \widetilde{\boldsymbol{\mu}}_{\ell'}^T \right) \mathbf{\Omega} \left(\widetilde{\bigoplus}_{\ell'} \widetilde{\boldsymbol{\mu}}_{\ell'} \right) - \widetilde{\mathbf{x}}_\ell^T \mathbf{\Omega} \left(\widetilde{\bigoplus}_{\ell'} \widetilde{\boldsymbol{\mu}}_{\ell'} \right) \\
&\text{where } \ \widetilde{\boldsymbol{\mu}}_\ell^T \mathbf{1}_{d_\ell} = \mathbf{0}
\end{aligned}$$

Again, we split $\widetilde{\bigoplus}_{\ell'} \widetilde{\boldsymbol{\mu}}_{\ell'}$ into $\widetilde{\boldsymbol{\mu}}_\ell \widetilde{\oplus} \widetilde{\boldsymbol{\mu}}_{\setminus \ell}$ and remove terms not containing $\widetilde{\boldsymbol{\mu}}_\ell$.

$$\underset{\widetilde{\boldsymbol{\mu}}_\ell}{\operatorname{argmin}} \begin{cases} \frac{1}{2}\left(\widetilde{\boldsymbol{\mu}}_\ell \otimes \mathbf{1}_{d_{\setminus\ell}}\right)^T \boldsymbol{\Omega}\left(\widetilde{\boldsymbol{\mu}}_\ell \otimes \mathbf{1}_{d_{\setminus\ell}}\right) \\ +\left(\mathbf{1}_{d_\ell} \otimes \widetilde{\boldsymbol{\mu}}_{\setminus\ell} - \widetilde{\mathbf{x}}_\ell\right)^T \boldsymbol{\Omega}\left(\widetilde{\boldsymbol{\mu}}_\ell \otimes \mathbf{1}_{d_{\setminus\ell}}\right) \end{cases}$$
$$\text{where } \widetilde{\boldsymbol{\mu}}_\ell^T \mathbf{1}_{d_\ell} = \mathbf{0}$$

Finally, recall that $\widetilde{\boldsymbol{\mu}}_\ell \otimes \mathbf{1}_{d_{\setminus\ell}} = \left(\mathbf{I}_{d_\ell} \otimes \mathbf{1}_{\setminus\ell}\right)\widetilde{\boldsymbol{\mu}}_\ell$. Let's define the following constants:

$$\mathbf{A}_\ell = \left(\mathbf{I}_{d_\ell} \otimes \mathbf{1}_{d_{\setminus\ell}}\right)^T \boldsymbol{\Omega}\left(\mathbf{I}_{d_\ell} \otimes \mathbf{1}_{d_{\setminus\ell}}\right) \tag{B.1}$$

$$\mathbf{b}_\ell = \left(\mathbf{1}_{d_\ell} \otimes \widetilde{\boldsymbol{\mu}}_{\setminus\ell} - \widetilde{\mathbf{x}}_\ell\right)^T \boldsymbol{\Omega}\left(\mathbf{I}_{d_\ell} \otimes \mathbf{1}_{d_{\setminus\ell}}\right) \tag{B.2}$$

Note that this definition is different from the one in the main paper; in the next section we will show that they are equivalent. With these constants, we can rewrite our minimization problem as:

$$\underset{\widetilde{\boldsymbol{\mu}}_\ell}{\operatorname{argmin}} \ \frac{1}{2}\widetilde{\boldsymbol{\mu}}_\ell^T \mathbf{A}_\ell \widetilde{\boldsymbol{\mu}}_\ell + \mathbf{b}_\ell^T \widetilde{\boldsymbol{\mu}}_\ell$$
$$\text{where } \widetilde{\boldsymbol{\mu}}_\ell^T \mathbf{1}_{d_\ell} = \mathbf{0}$$

This is an equality-constrained QP problem; these have closed-form solutions. Letting λ be the Lagrange multiplier that appears when considering the constraint, we know that the solution is the solution to a linear equation:

$$\begin{bmatrix} \mathbf{A}_\ell & \mathbf{1}_{d_\ell} \\ \mathbf{1}_{d_\ell}^T & \mathbf{0} \end{bmatrix} \begin{bmatrix} \widetilde{\boldsymbol{\mu}}_\ell \\ \lambda \end{bmatrix} = \begin{bmatrix} -\mathbf{b}_\ell \\ \mathbf{0} \end{bmatrix}$$

We then arrive at the solution with some simple algebra.

$$\begin{aligned} & \mathbf{A}_\ell \widetilde{\boldsymbol{\mu}}_\ell + \mathbf{1}_{d_\ell}\lambda = -\mathbf{b}_\ell \\ \implies \quad & \mathbf{1}_{d_\ell}^T \widetilde{\boldsymbol{\mu}}_\ell + \mathbf{1}_{d_\ell}^T \mathbf{A}_\ell^{-1} \mathbf{1}_{d_\ell}\lambda = -\mathbf{1}_{d_\ell}^T \mathbf{A}_\ell^{-1}\mathbf{b}_\ell \\ \implies \quad & \lambda = \frac{-\mathbf{1}_{d_\ell}^T \mathbf{A}_\ell^{-1}\mathbf{b}_\ell}{\mathbf{1}_{d_\ell}^T \mathbf{A}_\ell^{-1}\mathbf{1}_{d_\ell}} \\ \implies \quad & \mathbf{A}_\ell \widetilde{\boldsymbol{\mu}}_\ell = \frac{\mathbf{1}_{d_\ell}^T \mathbf{A}_\ell^{-1}\mathbf{b}_\ell}{\mathbf{1}_{d_\ell}^T \mathbf{A}_\ell^{-1}\mathbf{1}_{d_\ell}}\mathbf{1}_{d_\ell} - \mathbf{b}_\ell \end{aligned}$$

Which leads us to the following closed-form solution:

$$\widetilde{\boldsymbol{\mu}}_\ell = \frac{\mathbf{1}_{d_\ell}^T \mathbf{A}_\ell^{-1}\mathbf{b}_\ell}{\mathbf{1}_{d_\ell}^T \mathbf{A}_\ell^{-1}\mathbf{1}_{d_\ell}} \mathbf{A}_\ell^{-1}\mathbf{1}_{d_\ell} - \mathbf{A}_\ell^{-1}\mathbf{b}_\ell$$

B.1.4 Derivation of $\mathbf{A}_\ell$ and $\mathbf{b}_\ell$

While the derivation of $\mathbf{A}_\ell$ and $\mathbf{b}_\ell$ given in the previous section yields a formula for the coordinate-wise minima of $\widetilde{\boldsymbol{\mu}}_\ell$, calculating $\mathbf{A}_\ell$ and $\mathbf{b}_\ell$ using Equations B.1 and B.2 requires multiplying together very large matrices ($\boldsymbol{\Omega}$ is $d_\forall \times d_\forall$). In this section, we will produce a simpler formula involving only $d_\ell \times d_\ell$ matrix manipulations.

We will make use of the fact that $\boldsymbol{\Omega}$ has a lower dimensional parameterization in terms of $\boldsymbol{\Psi}_\ell$. We could instead have chosen to parameterize it in terms of the identifiable parameterization $\left(\widetilde{\boldsymbol{\Psi}}_\ell, \tau\right)$, but there is not much gain in doing so; it would merely make the formulas presented here more complicated.

$$\begin{aligned}
\mathbf{A}_\ell &= \left(\mathbf{I}_{d_\ell} \otimes \mathbf{1}_{d_{\setminus\ell}}\right)^T \mathbf{\Omega} \left(\mathbf{I}_{d_\ell} \otimes \mathbf{1}_{d_{\setminus\ell}}\right) \\
&= \left(\mathbf{I}_{d_\ell} \otimes \mathbf{1}_{d_{\setminus\ell}}\right)^T \left(\mathbf{\Psi}_\ell \otimes \mathbf{I}_{d_{\setminus\ell}} + \mathbf{I}_{d_\ell} \otimes \mathbf{\Psi}_{\setminus\ell}\right) \left(\mathbf{I}_{d_\ell} \otimes \mathbf{1}_{d_{\setminus\ell}}\right) \\
&= \left(\mathbf{\Psi}_\ell \otimes \mathbf{1}_{d_{\setminus\ell}}^T \mathbf{1}_{d_{\setminus\ell}} + \mathbf{I}_{d_\ell} \otimes \mathbf{1}_{d_{\setminus\ell}}^T \mathbf{\Psi}_{\setminus\ell} \mathbf{1}_{d_{\setminus\ell}}\right) \\
&= d_{\setminus\ell} \mathbf{\Psi}_\ell + \sum_{\ell' \neq \ell} \frac{d_\forall}{d_\ell d_{\ell'}} \theta_\ell \mathbf{I}_{d_\ell} && \text{(Lemma 7)} \\
&= d_{\setminus\ell} \mathbf{\Psi}_\ell + \theta_{\setminus\ell} \mathbf{I}_{d_\ell}
\end{aligned}$$

$$\begin{aligned}
\mathbf{b}_\ell &= \left(\mathbf{1}_{d_\ell} \otimes \widetilde{\boldsymbol{\mu}}_{\setminus\ell} - \widetilde{\mathbf{x}}_\ell\right)^T \mathbf{\Omega} \left(\mathbf{I}_{d_\ell} \otimes \mathbf{1}_{d_{\setminus\ell}}\right) \\
&= \left(\mathbf{1}_{d_\ell} \otimes \widetilde{\boldsymbol{\mu}}_{\setminus\ell} - \widetilde{\mathbf{x}}_\ell\right)^T \left(\mathbf{\Psi}_\ell \otimes \mathbf{1}_{d_{\setminus\ell}} + \mathbf{I}_{d_\ell} \otimes \mathbf{\Psi}_{\setminus\ell} \mathbf{1}_{d_{\setminus\ell}}\right) \\
&= \mathbf{1}_{d_\ell}^T \mathbf{\Psi}_\ell \otimes \widetilde{\boldsymbol{\mu}}_{\setminus\ell}^T \mathbf{1}_{d_{\setminus\ell}} + \mathbf{1}_{d_\ell} \otimes \widetilde{\boldsymbol{\mu}}_{d_{\setminus\ell}}^T \mathbf{\Psi}_{\setminus\ell} \mathbf{1}_{d_{\setminus\ell}} \\
&\quad - \widetilde{\mathbf{x}}_\ell^T \left(\mathbf{\Psi}_\ell \otimes \mathbf{1}_{d_{\setminus\ell}} + \mathbf{I}_{d_\ell} \otimes \mathbf{\Psi}_{\setminus\ell} \mathbf{1}_{d_{\setminus\ell}}\right)
\end{aligned}$$

Here, we can observe that $\widetilde{\boldsymbol{\mu}}_{\setminus\ell}^T \mathbf{1}_{d_{\setminus\ell}} = 0$ due to our constraints. Thus, we get:

$$\mathbf{b}_\ell = \left[\mathbf{1}_{d_{\setminus\ell}}^T \mathbf{\Psi}_\ell \mathbf{1}_{d_{\setminus\ell}}\right] \mathbf{1}_{d_\ell} - \widetilde{\mathbf{x}}_\ell^T \left(\mathbf{\Psi}_\ell \otimes \mathbf{1}_{d_{\setminus\ell}} + \mathbf{I}_{d_\ell} \otimes \mathbf{\Psi}_{\setminus\ell} \mathbf{1}_{d_{\setminus\ell}}\right)$$

Expanding out $\widetilde{\mathbf{x}}_\ell$ into $\mathbf{x}_\ell - m\mathbf{1}_{d_\forall}$ yields the definition of b_ℓ given in the main paper.

$$\begin{aligned}
&m\mathbf{1}_{d_\forall}^T \left(\mathbf{\Psi}_\ell \otimes \mathbf{1}_{d_{\setminus\ell}} + \mathbf{I}_{d_\ell} \otimes \mathbf{\Psi}_{\setminus\ell} \mathbf{1}_{d_{\setminus\ell}}\right) \\
&= m d_{\setminus\ell} \mathbf{1}_{d_\ell}^T \mathbf{\Psi}_\ell + m\theta_{\setminus\ell} \mathbf{1}_{d_{\setminus\ell}}
\end{aligned}$$

Unfortunately, this still involves a large multiplication; can we express it in a simpler way?

We can, but it involves introducing the concept of a matricization of a tensor: for a $d_1 \times ... \times d_K$ tensor $\mathcal{T}$, the ℓ-matricization of it, $\text{mat}_\ell[\mathcal{T}]$ is a $d_\ell \times d_{\setminus\ell}$ matrix constructed by stacking all axes together except ℓ. This allows us a more formal definition of $\mathbf{x}_\ell$: $\mathbf{x}_\ell \stackrel{\text{def}}{=} \text{vec mat}_\ell[\mathcal{D}]$.

We can then utilize a well-known convenient relation between Kronecker products and vectorization[1]:

$$\text{vec}[\mathbf{M}]^T (\mathbf{U} \otimes \mathbf{V}) = \text{vec}\left[\mathbf{U}^T \mathbf{M} \mathbf{V}\right]^T$$

Note that $\mathbf{\Psi}_\ell$ is symmetric, so the transpose is irrelevant. This allows us to simplify.

$$\begin{aligned}
&\text{vec mat}[\mathcal{D}]^T \left(\mathbf{\Psi}_\ell \otimes \mathbf{1}_{d_{\setminus\ell}} + \mathbf{I}_{d_\ell} \otimes \mathbf{\Psi}_{\setminus\ell} \mathbf{1}_{d_{\setminus\ell}}\right) \\
&= \text{vec}\left[\mathbf{\Psi}_\ell \text{mat}_\ell[\mathcal{D}] \mathbf{1}_{d_{\setminus\ell}}\right]^T + \text{vec}\left[\text{mat}_\ell[\mathcal{D}] \mathbf{\Psi}_{\setminus\ell} \mathbf{1}_{d_{\setminus\ell}}\right]^T
\end{aligned}$$

As a final step for simplification, we recall another relation from Kolda and Bader (2009). First, let $\mathcal{D} \times_\ell \mathbf{M}$ represent multiplying a tensor $\mathcal{D}$ along its ℓth axis by a matrix $\mathbf{M}$. Note that for matrices, $\mathbf{N} \times_0 \mathbf{M} = \mathbf{M}^T \mathbf{N}$ and $\mathbf{N} \times_1 \mathbf{M} = \mathbf{N}\mathbf{M}$. Intuitively, it corresponds to batch matrix multiplication when the tensor has more than two axes.

[1]There are two conventions for vec, rows-first or columns-first; that convention affects the order of multiplication presented here. We use the rows-first convention as that aligns with the convention used for mat.

$$\text{mat}_\ell\left[\mathcal{T}\right]\bigotimes_{\ell'\neq\ell}\mathbf{M}_{\ell'} = \text{mat}_\ell\left[\mathcal{T}\bigtimes_{\ell'\neq\ell}\mathbf{M}_{\ell'}\right]$$

This allows us to rewrite our expression as:

$$\begin{aligned}
&\text{vec}\left[\boldsymbol{\Psi}_\ell\text{mat}_\ell\left[\mathcal{D}\right]\mathbf{1}_{d_{\setminus\ell}}\right]^T + \text{vec}\left[\text{mat}_\ell\left[\mathcal{D}\right]\boldsymbol{\Psi}_{\setminus\ell}\mathbf{1}_{d_{\setminus\ell}}\right]^T\\
&= \text{vec}\left[\boldsymbol{\Psi}_\ell\text{mat}_\ell\left[\mathcal{D}\right]\mathbf{1}_{d_{\setminus\ell}}\right]^T\\
&\quad + \sum_{\ell'\neq\ell}\text{vec}\left[\text{mat}_\ell\left[\mathcal{D}\times_{\ell'}\boldsymbol{\Psi}_{\ell'}\right]\mathbf{1}_{d_{\setminus\ell}}\right]^T
\end{aligned}$$

These always result in vectors, so we can drop the vec prefix.

$$\boldsymbol{\Psi}_\ell\text{mat}_\ell\left[\mathcal{D}\right]\mathbf{1}_{d_{\setminus\ell}} + \sum_{\ell'\neq\ell}\text{mat}_\ell\left[\mathcal{D}\times_{\ell'}\boldsymbol{\Psi}_{\ell'}\right]\mathbf{1}_{d_{\setminus\ell}}$$

This looks complicated, but has a fairly straightforward interpretation. The $\boldsymbol{\Psi}_\ell\text{mat}_\ell\left[\mathcal{D}\right]\mathbf{1}_{d_{\setminus\ell}}$ term corresponds to summing over all axes of $\mathcal{D}$ except ℓ, and then transforming the sum by $\boldsymbol{\Psi}_\ell$. Each $\text{mat}_\ell\left[\mathcal{D}\times_{\ell'}\boldsymbol{\Psi}_{\ell'}\right]\mathbf{1}_{d_{\setminus\ell}}$ term can be interpreted as a batch matrix multiplication along axis ℓ', before summing over all axes but ℓ. Of course, in practice it is more efficient to perform some sums first before the multiplication, as given below, but it causes rather cumbersome notation.

$$\sum_{\ell'\neq\ell}\text{mat}_\ell\left[\left(\mathcal{D}\bigtimes_{\ell\neq\ell''\neq\ell'}\mathbf{1}_{\setminus\ell''}\right)\times_{\ell'}\boldsymbol{\Psi}_{\ell'}\mathbf{1}_{\setminus\ell'}\right]$$

To summarize, we have that:

$$\begin{aligned}
\mathbf{b}_\ell = &\left[\mathbf{1}_{d_{\setminus\ell}}^T\boldsymbol{\Psi}_\ell\mathbf{1}_{d_{\setminus\ell}}\right]\mathbf{1}_{d_\ell} + md_{\setminus\ell}\mathbf{1}_{d_\ell}^T\boldsymbol{\Psi}_\ell + m\theta_{\setminus\ell}\mathbf{1}_{d_{\setminus\ell}}\\
&- \boldsymbol{\Psi}_\ell\text{mat}_\ell\left[\mathcal{D}\right]\mathbf{1}_{d_{\setminus\ell}} - \sum_{\ell'\neq\ell}\text{mat}_\ell\left[\mathcal{D}\times_{\ell'}\boldsymbol{\Psi}_{\ell'}\right]\mathbf{1}_{d_{\setminus\ell}}
\end{aligned}$$

B.1.5 Invertibility of $\mathbf{A}_\ell$

Recall that, from the previous section, $\mathbf{A}_\ell$ can be written as:

$$\mathbf{A}_\ell = \left(\mathbf{I}_{d_\ell}\otimes\mathbf{1}_{d_{\setminus\ell}}\right)^T\boldsymbol{\Omega}\left(\mathbf{I}_{d_\ell}\otimes\mathbf{1}_{d_{\setminus\ell}}\right)$$

$\boldsymbol{\Omega}_\ell$ is a precision matrix, i.e. positive definite. Furthermore, $\left(\mathbf{I}_{d_\ell}\otimes\mathbf{1}_{d_{\setminus\ell}}\right)$ never maps nonzero vectors to the zero vector (it is a matrix that 'copy-and-pastes' the input vector $d_{\setminus\ell}$ times).

$$\begin{aligned}
\mathbf{v}^T\mathbf{A}_\ell\mathbf{v} &= \mathbf{v}^T\left(\mathbf{I}_{d_\ell}\otimes\mathbf{1}_{d_{\setminus\ell}}\right)^T\boldsymbol{\Omega}\left(\mathbf{I}_{d_\ell}\otimes\mathbf{1}_{d_{\setminus\ell}}\right)\mathbf{v} && \mathbf{v}\neq\mathbf{0}\\
&= \mathbf{w}^T\boldsymbol{\Omega}\mathbf{w} && \mathbf{w}\neq\mathbf{0}\\
&> 0 && \text{(positive-definiteness of } \boldsymbol{\Omega})
\end{aligned}$$

In fact, this argument shows that $\mathbf{A}_\ell$ is not merely invertible, but positive definite!

What if $\mathbf{\Omega}$ is merely positive semidefinite? Note that our NLL contains a $-\log|\mathbf{\Omega}|$ term, which diverges to infinity as $\mathbf{\Omega}$ approaches semidefiniteness. Since it is known that optimizers for $\mathbf{\Omega}$ converge (see for example Theorem 6 in the paper that introduced TeraLasso Greenewald et al., 2019), this cannot happen: $\mathbf{\Omega}$ must be positive definite.

Some work has been done investigating the case where we force $\mathbf{\Omega}$ to be restricted to the set of rank-k matrices, such as in the preprint by Andrew et al. (2024). The log-determinant becomes a log-pseudodeterminant. In this case, $\mathbf{A}_\ell$ is indeed not invertible; the QP problem has multiple solutions. However, in such work the nullspace of $\mathbf{\Omega}$ is known[2]: we can add linear constraints to our QP problem to force our domain to be orthogonal to the nullspace, recovering uniqueness-of-coordinate-wise-solution and hence guaranteeing that the coordinate descent of Algorithm 1 converges.

B.1.6 Proof of global optimality

In this section, we will prove Theorem 4 in the main paper.

Theorem 1. *The maximum likelihood of the noncentral Kronecker-sum-structured normal distribution has a unique maximum, which is global. The estimator defined by Algorithm 1 converges to this.*

The fact that Algorithm 1 converges follows from the fact that coordinate descent converges to stationary points if the coordinate-wise minima are unique, which we know is true given that our estimators for $\boldsymbol{\mu}_\ell, m$, and $\mathbf{\Omega}$ are unique (estimating $\mathbf{\Omega}$ is not considered here, but it is a strictly convex optimization problem so whatever drop-in-algorithm we use for it should give a unique answer, potentially differing in the precise parameterization of $\mathbf{\Omega}$ but not $\mathbf{\Omega}$ itself).

To prove that the negative log-likelihood has a unique minimum, we intend to prove the following chain of implications:

$$\begin{aligned}
&(\boldsymbol{\omega}, \mathbf{\Omega}) \in \text{local minima}\,(\mathfrak{A}) \\
\underset{1}{\iff}\ &(\boldsymbol{\omega}, \mathbf{\Omega}, s=1) \in \text{local minima}\,\left(\mathfrak{A}'\right) \\
\underset{2}{\iff}\ &(\mathbf{\Gamma}) \in \text{local minima}\,(\mathfrak{B}) \\
\underset{3}{\iff}\ &(\mathbf{\Gamma}) \in \text{global minima}\,(\mathfrak{B})
\end{aligned}$$

We will first focus on the case where $\mathbf{\Omega}$ is estimated with regularizers, rather than the low-rank subspace restriction. Letting ρ be some convex regularizer for the precision matrix (typically the graphical lasso, $\rho(\mathbf{\Omega}) = \sum_\ell \rho_\ell \|\mathbf{\Psi}_\ell\|_{1,od}$), we define our problems as:

$$\mathfrak{A} = \begin{cases} \underset{\boldsymbol{\omega}, \mathbf{\Omega}}{\text{argmin}} & -\log|\mathbf{\Omega}| + (\mathbf{x}-\boldsymbol{\omega})^T\mathbf{\Omega}(\mathbf{x}-\boldsymbol{\omega}) + \rho\,(\mathbf{\Omega}) \\ \text{where} & \mathbf{\Omega} \in \mathbb{K}_{\mathbf{M}} \succeq \mathbf{0} \\ & \boldsymbol{\omega} \in \mathbb{K}_{\boldsymbol{v}} \end{cases}$$

$$\mathfrak{A}' = \begin{cases} \underset{\boldsymbol{\omega}, \mathbf{\Omega}, s}{\text{argmin}} & -\log|\mathbf{\Gamma}| + \begin{bmatrix}\boldsymbol{x}^T & 1\end{bmatrix}\mathbf{\Gamma}\begin{bmatrix}\boldsymbol{x} \\ 1\end{bmatrix} + \rho\,(\mathbf{\Omega}) \\ \text{where} & \mathbf{\Omega} \in \mathbb{K}_{\mathbf{M}} \succeq \mathbf{0} \\ & \boldsymbol{\omega} \in \mathbb{K}_{\boldsymbol{v}} \\ & s > 0 \\ & \mathbf{\Gamma} = \begin{bmatrix} \mathbf{\Omega} & -\mathbf{\Omega}\boldsymbol{\omega} \\ -\boldsymbol{\omega}^T\mathbf{\Omega} & \frac{1}{s} + \boldsymbol{\omega}^T\mathbf{\Omega}\boldsymbol{\omega} \end{bmatrix} \end{cases}$$

$$\mathfrak{B} = \begin{cases} \underset{\mathbf{\Gamma}}{\text{argmin}} & -\log|\mathbf{\Gamma}| + \begin{bmatrix}\boldsymbol{x}^T & 1\end{bmatrix}\mathbf{\Gamma}\begin{bmatrix}\boldsymbol{x} \\ 1\end{bmatrix} + \rho\left(\mathbf{\Gamma}_{1:d_\forall, 1:d_\forall}\right) \\ \text{where} & \mathbf{\Gamma}_{1:d_\forall, 1:d_\forall} \in \mathbb{K}_{\mathbf{M}} \\ & \mathbf{\Gamma}_{1:d_\forall, d_\forall+1} \in \mathbb{K}_{\mathbf{M}\boldsymbol{v}} \\ & \mathbf{\Gamma} \succeq \mathbf{0} \end{cases}$$

[2]The work finds an exact solution for the eigenvectors of $\mathbf{\Psi}_\ell$.

Recall that $\mathbb{K}_{\mathbf{M}\boldsymbol{v}}$ is a linear subspace, so $\mathfrak{B}$ corresponds to a conic program with linear constraints. The objective is strictly convex. Thus, $\mathcal{B}$ has a unique local minimum, which is also the global minimum. This satisfies implication (3).

Next, note that there is a one-to-one correspondence between $(\boldsymbol{\omega}, \boldsymbol{\Omega}, s)$ and $\boldsymbol{\Gamma}$. $\boldsymbol{\Omega} = \boldsymbol{\Gamma}_{1:d_\forall, 1:d_\forall}$, and since $\boldsymbol{\Omega}$ is guaranteed-to-be-invertible we can get $\boldsymbol{\omega}$ from $\boldsymbol{\Gamma}_{1:d_\forall, d_\forall+1}$. Finally, $\frac{1}{s}$ can be derived with knowledge of the previous two parameters from $\boldsymbol{\Gamma}_{d_\forall+1, d_\forall+1}$. Since the correspondence is one-to-one, the minima of $\mathfrak{A}'$ and $\mathfrak{B}$ are identical[3] - since $\mathfrak{B}$ has a unique minimum, so does $\mathfrak{A}'$. Thus, to prove implication (2) we need only show that the minimum of $\mathfrak{A}'$ occurs when $s = 1$.

$$
\begin{aligned}
\log|\boldsymbol{\Gamma}| &= \log\left(|\boldsymbol{\Omega}|\left|\frac{1}{s} + \boldsymbol{\omega}^T\boldsymbol{\Omega}\boldsymbol{\omega} - \boldsymbol{\omega}^T\boldsymbol{\Omega}\boldsymbol{\Omega}^{-1}\boldsymbol{\Omega}\boldsymbol{\omega}\right|\right) && \text{(Block matrix determinant)} \\
&= \log|\boldsymbol{\Omega}| - \log s \\
\begin{bmatrix}\boldsymbol{x}^T & 1\end{bmatrix}\boldsymbol{\Gamma}\begin{bmatrix}\boldsymbol{x} \\ 1\end{bmatrix} &= \begin{bmatrix}\mathbf{x}^T\boldsymbol{\Omega} - \boldsymbol{\omega}^T\boldsymbol{\Omega} & \frac{1}{s} + \boldsymbol{\omega}^T\boldsymbol{\Omega}\boldsymbol{\omega} - \mathbf{x}^T\boldsymbol{\Omega}\boldsymbol{\omega}\end{bmatrix}\begin{bmatrix}\mathbf{x} \\ 1\end{bmatrix} \\
&= \mathbf{x}^T\boldsymbol{\Omega}\mathbf{x} - \boldsymbol{\omega}^T\boldsymbol{\Omega}\mathbf{x} + \frac{1}{s} + \boldsymbol{\omega}^T\boldsymbol{\Omega}\boldsymbol{\omega} - \mathbf{x}^T\boldsymbol{\Omega}\boldsymbol{\omega} \\
&= (\mathbf{x} - \boldsymbol{\omega})^T\boldsymbol{\Omega}(\mathbf{x} - \boldsymbol{\omega}) + \frac{1}{s}
\end{aligned}
$$

Firstly, note that all terms involving s are separable from the rest. Differentiating with respect to s, we see that $0 = \frac{1}{s} - \frac{1}{s^2}$ at the minimum, i.e. that $s = s^2$; since $s > 0$, this leaves $s = 1$ must hold at the minimum; thus, implication (2) holds. Secondly, note that these equations lead to the same NLL as $\mathfrak{A}$, except with an additive $\frac{1}{s} + \log s$ term thrown in. Since we know $s = 1$ at the minimum, $\mathfrak{A}'$ reduces to $\mathfrak{A}$; implication (1) holds as well.

This completes the proof; all three implications hold. The log-likelihood of $\mathfrak{A}'$ and $\mathfrak{B}$ at $s = 1$ are the same, and hence all local minima are global minima. This holds when using regularization, but what about restrictions to a low-rank subspace?

Global optimality under low-rank restrictions

As mentioned in previous sections, most methods of estimating $\boldsymbol{\Omega}$ use regularization to guarantee existence of a solution. Some restrict $\boldsymbol{\Omega}$ to a low-rank subspace, which requires changing determinants to pseudodeterminants. All the implications still hold in this case, but the block-determinant formula no longer holds and the $\mathfrak{A}' \to \mathfrak{B}$ mapping is no longer one-to-one, since it is through invertibility of $\boldsymbol{\Omega}$ that we recover $\boldsymbol{\omega}$. Under the low rank assumption, we can add the following constraint to $\mathfrak{A}$ and $\mathfrak{A}'$ to recover uniqueness.

$$\boldsymbol{\omega} \perp \text{nullspace}[\boldsymbol{\Omega}]$$

The constraints work by enforcing $\boldsymbol{\omega}$ to be orthogonal to the nullspace of $\boldsymbol{\Omega}$. Note that this has no effect on $\mathfrak{B}$, since any component of $\boldsymbol{\omega}$ that is not orthogonal to the nullspace gets mapped to zero. This preserves the one-to-one mapping needed for implication (2), and the convexity needed for implication (3).

What needs to be re-proven for implication (1) to hold is the following:

1. Local minima of $\mathfrak{A}'$ occur only at $s = 1$
2. At $s = 1$, $\mathfrak{A}$ and $\mathfrak{A}'$ have the same solutions.

Unfortunately, pseudodeterminants are *much* harder to work with than typical determinants. We rely on two results from the literature, from Holbrook (2018) and Hartwig (1976):

[3]To ensure no saddle points were introduced, it is easy to check that the gradient of one parameterization equals zero if the gradient of the other does too at the corresponding point.

Theorem 2.20 (Holbrook 2018)**.**

$$d\det{}^{\dagger}\mathbf{A} = \det{}^{\dagger}[\mathbf{A}]\,\mathrm{tr}\left[\mathbf{A}^{\dagger}d\mathbf{A}\right]$$

Where $\det^{\dagger}$ *is the pseudodeterminant and* $\mathbf{A}^{\dagger}$ *is the pseudoinverse.*

Corollary 5.

$$\frac{d\,\log\det{}^{\dagger}\mathbf{\Gamma}}{ds} = \mathrm{tr}\left[\mathbf{\Gamma}^{\dagger}\frac{d\mathbf{\Gamma}}{ds}\right]$$

The second theorem, by Hartwig, gives an expression for the pseudoinverse of a 'bordered matrix', i.e. a matrix of the form $\begin{bmatrix}\mathbf{A} & \mathbf{c}\\ \mathbf{b} & d\end{bmatrix}$. However, they split the problem into several sub-cases, which are not immediately apparent from the form of $\mathbf{A}, \mathbf{b}, \mathbf{c}$, and d. Thus, we will first manipulate our matrix into one of these sub-cases before stating the theorem.

Recall that $\mathbf{\Omega}$ is positive semi-definite, and thus it has an eigendecomposition $\begin{bmatrix}\mathbf{V}_1 & \mathbf{V}_2\end{bmatrix}\begin{bmatrix}\mathbf{\Lambda} & \mathbf{0}\\ \mathbf{0} & \mathbf{0}\end{bmatrix}\begin{bmatrix}\mathbf{V}_1^T\\ \mathbf{V}_2^T\end{bmatrix}$. Letting $\mathbf{V} = \begin{bmatrix}\mathbf{V}_1 & \mathbf{V}_2\end{bmatrix}$, observe the following:

$$\begin{aligned}
&\begin{bmatrix}\mathbf{V}^T & \mathbf{0}\\ \mathbf{0} & 1\end{bmatrix}\begin{bmatrix}\mathbf{\Omega} & -\mathbf{\Omega}\boldsymbol{\omega}\\ -\boldsymbol{\omega}^T\mathbf{\Omega} & \frac{1}{s}+\boldsymbol{\omega}^T\mathbf{\Omega}\boldsymbol{\omega}\end{bmatrix}\begin{bmatrix}\mathbf{V} & \mathbf{0}\\ \mathbf{0} & 1\end{bmatrix}\\
&\quad= \begin{bmatrix}\mathbf{V}^T\mathbf{\Omega}\mathbf{V} & -\mathbf{V}^T\mathbf{\Omega}\boldsymbol{\omega}\\ \boldsymbol{\omega}^T\mathbf{\Omega}\mathbf{V} & \frac{1}{s}+\boldsymbol{\omega}^T\mathbf{\Omega}\boldsymbol{\omega}\end{bmatrix}\\
&\quad= \begin{bmatrix}\mathbf{\Lambda} & \mathbf{0} & -\mathbf{\Lambda}\mathbf{V}_1^T\boldsymbol{\omega}\\ \mathbf{0} & \mathbf{0} & \mathbf{0}\\ -\boldsymbol{\omega}^T\mathbf{V}_1\mathbf{\Lambda} & \mathbf{0} & \frac{1}{s}+\boldsymbol{\omega}^T\mathbf{\Omega}\boldsymbol{\omega}\end{bmatrix}
\end{aligned}$$

We will now find the pseudoinverse of this new matrix, as $\mathbf{U}\mathbf{M}^{\dagger}\mathbf{U}^T = \left(\mathbf{U}\mathbf{M}\mathbf{U}^T\right)^{\dagger}$ for unitary matrices $\mathbf{U}$. In fact, the bottom-right entry is entirely unchanged by this transformation, and it will turn out to be all we need for our proof.

Theorem (Hartwig (1976))**.** *Suppose we have a matrix of the form* $\begin{bmatrix}\mathbf{\Sigma} & \mathbf{0} & \mathbf{q}\\ \mathbf{0} & \mathbf{0} & \mathbf{0}\\ \mathbf{p}^T & \mathbf{0} & d\end{bmatrix}$ *for diagonal* $\mathbf{\Sigma}$ *of full rank, and* $z = d - \mathbf{p}^T\mathbf{\Sigma}^{-1}\mathbf{q} \neq 0$*. Then, the lower right hand entry of the pseudoinverse is* $\frac{1}{z}$*. Furthermore, the matrix has rank* $1 + \mathrm{rank}\mathbf{\Sigma}$*.*

This is, of course, a special case of their full result giving a formula for the entire pseudoinverse - but it is all we need.

Corollary 6. *The lower right hand entry of* $\mathbf{\Gamma}^{\dagger}$ *is* s*.*

Proof. We already saw that our matrix can be turned into the form required by Hartwig's theorem without changing what the lower right hand entry will be. It then suffices to observe that $z = \frac{1}{s} + \boldsymbol{\omega}^T\mathbf{\Omega}\boldsymbol{\omega} - \boldsymbol{\omega}^T\mathbf{V}_1\mathbf{\Lambda}\mathbf{\Lambda}^{-1}\mathbf{\Lambda}\mathbf{V}_1^T\boldsymbol{\omega} = \frac{1}{s}$. □

From Corollary 5, the derivative of the log pseudodeterminant is $\mathrm{tr}\left[\mathbf{\Gamma}^{\dagger}\begin{bmatrix}\mathbf{0} & \mathbf{0}\\ \mathbf{0} & -\frac{1}{s^2}\end{bmatrix}\right] = \mathrm{tr}\begin{bmatrix}\mathbf{0} & \frac{-1}{s^2}\mathbf{\Gamma}^{\dagger}_{1:d_{\forall},d_{\forall}+1}\\ \mathbf{0} & \frac{-1}{s^2}\mathbf{\Gamma}^{\dagger}_{d_{\forall}+1,d_{\forall}+1}\end{bmatrix}$. Thus, it only depends on the bottom right entry of the pseudoinverse; in particular, it is $\frac{1}{s}$. From this, the arguments in the previous section hold, and we can show that any minima occur only at $s = 1$.

All that remains is to show that at $s = 1$, $\mathfrak{A}$ and $\mathfrak{A}'$ have the same solutions. We already know from earlier that the pseudodeterminant of $\mathbf{\Gamma}$ is the same as the pseudodeterminant of $\begin{bmatrix}\mathbf{\Lambda} & \mathbf{0} & -\mathbf{\Lambda}\mathbf{V}_1^T\boldsymbol{\omega}\\ \mathbf{0} & \mathbf{0} & \mathbf{0}\\ -\boldsymbol{\omega}^T\mathbf{V}_1\mathbf{\Lambda} & \mathbf{0} & 1+\boldsymbol{\omega}^T\mathbf{\Omega}\boldsymbol{\omega}\end{bmatrix}$.

Removing out the zero rows and columns, we are left with $\begin{bmatrix} \mathbf{\Lambda} & -\mathbf{\Lambda}\mathbf{V}_1^T\boldsymbol{\omega} \\ -\boldsymbol{\omega}^T\mathbf{V}_1\mathbf{\Lambda} & 1+\boldsymbol{\omega}^T\mathbf{\Omega}\boldsymbol{\omega} \end{bmatrix}$, which by Hartwig is a full rank matrix (recall the theorem stated its rank as $1+\text{rank}\mathbf{\Lambda}$ and that $\mathbf{\Lambda}$ corresponds to the non-zero eigenvalues). Hence, we can use the ordinary block determinant rule.

$$\begin{aligned}\det{}^\dagger\mathbf{\Gamma} &= \det\begin{bmatrix} \mathbf{\Lambda} & -\mathbf{\Lambda}\mathbf{V}_1^T\boldsymbol{\omega} \\ -\boldsymbol{\omega}^T\mathbf{V}_1\mathbf{\Lambda} & 1+\boldsymbol{\omega}^T\mathbf{\Omega}\boldsymbol{\omega} \end{bmatrix} \\ &= \det\left[\mathbf{\Lambda}\right]\det\left[1+\boldsymbol{\omega}^T\mathbf{\Omega}\boldsymbol{\omega}-\boldsymbol{\omega}^T\mathbf{V}_1\mathbf{\Lambda}\mathbf{V}_1^T\boldsymbol{\omega}\right]\end{aligned}$$

Recall that we required $\boldsymbol{\omega}$ to be orthogonal to the nullspace of $\mathbf{\Omega}$, resulting in:

$$\begin{aligned}\det{}^\dagger\mathbf{\Gamma} &= \det\left[\mathbf{\Lambda}\right]\det\left[1+\boldsymbol{\omega}^T\mathbf{V}_1\mathbf{\Lambda}\mathbf{V}_1^T\boldsymbol{\omega}-\boldsymbol{\omega}^T\mathbf{V}_1\mathbf{\Lambda}\mathbf{V}_1^T\boldsymbol{\omega}\right] \\ &= \det\mathbf{\Lambda} \\ &= \det{}^\dagger\mathbf{\Omega}\end{aligned}$$

Thus, we have that:

$$\mathfrak{A}' = \begin{cases} \underset{\boldsymbol{\omega},\mathbf{\Omega},s}{\text{argmin}} & -\log\det{}^\dagger\mathbf{\Gamma}+\begin{bmatrix}\boldsymbol{x}^T & 1\end{bmatrix}\mathbf{\Gamma}\begin{bmatrix}\boldsymbol{x}\\1\end{bmatrix}+\rho\left(\mathbf{\Omega}\right) \\ \text{where} & \mathbf{\Omega}\in\mathbb{K}_{\mathbf{M}}\succeq\mathbf{0} \\ & \boldsymbol{\omega}\in\mathbb{K}_{\boldsymbol{v}} \\ & \boldsymbol{\omega}\perp\text{nullspace }\mathbf{\Omega} \\ & s>0 \end{cases}$$

is equivalent to:

$$\mathfrak{A} = \begin{cases} \underset{\boldsymbol{\omega},\mathbf{\Omega}}{\text{argmin}} & -\log\det{}^\dagger\mathbf{\Omega}+\left(\mathbf{x}-\boldsymbol{\omega}\right)^T\mathbf{\Omega}\left(\mathbf{x}-\boldsymbol{\omega}\right)+\rho\left(\mathbf{\Omega}\right) \\ \text{where} & \mathbf{\Omega}\in\mathbb{K}_{\mathbf{M}}\succeq\mathbf{0} \\ & \boldsymbol{\omega}\in\mathbb{K}_{\boldsymbol{v}} \\ & \boldsymbol{\omega}\perp\text{nullspace }\mathbf{\Omega} \end{cases}$$

This completes the proof.

B.2 Erdos-Renyi synthetic data

We mentioned in the results section that we repeated our experiments with Erdos-Renyi graphs (p=0.05) instead of Barabasi-Albert graphs. We found identical results, except that zero-mean algorithms perform much worse under Poisson noise. See Figure B.1 for the results, which use the same mean distributions as from the main paper.

B.3 Number of iterations needed

In the paper, we mentioned that, in practice, typically only two iterations were needed of our flip-flop algorithm to recover good performance. One iteration of our algorithm is equivalent to projecting the mean onto $\mathbb{K}_v$, removing it, and then estimating the graph. To understand how many iterations are needed, we perform the same experiments as in the main paper at different levels of iterations. We only present results with GmGM, as TeraLasso performs analogously here in all other experiments. The results are given in Figure B.2; we can see that in most cases, with just two iterations our algorithm recovers the performance of our algorithm iterated to convergence. There is only one exception, when our algorithm exactly follows the KS-structured Gaussian distribution. In Figure B.3, we investigate this case further, seeing that 3 iterations are sufficient.

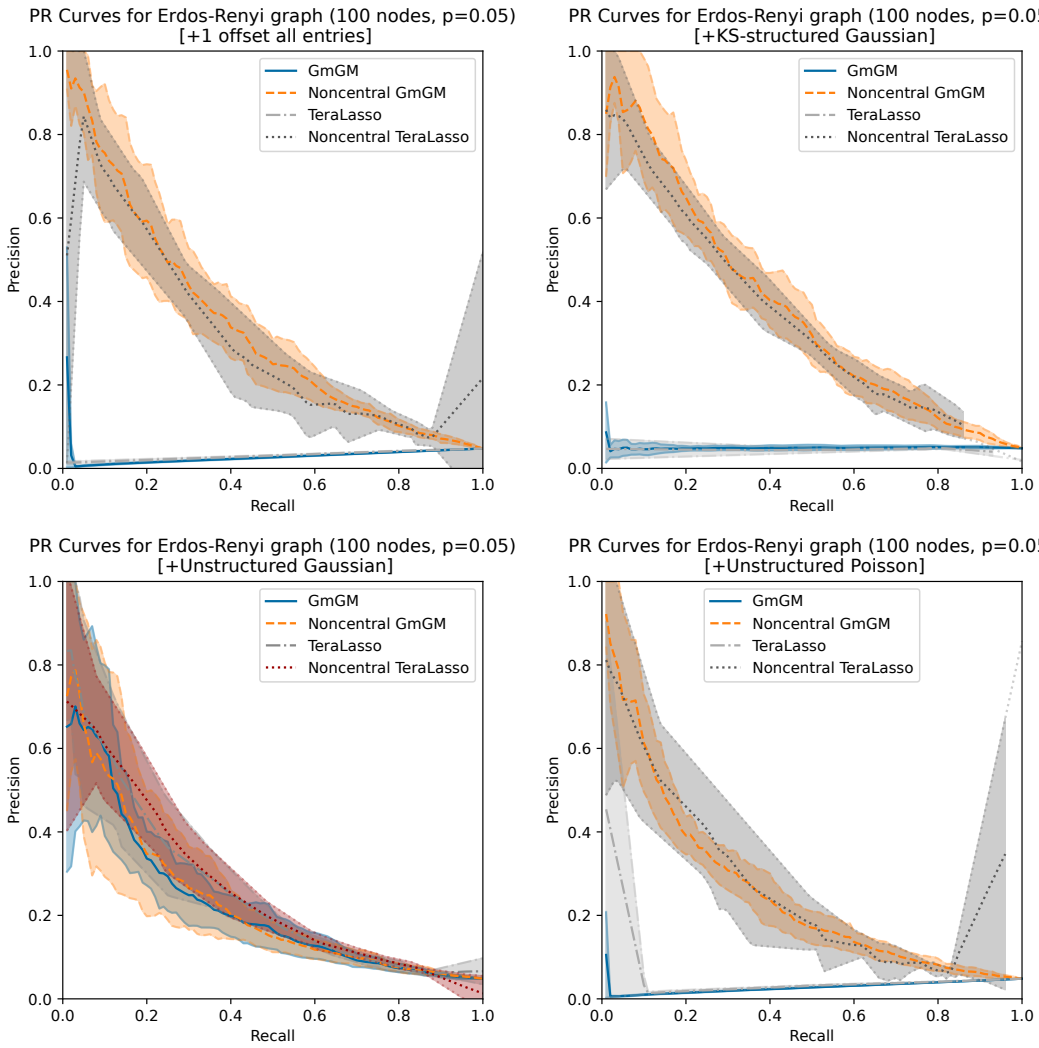


Figure B.1: Precision and recall for a synthetic dataset. Error bars are the best/worst performance over 10 trials; the center line is average performance.

Bibliography for Paper 2 (appendix)

Andrew, B., Westhead, D. R., & Cutillo, L. (2024, July). Making Multi-Axis Gaussian Graphical Models Scalable to Millions of Samples and Features [arXiv:2407.19892 [cs, q-bio, stat]]. https://doi.org/10.48550/arXiv.2407.19892

Greenewald, K., Zhou, S., & Hero, A. (2019). Tensor graphical lasso (TeraLasso) [Publisher: [Royal Statistical Society, Oxford University Press]]. *Journal of the Royal Statistical Society. Series B (Statistical Methodology)*, *81*(5), 901–931. Retrieved July 16, 2024, from https://www.jstor.org/stable/26833080

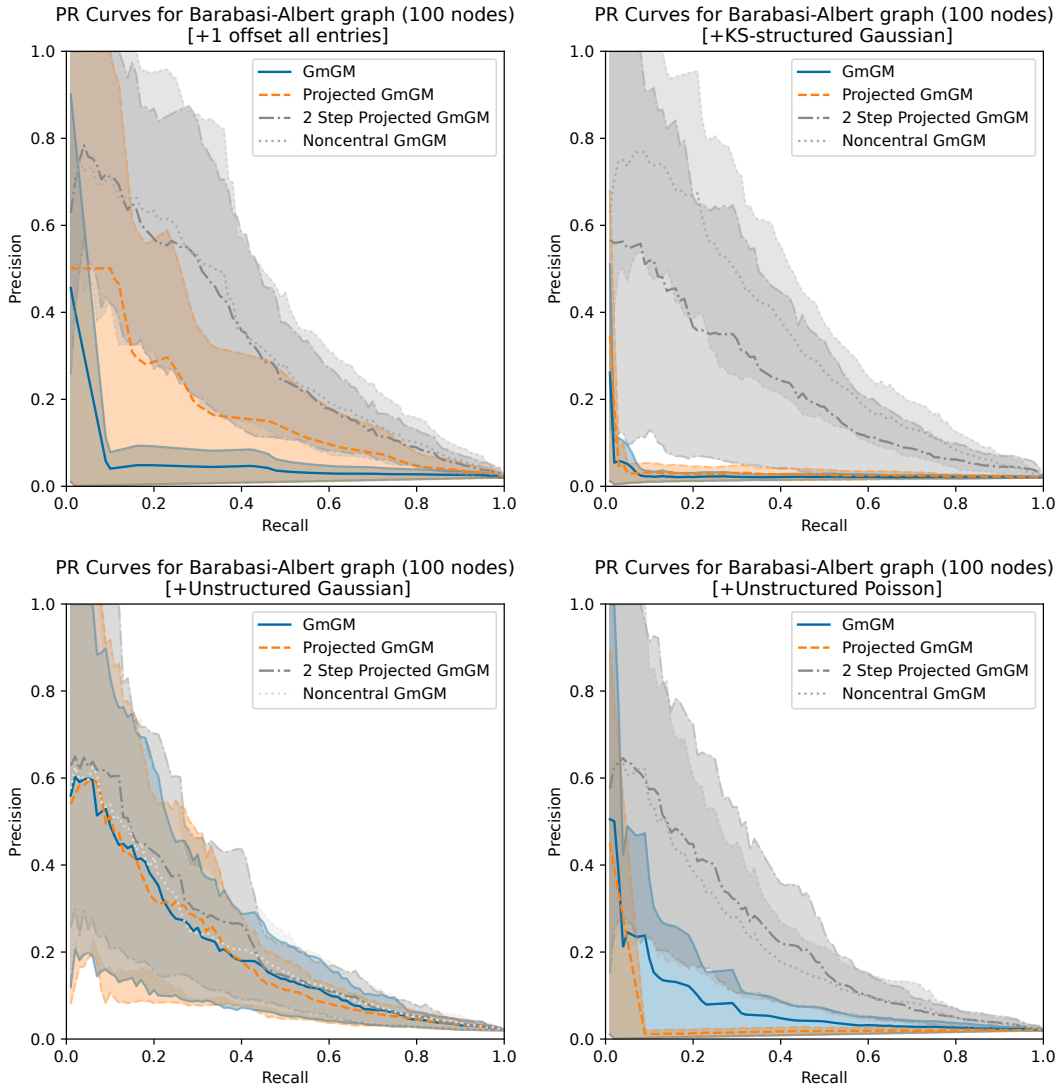


Figure B.2: Precision and recall on a synthetic dataset. Error bars are the best/worst performance over 10 trials; the center line is average performance. Here, we compare 0 iterations (GmGM), 1 iteration (Projected GmGM), 2 iterations (2-Step Projected GmGM), and until convergence (Noncentral GmGM).

Hartwig, R. E. (1976). Singular Value Decomposition and the Moore–Penrose Inverse of Bordered Matrices [Publisher: Society for Industrial and Applied Mathematics]. *SIAM Journal on Applied Mathematics*, *31*(1), 31–41. https://doi.org/10.1137/0131003

Holbrook, A. (2018). Differentiating the pseudo determinant. *Linear Algebra and its Applications*, *548*, 293–304. https://doi.org/10.1016/j.laa.2018.03.018

Kolda, T. G., & Bader, B. W. (2009). Tensor Decompositions and Applications [Publisher: Society for Industrial and Applied Mathematics]. *SIAM Review*, *51*(3), 455–500. https://doi.org/10.1137/07070111X

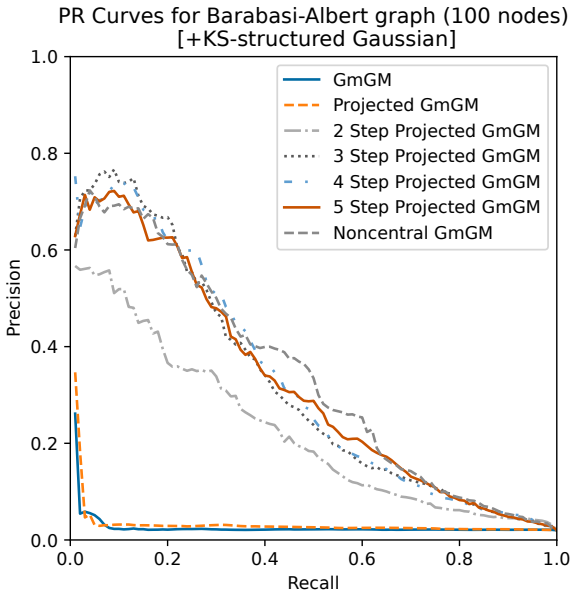


Figure B.3: Precision and recall on a synthetic dataset for various iterations of our algorithm. Confidence intervals omitted for clarity.

Appendix C

The Cartesian Gaussian additive noise model for directed network inference in omics data

Supplementary Material for the Strong Product Model for Network Inference without Independence Assumptions

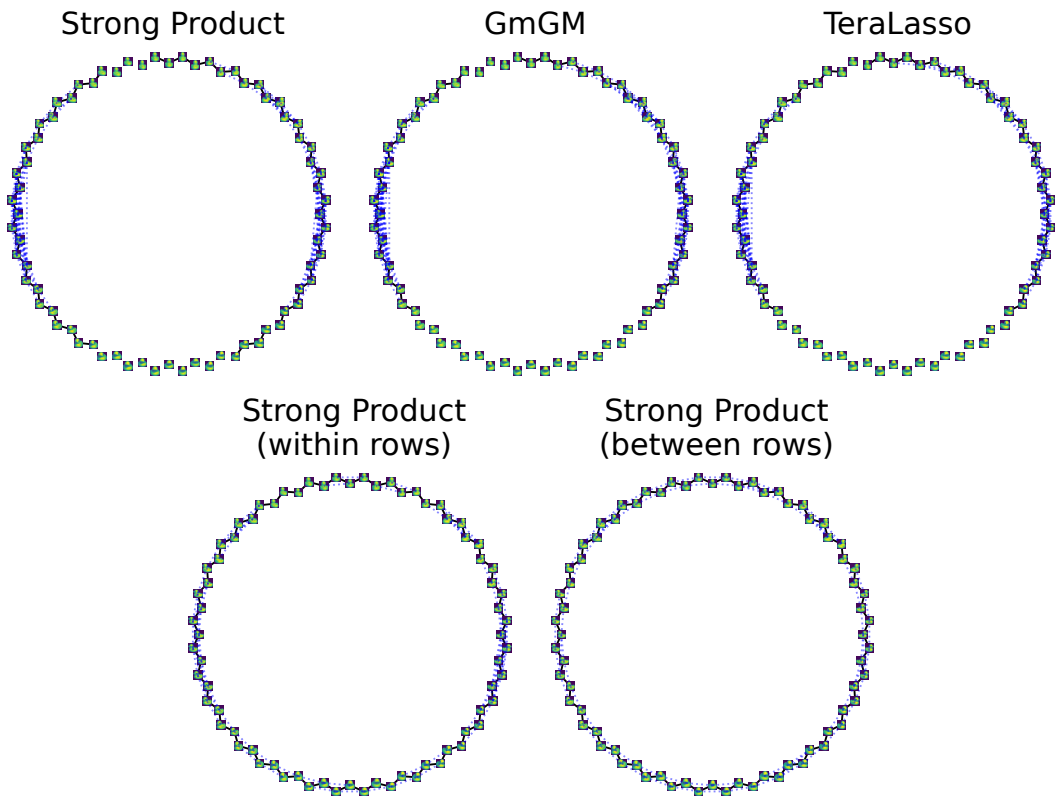


Figure C.1: The learned connections between the video frames for various algorithms. Black connections indicate connections between adjacent frames, blue indicate non-adjacent frames. This is an enlarged version of Figure 4 in the main paper.

C.1 Additional experiments

In this section, we briefly recount the additional experiments we ran. All data is generated identically to that in the main paper, except for the differences we note. Figure C.3 demonstrates that we do not experience a degradation in performance when we run our algorithm on Kronecker-sum-structured and Kronecker-product-structured data; we still perform comparably to prior work.

In Figure C.4, we demonstrate that performance improves as our factor graphs get larger. We analyze the AUPR as y varies for 100-node and 500-node factor graphs in Figure C.5. This latter experiment takes significantly longer to run than other experiments in this paper, so we do not repeat it for 1000-node graphs. Figure C.2 shows that performance is not affected by the value of ρ.

C.2 Derivation of gradients

In this section, we will derive the gradients of our negative log-likelihood, with respect to our three parameter matrices. The negative log-likelihood is given below, for reference, with irrelevant additive and multiplicative constants removed.

$$\begin{aligned}\text{NLL} &= -\log\left|\boldsymbol{\Psi}_1 \oplus \boldsymbol{\Psi}_{2,w} + \boldsymbol{\Psi}_1 \otimes \boldsymbol{\Psi}_{2,b}\right| + \operatorname{tr}\left[\boldsymbol{\Psi}_1 \mathbf{S}_1\right] + \operatorname{tr}\left[\boldsymbol{\Psi}_{2,w}\mathbf{S}_2\right] + \operatorname{tr}\left[\boldsymbol{\Psi}_1 \mathbf{X}\boldsymbol{\Psi}_{2,b}\mathbf{X}^T\right] \\ &= -\log\left|\boldsymbol{\Psi}_1 \otimes \boldsymbol{\Theta} + \mathbf{I} \otimes \boldsymbol{\Psi}_{2,w}\right| + \operatorname{tr}\left[\boldsymbol{\Psi}_{2,w}\mathbf{S}_2\right] + \operatorname{tr}\left[\boldsymbol{\Psi}_1 \mathbf{X}\boldsymbol{\Theta}\mathbf{X}^T\right]\end{aligned}$$

To derive the gradients, we will need to take derivatives with respect to matrices. Standard relations that we will use are provided below, for reference.

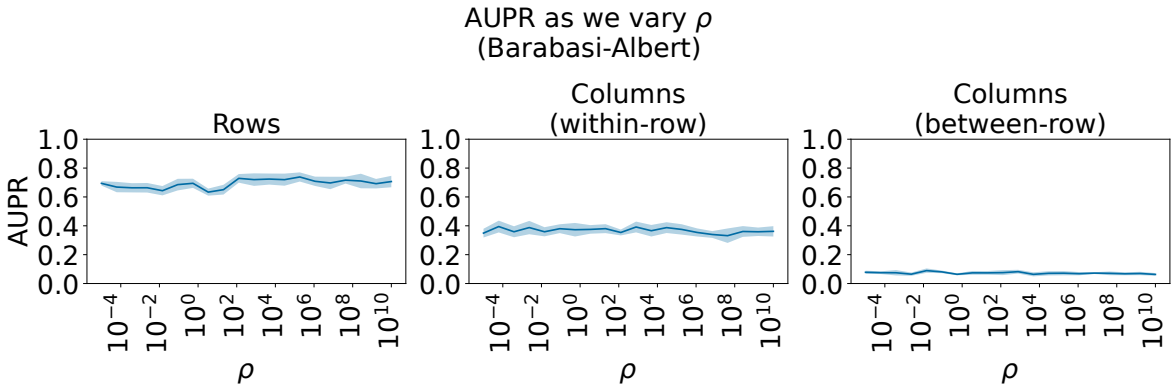


Figure C.2: The value of ρ does not significantly affect the results. Experiment performed on 100-node graphs with the standard 1:1 x ratio for data generated from the strong product model, with Barabasi-Albert factor graphs.

$$
\begin{aligned}
d\log|\mathbf{M}| &= \mathrm{tr}\left[\mathbf{M}^{-1}d\mathbf{M}\right] \\
\frac{d}{d\mathbf{M}}\mathrm{tr}\left[\mathbf{M}\mathbf{N}\right] &= \mathbf{N} \\
d\left(\mathbf{M}\otimes\mathbf{N}\right) &= d\mathbf{M}\otimes\mathbf{N} + \mathbf{M}\otimes d\mathbf{N} \\
d\left(\mathbf{M}\oplus\mathbf{N}\right) &= d\mathbf{M}\oplus d\mathbf{N}
\end{aligned}
$$

The $\oplus$ relation follows directly from the $\otimes$ relation. Using these equations, the derivative of he negative log likelihood is straightforward; we just need an expression for the derivative of the strong product.

$$
d\left(\mathbf{\Psi}_1\oplus\mathbf{\Psi}_{2,w}+\mathbf{\Psi}_1\otimes\mathbf{\Psi}_{2,b}\right) = d\mathbf{\Psi}_1\oplus d\mathbf{\Psi}_{2,w}+d\mathbf{\Psi}_1\otimes\mathbf{\Psi}_{2,b}+\mathbf{\Psi}_1\otimes d\mathbf{\Psi}_{2,b}
$$

Rather than differentiating with respect to entire matrices, we will differentiate with respect to a single element. We will let $\mathbf{J}^{ij}$ be the matrix of zeros except for a 1 at position (i, j), i.e. $\frac{d}{dx_{ij}}\mathbf{X}=\mathbf{J}^{ij}$.

$$
\begin{aligned}
\frac{d}{d\psi_{1,ij}}\left(\mathbf{\Psi}_1\oplus\mathbf{\Psi}_{2,w}+\mathbf{\Psi}_1\otimes\mathbf{\Psi}_{2,b}\right) &= \mathbf{J}^{ij}\oplus\mathbf{0}+\mathbf{J}^{ij}\otimes\mathbf{\Psi}_{2,b} \\
&= \mathbf{J}^{ij}\otimes\mathbf{I}+\mathbf{J}^{ij}\otimes\mathbf{\Psi}_{2,b} \\
&= \mathbf{J}^{ij}\otimes\mathbf{\Theta} \\
&= \left(\mathbf{I}\otimes\mathbf{\Theta}\right)\left(\mathbf{J}^{ij}\otimes\mathbf{I}\right)
\end{aligned}
$$

From the definition of the stridewise-blockwise trace, and letting sp abbreviate the strong product, we have that:

$$
\begin{aligned}
\frac{d}{d\mathbf{\Psi}_1}\log|sp| &= \mathrm{tr}_{d_2}\left[sp^{-1}\left(\mathbf{I}\otimes\mathbf{\Theta}\right)\right] \\
&= \mathrm{tr}_{d_2}\left[\left(\mathbf{\Psi}_1\oplus\mathbf{\Psi}_{2,w}\mathbf{\Theta}^{-1}\right)^{-1}\right]
\end{aligned}
$$

Thus, our derivative with respect to $\mathbf{\Psi}_1$ is just:

$$
\frac{d}{d\mathbf{\Psi}_1}\mathrm{NLL} = -\mathrm{tr}_{d_2}\left[\left(\mathbf{\Psi}_1\oplus\mathbf{\Psi}_{2,w}\mathbf{\Theta}^{-1}\right)^{-1}\right]+\mathbf{X}\mathbf{\Theta}\mathbf{X}^T
$$

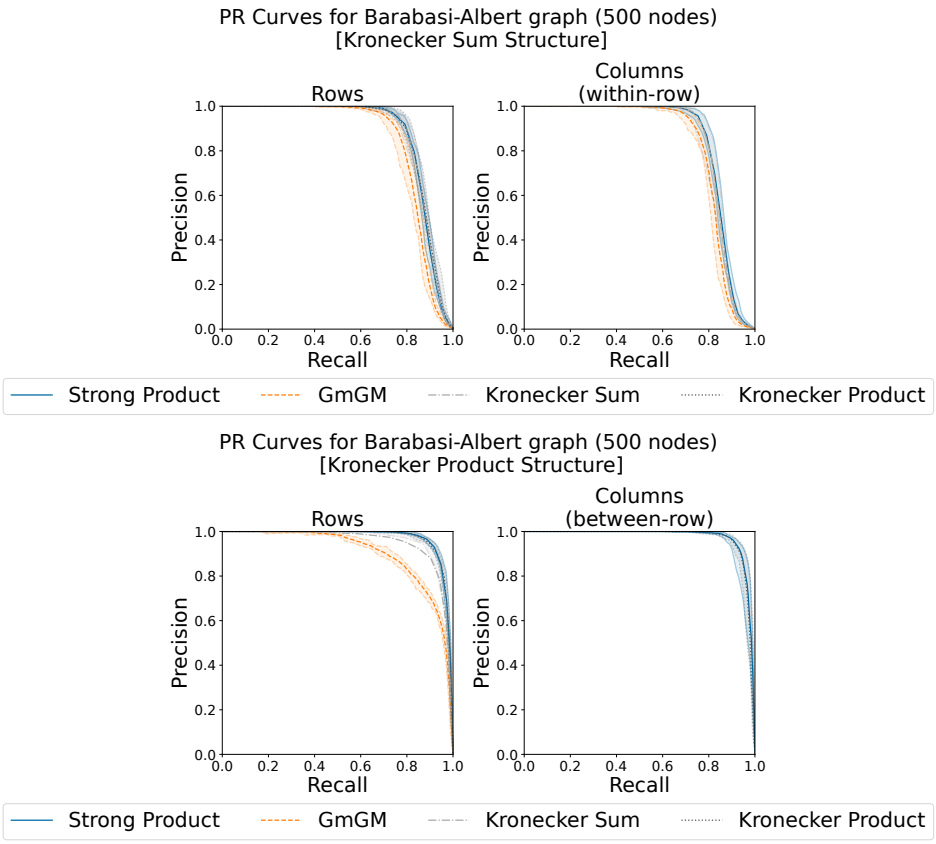


Figure C.3: Performance of algorithms when data is generated from the Kronecker sum or Kronecker product structures.

The gradient with respect to $\mathbf{\Psi}_{2,w}$ is fairly easy to compute.

$$\frac{d}{d\psi_{2,w;ij}}\left(\mathbf{\Psi}_1 \oplus \mathbf{\Psi}_{2,w} + \mathbf{\Psi}_1 \otimes \mathbf{\Psi}_{2,b}\right) = \mathbf{I} \otimes \mathbf{J}^{ij}$$

$$\frac{d}{d\mathbf{\Psi}_{2,w}}\text{NLL} = -\text{tr}^{d_1}\left[\left(\mathbf{\Psi}_1 \otimes \mathbf{\Theta} + \mathbf{I} \otimes \mathbf{\Psi}_{2,w}\right)^{-1}\right] + \mathbf{S}_2$$

We now have a choice to consider our method as a function of $\mathbf{\Psi}_{2,b}$ or $\mathbf{\Theta}$; so far, parameterizing in terms of $\mathbf{\Theta}$ has simplified our equations, so we will continue to do so.

$$\frac{d}{d\theta_{ij}}\left(\mathbf{\Psi}_1 \otimes \mathbf{\Theta} + \mathbf{I} \otimes \mathbf{\Psi}_{2,w}\right) = \mathbf{\Psi}_1 \otimes \mathbf{J}^{ij}$$

$$\begin{aligned}\frac{d}{d\mathbf{\Theta}}\text{NLL} &= -\text{tr}^{d_1}\left[\left(\mathbf{\Psi}_1 \otimes \mathbf{\Theta} + \mathbf{I} \otimes \mathbf{\Psi}_{2,w}\right)^{-1}\left(\mathbf{\Psi}_1 \otimes \mathbf{I}\right)\right] + \mathbf{X}^T\mathbf{\Psi}_1\mathbf{X} \\ &= -\text{tr}^{d_1}\left[\left(\mathbf{I} \otimes \mathbf{\Theta} + \mathbf{\Psi}_1^{-1} \otimes \mathbf{\Psi}_{2,w}\right)^{-1}\right] + \mathbf{X}^T\mathbf{\Psi}_1\mathbf{X}\end{aligned}$$

C.2.1 Efficient gradients

Earlier, we derived the following gradients:

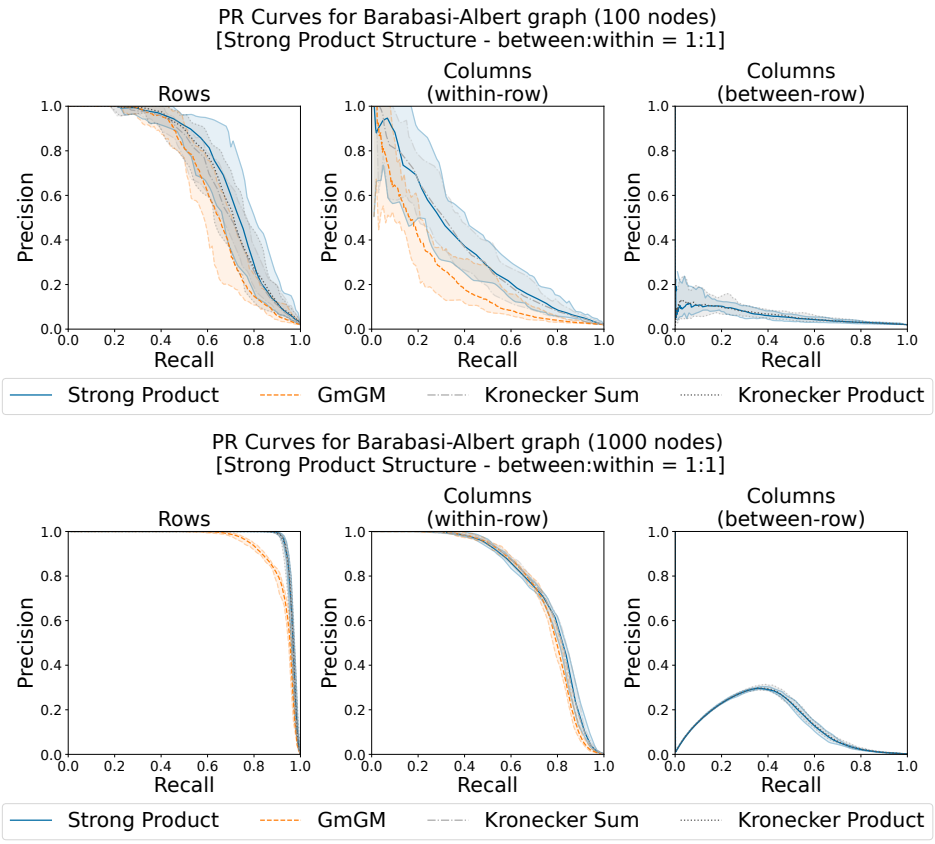


Figure C.4: The effect of different graph sizes on performance.

$$\frac{d}{d\mathbf{\Psi}_1}\text{NLL} = -\text{tr}_{d_2}\left[\left(\mathbf{\Psi}_1 \oplus \mathbf{\Psi}_{2,w}\mathbf{\Theta}^{-1}\right)^{-1}\right] + \mathbf{X}\mathbf{\Theta}\mathbf{X}^T$$
$$\frac{d}{d\mathbf{\Psi}_{2,w}}\text{NLL} = -\text{tr}^{d_1}\left[\left(\mathbf{\Psi}_1 \otimes \mathbf{\Theta} + \mathbf{I} \otimes \mathbf{\Psi}_{2,w}\right)^{-1}\right] + \mathbf{S}_2$$
$$\frac{d}{d\mathbf{\Theta}}\text{NLL} = -\text{tr}^{d_1}\left[\left(\mathbf{I} \otimes \mathbf{\Theta} + \mathbf{\Psi}_1^{-1} \otimes \mathbf{\Psi}_{2,w}\right)^{-1}\right] + \mathbf{X}^T\mathbf{\Psi}_1\mathbf{X}$$

These derivatives, while valid, have little practical utility; they require inverting a $d_1d_2 \times d_1d_2$ matrix, which is an $O(d_1^3d_2^3)$ operation that in practice requires $O(d_1^2d_2^2)$ space.

Because all parameters are positive definite, we know two key facts:

1. They each have an eigendecomposition
2. Every pair can be simultaneously diagonalized

In particular, we can express our parameters as follows:

$$\mathbf{\Psi}_1 = \mathbf{V}\mathbf{\Lambda}\mathbf{V}^T$$
$$\mathbf{\Psi}_{2,w} = \mathbf{P}\mathbf{D}\mathbf{P}^T$$
$$\mathbf{\Theta} = \mathbf{P}\mathbf{P}^\mathbf{T}$$

$\mathbf{V}$ is orthonormal, and $\mathbf{\Lambda}$ and $\mathbf{D}$ are diagonal. Importantly, $\mathbf{P}$ is **not orthonormal**. Furthermore, note that $\mathbf{\Psi}_{2,w}\mathbf{\Theta}^{-1} = \mathbf{P}\mathbf{D}\mathbf{P}^{-1}$. Computing these decompositions takes $O(d_1^3 + d_2^3)$ time and $O(d_1^2 + d_2^2)$

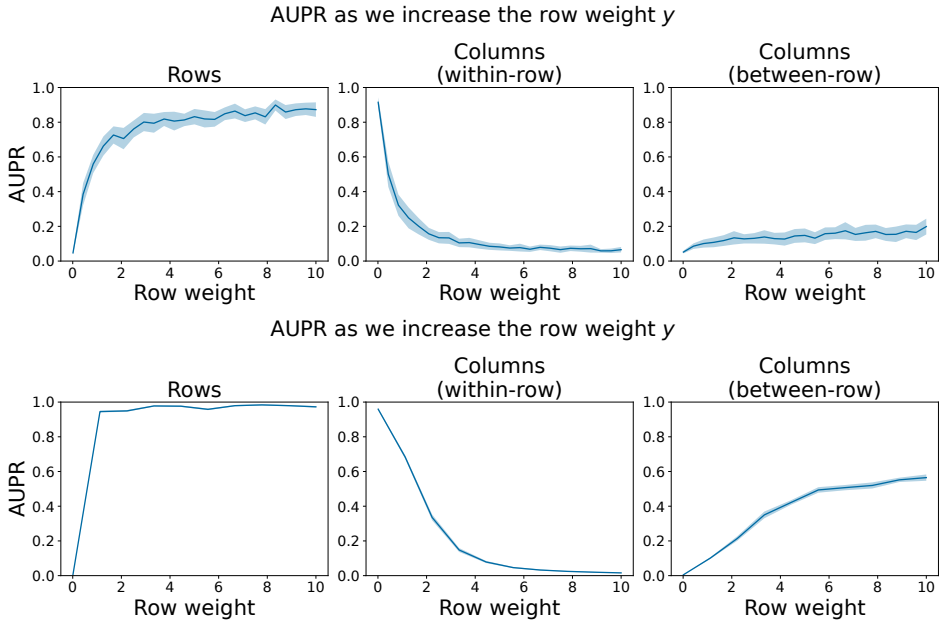


Figure C.5: Variance of AUPR as y changes for 100-node (top) and 500-node (bottom) graphs. Error bars represent standard deviations, computed over 50 trials (top) and 25 trials (bottom).

space. Thanks to a lemma by Dahl et al. (2013), we have a convenient way of accessing this diagonal core:

$$\mathrm{tr}_{d_2}\left[(\mathbf{M}\otimes\mathbf{I})\,\mathbf{A}\,(\mathbf{N}\otimes\mathbf{I})\right] = \mathbf{M}\mathrm{tr}_{d_2}\left[\mathbf{A}\right]\mathbf{N} \qquad \text{Lemma 2 (Dahl et al., 2013)}$$

$$\mathrm{tr}_{d_2}\left[(\mathbf{I}\otimes\mathbf{M})\,\mathbf{A}\,(\mathbf{I}\otimes\mathbf{N})\right] = \mathrm{tr}_{d_2}\left[(\mathbf{I}\otimes\mathbf{NM})\,\mathbf{A}\right] \qquad \text{(Cyclic Property)}$$

The latter fact follows directly from the definition of the stridewise-blockwise trace. Analogous formulas hold for tr^{d_1}, as long as the order of the Kronecker products is reversed. Using these, we can re-express our gradients.

$$
\begin{aligned}
\frac{d}{d\boldsymbol{\Psi}_1}\text{NLL} &= -\text{tr}_{d_2}\left[\left(\boldsymbol{\Psi}_1 \oplus \boldsymbol{\Psi}_{2,w}\boldsymbol{\Theta}^{-1}\right)^{-1}\right] + \mathbf{X}\boldsymbol{\Theta}\mathbf{X}^T \\
&= -\text{tr}_{d_2}\left[\left(\mathbf{V}\boldsymbol{\Lambda}\mathbf{V}^T \otimes \mathbf{I} + \mathbf{I} \otimes \mathbf{P}\mathbf{D}\mathbf{P}^{-1}\right)^{-1}\right] + \mathbf{X}\boldsymbol{\Theta}\mathbf{X}^T \\
&= -\text{tr}_{d_2}\left[(\mathbf{V}\otimes\mathbf{P})\left(\boldsymbol{\Lambda}\otimes\mathbf{V}^T\mathbf{V} + \mathbf{P}^{-1}\mathbf{P}\otimes\mathbf{D}\right)^{-1}(\mathbf{V}\otimes\mathbf{P})^{-1}\right] + \mathbf{X}\boldsymbol{\Theta}\mathbf{X}^T \\
&= -\mathbf{V}\text{tr}_{d_2}\left[(\boldsymbol{\Lambda}\otimes\mathbf{I} + \mathbf{I}\otimes\mathbf{D})^{-1}\right]\mathbf{V}^T + \mathbf{X}\boldsymbol{\Theta}\mathbf{X}^T \\
&= -\mathbf{V}\text{tr}_{d_2}\left[(\boldsymbol{\Lambda}\oplus\mathbf{D})^{-1}\right]\mathbf{V}^T + \mathbf{X}\boldsymbol{\Theta}\mathbf{X}^T \\
\frac{d}{d\boldsymbol{\Psi}_{2,w}}\text{NLL} &= -\text{tr}^{d_1}\left[\left(\boldsymbol{\Psi}_1\otimes\boldsymbol{\Theta} + \mathbf{I}\otimes\boldsymbol{\Psi}_{2,w}\right)^{-1}\right] + \mathbf{S}_2 \\
&= -\text{tr}^{d_1}\left[\left(\mathbf{I}\otimes\mathbf{P}^{-T}\right)(\boldsymbol{\Lambda}\otimes\mathbf{I} + \mathbf{I}\otimes\mathbf{D})^{-1}\left(\mathbf{I}\otimes\mathbf{P}^{-1}\right)\right] + \mathbf{S}_2 \\
&= -\mathbf{P}^{-T}\text{tr}^{d_1}\left[(\boldsymbol{\Lambda}\oplus\mathbf{D})^{-1}\right]\mathbf{P}^{-1} + \mathbf{S}_2 \\
\frac{d}{d\boldsymbol{\Theta}}\text{NLL} &= -\text{tr}^{d_1}\left[\left(\mathbf{I}\otimes\boldsymbol{\Theta} + \boldsymbol{\Psi}_1^{-1}\otimes\boldsymbol{\Psi}_{2,w}\right)^{-1}\right] + \mathbf{X}^T\boldsymbol{\Psi}_1\mathbf{X} \\
&= -\mathbf{P}^{-T}\text{tr}^{d_1}\left[\left(\mathbf{I}\otimes\mathbf{I} + \boldsymbol{\Lambda}^{-1}\otimes\mathbf{D}\right)^{-1}\right]\mathbf{P}^{-1} + \mathbf{X}^T\boldsymbol{\Psi}_1\mathbf{X} \\
&= -\mathbf{P}^{-T}\text{tr}^{d_1}\left[(\boldsymbol{\Lambda}\otimes\mathbf{I} + \mathbf{I}\otimes\mathbf{D})^{-1}(\boldsymbol{\Lambda}\otimes\mathbf{I})\right]\mathbf{P}^{-1} + \mathbf{X}^T\boldsymbol{\Psi}_1\mathbf{X} \\
&= -\mathbf{P}^{-T}\text{tr}^{d_1}\left[(\boldsymbol{\Lambda}\oplus\mathbf{D})^{-1}(\boldsymbol{\Lambda}\otimes\mathbf{I})\right]\mathbf{P}^{-1} + \mathbf{X}^T\boldsymbol{\Psi}_1\mathbf{X}
\end{aligned}
$$

Computing the stridewise-blockwise trace can now be done much more efficiently, as its input is a diagonal matrix. It can be done in $O(d_1 d_2)$ space. As a sanity check, we can observe that when $\boldsymbol{\Theta} = \mathbf{I}$ (i.e. $\boldsymbol{\Psi}_{2,b} = \mathbf{0}$), the gradients become those of the Kronecker-sum model (note that $\mathbf{P}$ can in that case be chosen to be the eigenvectors of $\boldsymbol{\Psi}_{2,w}$), and when $\boldsymbol{\Psi}_1 = \mathbf{I}$ then the other gradients become identical. Both of these should be expected from the structure of the strong product.

C.2.2 Addition of frobenius norm of kogarithm

Since $\frac{d}{d\mathbf{M}}\|\log\mathbf{M}\|_F = 2\mathbf{M}^{-1}\log\mathbf{M}$, we have that our gradients are:

$$
\begin{aligned}
\frac{d}{d\boldsymbol{\Psi}_1}\text{NLL} &= -\mathbf{V}\text{tr}_{d_2}\left[(\boldsymbol{\Lambda}\oplus\mathbf{D})^{-1}\right]\mathbf{V}^T + \mathbf{X}\boldsymbol{\Theta}\mathbf{X}^T + 2\rho\boldsymbol{\Psi}_1^{-1}\log\boldsymbol{\Psi}_1 \\
\frac{d}{d\boldsymbol{\Psi}_{2,w}}\text{NLL} &= -\mathbf{P}^{-T}\text{tr}^{d_1}\left[(\boldsymbol{\Lambda}\oplus\mathbf{D})^{-1}\right]\mathbf{P}^{-1} + \mathbf{S}_2 + 2\rho\boldsymbol{\Psi}_{2,w}^{-1}\log\boldsymbol{\Psi}_{2,w} \\
\frac{d}{d\boldsymbol{\Theta}}\text{NLL} &= -\mathbf{P}^{-T}\text{tr}^{d_1}\left[(\boldsymbol{\Lambda}\oplus\mathbf{D})^{-1}(\boldsymbol{\Lambda}\otimes\mathbf{I})\right]\mathbf{P}^{-1} + \mathbf{X}^T\boldsymbol{\Psi}_1\mathbf{X} + 2\rho\boldsymbol{\Theta}^{-1}\log\boldsymbol{\Theta}
\end{aligned}
$$

C.2.3 Riemannian gradients

Recall from the main paper the definition of the Riemannian gradient and retraction.

$$
\begin{aligned}
\text{grad}^{\mathbb{S}_{++}}_{\mathbf{M}} f &= \mathbf{M}\left(\text{grad}^{\mathbb{E}}_{\mathbf{M}} f\right)\mathbf{M} && \text{(gradient)} \\
\text{R}^{\mathbb{S}_{++}}_{\mathbf{M}}(\boldsymbol{\xi}) &= \mathbf{M}e^{\mathbf{M}^{-1}\boldsymbol{\xi}} && \text{(retraction)} \\
\mathbf{M}_{t+1} &= \text{R}^{\mathbb{S}_{++}}_{\mathbf{M}_t}\left(-\eta_t\,\text{grad}^{\mathbb{S}_{++}}_{\mathbf{M}_t} f\right) && \text{(update)}
\end{aligned}
$$

There is ample opportunity for simplification. Note that:

$$\begin{aligned}\mathbf{M}_{t+1} &= \mathbf{M}_t e^{\left(-\eta_t \mathrm{grad}^{\mathbb{E}}_{\mathbf{M}_t} f\right)\mathbf{M}_t} \\ \mathbf{M}^{-1}\log\mathbf{M} &= (\log\mathbf{M})\,\mathbf{M}^{-1}\end{aligned}$$

The second statement follows from simultaneous diagonalizability. What happens when we right-multiply our Euclidean gradients by $\mathbf{M}$?

$$\begin{aligned}\left(\frac{d}{d\mathbf{\Psi}_1}\mathrm{NLL}\right)\mathbf{\Psi}_1 &= -\mathbf{V}\mathrm{tr}_{d_2}\left[(\mathbf{\Lambda}\oplus\mathbf{D})^{-1}\right]\mathbf{\Lambda}\mathbf{V}^T + \mathbf{X}\mathbf{\Theta}\mathbf{X}^T\mathbf{\Psi}_1 + 2\rho\log\mathbf{\Psi}_1 \\ \left(\frac{d}{d\mathbf{\Psi}_{2,w}}\mathrm{NLL}\right)\mathbf{\Psi}_{2,w} &= -\mathbf{P}^{-T}\mathrm{tr}^{d_1}\left[(\mathbf{\Lambda}\oplus\mathbf{D})^{-1}\right]\mathbf{D}\mathbf{P}^T + \mathbf{S}_2\mathbf{\Psi}_{2,w} + 2\rho\log\mathbf{\Psi}_{2,w} \\ \left(\frac{d}{d\mathbf{\Theta}}\mathrm{NLL}\right)\mathbf{\Theta} &= -\mathbf{P}^{-T}\mathrm{tr}^{d_1}\left[(\mathbf{\Lambda}\oplus\mathbf{D})^{-1}(\mathbf{\Lambda}\otimes\mathbf{I})\right]\mathbf{P}^T + \mathbf{X}^T\mathbf{\Psi}_1\mathbf{X}\mathbf{\Theta} + 2\rho\log\mathbf{\Theta}\end{aligned}$$

Note that this is actually (slightly) cheaper to compute than the unmodified gradient, since diagonal matrix multiplications are cheap, we removed many matrix multiplications, and most of the few matrix multiplications we did add are duplicates.

C.3 Efficient evaluation of log-likelihood

Recall that the negative log likelihood is as follows, ignoring irrelevant constants.

$$\mathrm{NLL} = -\log\left|\mathbf{\Psi}_1\otimes\mathbf{\Theta} + \mathbf{I}\otimes\mathbf{\Psi}_{2,w}\right| + \mathrm{tr}\left[\mathbf{\Psi}_{2,w}\mathbf{S}_2\right] + \mathrm{tr}\left[\mathbf{\Psi}_1\mathbf{X}\mathbf{\Theta}\mathbf{X}^T\right]$$

For calculating our gradients, we already used certain decompositions, which we can re-use in the evaluation of our NLL. Evaluating the NLL is useful for convergence checking and line searching.

$$\begin{aligned}\mathrm{NLL} &= -\log|\mathbf{I}\otimes\mathbf{P}|\,|\mathbf{\Lambda}\otimes\mathbf{I}+\mathbf{I}\otimes\mathbf{D}|\left|\mathbf{I}\otimes\mathbf{P}^T\right| + \mathrm{tr}\left[\mathbf{\Psi}_{2,w}\mathbf{S}_2\right] + \mathrm{tr}\left[\mathbf{\Psi}_1\mathbf{X}\mathbf{\Theta}\mathbf{X}^T\right] \\ &= -\log|\mathbf{\Lambda}\otimes\mathbf{I}+\mathbf{I}\otimes\mathbf{D}|\,|\mathbf{\Theta}|^{d_1} + \mathrm{tr}\left[\mathbf{\Psi}_{2,w}\mathbf{S}_2\right] + \mathrm{tr}\left[\mathbf{\Psi}_1\mathbf{X}\mathbf{\Theta}\mathbf{X}^T\right] \\ &= -\log|\mathbf{\Lambda}\oplus\mathbf{D}| - d_1\log|\mathbf{\Theta}| + \mathrm{tr}\left[\mathbf{\Psi}_{2,w}\mathbf{S}_2\right] + \mathrm{tr}\left[\mathbf{\Psi}_1\mathbf{X}\mathbf{\Theta}\mathbf{X}^T\right]\end{aligned}$$

The two trace terms require taking traces of matrices already computed for the gradients, so could be re-used. The log determinant of a Kronecker sum of diagonal matrices is easy to compute (sum of the logs of the diagonals); thus, the cost reduces to creating the decompositions and finding the determinant of $\mathbf{\Theta}$, both of which are expensive operations but are comparable to the cost of gradient evaluation. In fact, the Cholesky factors of $\mathbf{\Theta}$ are computed during simultaneous diagonalization, making determinant computation easy.

Bibliography for Paper 3 (appendix)

Dahl, A., Hore, V., Iotchkova, V., & Marchini, J. (2013, December). Network inference in matrix-variate Gaussian models with non-independent noise [arXiv:1312.1622 [stat]]. https://doi.org/10.48550/arXiv.1312.1622

Appendix D

The Cartesian Gaussian additive noise model for directed network inference in omics data

Supplementary Material for the Strong Product Model for Network Inference without Independence Assumptions

Overview

In the supplementary material, we cover how to derive the likelihood from the generative process (Section D.1) as well as derivation of the gradients (Section D.2). We also cover the proofs of Theorems 6-9 in Section D.3.

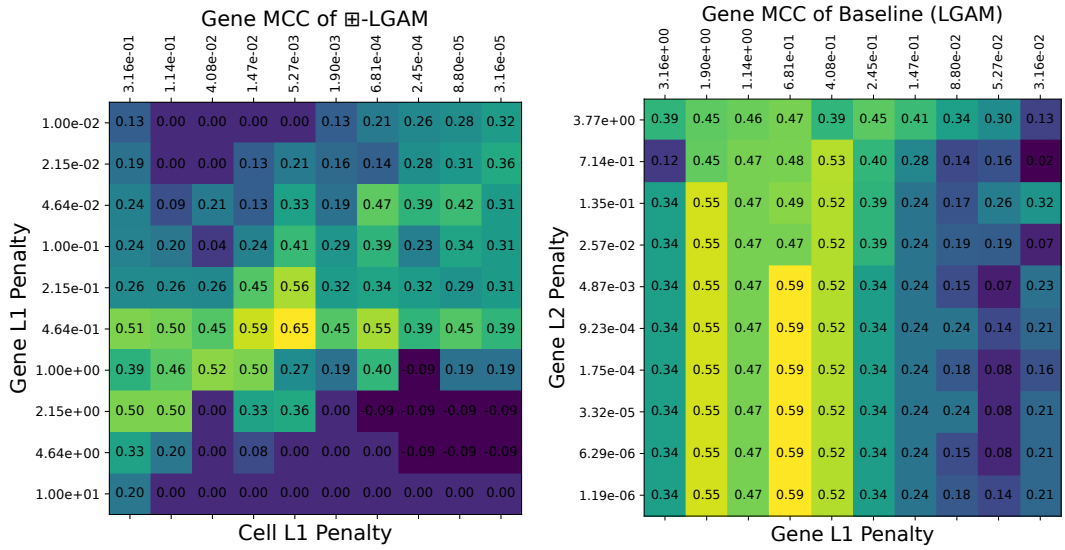


Figure D.1: (Left) MCC performance on the gene graph as we vary the axis-wise L1 penalties in our ⊞-LGAM model. (Right) MCC performance on the gene graph as we vary the gene L1 and L2 penalties in the baseline LGAM model.

D.1 Model derivation

In this section, we derive the joint distribution of data drawn from the generative process of the ⊞-LGAM model.

$$\mathcal{X} = \left(\sum_{\ell}^{K} \mathcal{X} \times_\ell \mathbf{C}_\ell^T\right) + \mathcal{E} \qquad \text{(generative process)}$$

$$\iff \sum_{\ell}^{K} \frac{1}{K}\mathcal{X} \times_\ell \mathbf{I} = \left(\sum_{\ell}^{K} \mathcal{X} \times_\ell \mathbf{C}_\ell^T\right) + \mathcal{E}$$

$$\iff \sum_{\ell}^{K} \mathcal{X} \times_\ell \left(\frac{1}{K}\mathbf{I} - \mathbf{C}_\ell^T\right) = \mathcal{E}$$

$$\iff \left(\bigoplus_{\ell} \left(\frac{1}{K}\mathbf{I} - \mathbf{C}_\ell^T\right)\right) \text{vec}\left[\mathcal{X}\right] = \text{vec}\left[\mathcal{E}\right] \qquad \text{(Sylvester Equation)}$$

$$\iff \text{vec}\left[\mathcal{X}\right] \sim \mathcal{N}\left(\mathbf{0}, \left(\bigoplus_{\ell} \left(\frac{1}{K}\mathbf{I} - \mathbf{C}_\ell^T\right)\right)^{-1}\left(\bigoplus_{\ell} \left(\frac{1}{K}\mathbf{I} - \mathbf{C}_\ell^T\right)\right)^{-T}\right)$$

Letting $\mathbf{L}_\ell = \sqrt{d_{\setminus \ell}}\left(\frac{1}{K}\mathbf{I} - \mathbf{C}_\ell\right)$ and $\mathbf{L} = \bigoplus_\ell \frac{1}{\sqrt{d_{\setminus \ell}}}\mathbf{L}_\ell$, we then have the expression used in the main paper:

$$\text{vec}\left[\mathcal{X}\right] \sim \mathcal{N}\left(\mathbf{0}, \mathbf{L}^{-T}\mathbf{L}^{-1}\right)$$

Note that $\left(\mathbf{L}^{-T}\mathbf{L}^{-1}\right)^{-1} = \mathbf{L}\mathbf{L}^T$, and hence $\mathbf{L}$ is the Cholesky decomposition of the precision matrix.

As an aside, note that our first step involved splitting $\mathcal{X}$ into a sum $\sum_\ell^K \frac{1}{K}\mathcal{X}$. We could have split it differently; $\mathcal{X} = \sum_\ell^K c_\ell \mathcal{X}$ is valid for any set of constants c_ℓ where $\sum_\ell c_\ell = 1$. This is a manifestation of the non-identifiability of the Kronecker sum mentioned in the background! It doesn't affect the structure of the networks.

D.2 Gradient derivation

Here we'll find the gradient for the non-regularizer portion of our objective function. We give the gradient in the full tensor-variate generality, which is notationally cumbersome. See Gradient 3 at the end of this section for the gradient when the data is only matrix-variate, the most common and most easily expressible case. We repeat our objective below for convenience.

$$\hat{\mathbf{L}} = \underset{\mathbf{L}\in\mathbb{L}_{++}^{\oplus}}{\operatorname{argmin}} \operatorname{tr}\left[\mathbf{L}\mathbf{L}^T\mathbf{S}\right] - \log\det\left[\mathbf{L}\mathbf{L}^T\right] + \sum_i \left(\rho_i\|\mathbf{L}^{(i)}\|_F^2 + \lambda_i\|\mathbf{L}^{(i)}\|_{od}\right)$$

We'll break the gradient up term-by-term. It is difficult to express, so we will introduce some indexing notation. $\mathcal{X}_{ab\mathbf{c}}$ indexes $\mathcal{X}$ such that it grabs the ath slice along the first axis, the bth slice along the second, and the $\mathbf{c}_i$th slice along axis $i+2$. $\mathbf{X}_{\mathbf{c}}$ is the matrix picked out by $\mathbf{c}$: $[\mathcal{X}_{ab\mathbf{c}}]_{ab}$. Furthermore, $\mathbf{X}_{\mathbf{c}}^{(i;j)}$ will denote the matrix where $\mathbf{c}$ indexes all axes but the ith and the jth. If $\mathbf{X}^{(j)}$ appears without a subscript, then it indicates the *matricization* of $\mathcal{X}$: the reshaping of it into a $d_j \times d_{\setminus j}$ matrix by flattening all non-j axes.

Gradient 1. *The gradient of* $\operatorname{tr}\left[\mathbf{L}\mathbf{L}^T\mathbf{S}\right]$ *over* $\mathbb{L}_{++}^{\oplus}$ *with respect to* $\mathbf{L}^{(\ell)}$ *is:*

$$\frac{2}{d_{\setminus\ell}}\mathbf{X}^{(\ell)}\mathbf{X}^{(\ell),T}\mathbf{L}^{(\ell)} + \sum_{\mathbf{c},\ell'\neq\ell}\frac{2}{\sqrt{d_{\setminus\ell}d_{\setminus\ell'}}}\mathbf{X}_{\mathbf{c}}^{(\ell;\ell')}\mathbf{L}^{(\ell'),T}\mathbf{X}_{\mathbf{c}}^{(\ell;\ell'),T}$$

Proof. Recall the representation $\mathbf{L} = \bigoplus_\ell \frac{1}{\sqrt{d_{\setminus\ell}}}\mathbf{L}^{(\ell)}$.

$$\begin{aligned}
\operatorname{tr}\left[\mathbf{L}\mathbf{L}^T\mathbf{S}\right] &= \operatorname{tr}\left[\left(\bigoplus_\ell \frac{1}{\sqrt{d_{\setminus\ell}}}\mathbf{L}^{(\ell)}\right)\left(\bigoplus_\ell \frac{1}{\sqrt{d_{\setminus\ell}}}\mathbf{L}^{(\ell)}\right)^T \operatorname{vec}\mathcal{X}\operatorname{vec}\mathcal{X}^T\right] \\
&= \operatorname{tr}\left[\operatorname{vec}\left[\sum_\ell \mathcal{X}\times_\ell \frac{1}{\sqrt{d_{\setminus\ell}}}\mathbf{L}^{(\ell),T}\right]\operatorname{vec}\left[\sum_\ell \mathcal{X}\times_\ell \frac{1}{\sqrt{d_{\setminus\ell}}}\mathbf{L}^{(\ell),T}\right]^T\right] \\
&= \operatorname{tr}\left[\sum_\ell \operatorname{vec}\left[\mathcal{X}\times_\ell \frac{1}{\sqrt{d_{\setminus\ell}}}\mathbf{L}^{(\ell),T}\right]\sum_\ell \operatorname{vec}\left[\mathcal{X}\times_\ell \frac{1}{\sqrt{d_{\setminus\ell}}}\mathbf{L}^{(\ell),T}\right]^T\right] \\
&= \sum_{\ell\ell'}\operatorname{tr}\left[\operatorname{vec}\left[\mathcal{X}\times_\ell \frac{1}{\sqrt{d_{\setminus\ell}}}\mathbf{L}^{(\ell),T}\right]\operatorname{vec}\left[\mathcal{X}\times_{\ell'} \frac{1}{\sqrt{d_{\setminus\ell'}}}\mathbf{L}^{(\ell'),T}\right]^T\right] \\
&= \sum_{\ell\ell' k}\operatorname{vec}\left[\mathcal{X}\times_\ell \frac{1}{\sqrt{d_{\setminus\ell}}}\mathbf{L}^{(\ell),T}\right]_k \operatorname{vec}\left[\mathcal{X}\times_{\ell'} \frac{1}{\sqrt{d_{\setminus\ell'}}}\mathbf{L}^{(\ell'),T}\right]_k
\end{aligned}$$

Without loss of generality, let's look at what happens for the first two axes.

$$\begin{aligned}
\left[\mathcal{X} \times_1 \mathbf{P}^T\right]_{ab\mathbf{c}} &= \sum_i \mathcal{X}_{ib\mathbf{c}}\mathbf{P}_{ai} && \text{(definition)} \\
\sum_{ab\mathbf{c}} \left[\mathcal{X} \times_1 \mathbf{P}^T\right]_{ab\mathbf{c}} \left[\mathcal{X} \times_1 \mathbf{P}^T\right]_{ab\mathbf{c}} &= \sum_{ab\mathbf{c}}\sum_{ij} \mathbf{P}_{ai}\mathbf{P}_{aj}\mathcal{X}_{ib\mathbf{c}}\mathcal{X}_{jb\mathbf{c}} \\
&= \sum_{b\mathbf{c}}\sum_{aij} \mathbf{P}_{ai}\mathcal{X}_{i(b\mathbf{c})}\mathcal{X}_{j(b\mathbf{c})}\mathbf{P}_{aj} \\
&= \sum_{b\mathbf{c}} \text{tr}\left[\mathbf{P}\mathbf{P}^T\mathbf{X}_{b\mathbf{c}}\mathbf{X}_{b\mathbf{c}}^T\right] \\
&= \text{tr}\left[\mathbf{P}\mathbf{P}^T\sum_{b\mathbf{c}}\mathbf{X}_{b\mathbf{c}}\mathbf{X}_{b\mathbf{c}}^T\right] \\
&= \text{tr}\left[\mathbf{P}\mathbf{P}^T\mathbf{X}^{(1)}\mathbf{X}^{(1),T}\right] \\
\left[\mathcal{X} \times_2 \mathbf{Q}^T\right]_{ab\mathbf{c}} &= \sum_i \mathcal{X}_{ai\mathbf{c}}\mathbf{Q}_{bi} && \text{(definition)} \\
\sum_{ab\mathbf{c}} \left[\mathcal{X} \times_1 \mathbf{P}^T\right]_{ab\mathbf{c}} \left[\mathcal{X} \times_2 \mathbf{Q}^T\right]_{ab\mathbf{c}} &= \sum_{ab\mathbf{c}}\sum_{ij} \mathbf{P}_{ai}\mathcal{X}_{ib\mathbf{c}}\mathbf{Q}_{bj}\mathcal{X}_{aj\mathbf{c}} \\
&= \sum_{\mathbf{c}} \text{tr}\left[\mathbf{P}\mathbf{X}_{\mathbf{c}}\mathbf{Q}\mathbf{X}_{\mathbf{c}}^T\right]
\end{aligned}$$

Returning to our equality from before:

$$\text{tr}\left[\mathbf{L}\mathbf{L}^T\mathbf{S}\right] = \sum_{\ell\ell' k} \text{vec}\left[\mathcal{X} \times_\ell \frac{1}{\sqrt{d_{\setminus\ell}}}\mathbf{L}^{(\ell),T}\right]_k \text{vec}\left[\mathcal{X} \times_{\ell'} \frac{1}{\sqrt{d_{\setminus\ell'}}}\mathbf{L}^{(\ell'),T}\right]_k$$

Let's look at just the terms involving $\mathbf{L}^{(1)}$:

$$\begin{aligned}
&\sum_k \text{vec}\left[\mathcal{X} \times_1 \frac{1}{\sqrt{d_{\setminus 1}}}\mathbf{L}^{(1),T}\right]_k \text{vec}\left[\mathcal{X} \times_1 \frac{1}{\sqrt{d_{\setminus 1}}}\mathbf{L}^{(1),T}\right]_k \\
+&\sum_{\ell' k} \text{vec}\left[\mathcal{X} \times_1 \frac{1}{\sqrt{d_{\setminus 1}}}\mathbf{L}^{(1),T}\right]_k \text{vec}\left[\mathcal{X} \times_{\ell'} \frac{1}{\sqrt{d_{\setminus\ell'}}}\mathbf{L}^{(\ell'),T}\right]_k \\
+&\sum_{\ell k} \text{vec}\left[\mathcal{X} \times_\ell \frac{1}{\sqrt{d_{\setminus\ell}}}\mathbf{L}^{(\ell),T}\right]_k \text{vec}\left[\mathcal{X} \times_1 \frac{1}{\sqrt{d_{\setminus 1}}}\mathbf{L}^{(1),T}\right]_k
\end{aligned}$$

Which we can rewrite as:

$$\frac{1}{d_{\setminus 1}}\text{tr}\left[\mathbf{L}^{(1)}\mathbf{L}^{(1),T}\mathbf{X}^{(1)}\mathbf{X}^{(1),T}\right] + \sum_{\mathbf{c},\ell\neq 1} \frac{2}{\sqrt{d_{\setminus\ell}d_{\setminus 1}}}\text{tr}\left[\mathbf{L}^{(1)}\mathbf{X}_{\mathbf{c}}^{(\ell)}\mathbf{L}^{(1;\ell)}\mathbf{X}_{\mathbf{c}}^{(1;\ell),T}\right]$$

Differentiation w.r.t. $\mathbf{L}^{(1)}$ then follows from standard derivative-of-trace properties:

$$\frac{2}{d_{\setminus 1}}\mathbf{X}^{(1)}\mathbf{X}^{(1),T}\mathbf{L}^{(1)} + \sum_{\mathbf{c},\ell\neq 1} \frac{2}{\sqrt{d_{\setminus\ell}d_{\setminus 1}}}\mathbf{X}_{\mathbf{c}}^{(1;\ell)}\mathbf{L}^{(\ell),T}\mathbf{X}_{\mathbf{c}}^{(1;\ell),T}$$

We can then repeat the same analysis for each axis.

□

We can see that the $\mathcal{X}$-dependent portion of the gradient is quadratic in $\mathcal{X}$ - this observation will be useful when we get to proving statistical recovery rates (Theorem 6), so let's take a moment to understand the properties of this quadratic.

Lemma 8. *Let* $\mathrm{vec}\,[\mathcal{X}]^T\,\mathbf{A}^{(i,j,\ell)}\mathrm{vec}\,[\mathcal{X}]$ *be the quadratic form in the gradient term of* $\mathbf{L}_{ij}^{(\ell)}$*. Let* R_{op}, R_F *be the maximal operator and Frobenius norms of any factor. Then:*

$$\|\mathbf{A}^{(i,j,\ell)}\|_{op} \leq \sum_{\ell'} \frac{2R_{op}}{\sqrt{d_{\setminus\ell}d_{\setminus\ell'}}}$$
$$\|\mathbf{A}^{(i,j,\ell)}\|_F \leq 2R_F\left(\frac{1}{\sqrt{d_{\setminus\ell}}} + \sum_{\ell'\neq\ell}\frac{1}{\sqrt{d_\ell d_{\ell'}}}\right)$$

Proof. The (i,j)th element first term of the quadratic, $\mathbf{X}^{(\ell)}\mathbf{X}^{(\ell),T}\mathbf{L}^{(\ell)}$, can be written as $\mathbf{e}_i^T\mathbf{X}^{(\ell)}\mathbf{X}^{(\ell),T}\mathbf{L}^{(\ell)}\mathbf{e}_j$. Let's massage it into a more convenient form:

$$\begin{aligned}
\mathbf{e}_i^T\mathbf{X}^{(\ell)}\mathbf{X}^{(\ell),T}\mathbf{L}^{(\ell)}\mathbf{e}_j &= \mathrm{tr}\left[\mathbf{X}^{(1)}\mathbf{X}^{(\ell),T}\mathbf{L}^{(\ell)}\mathbf{e}_j\mathbf{e}_i^T\right] \\
&= \mathrm{vec}\left[\mathbf{X}^{(\ell)}\right]^T\left(\mathbf{L}^{(\ell)}\mathbf{e}_j\mathbf{e}_i^T\otimes\mathbf{I}_{d_{\setminus\ell}}\right)\mathrm{vec}\left[\mathbf{X}^{(\ell)}\right] \\
\|\frac{2}{d_{\setminus\ell}}\left(\mathbf{L}^{(\ell)}\mathbf{e}_j\mathbf{e}_i^T\otimes\mathbf{I}_{d_{\setminus\ell}}\right)\|_{op} &= \frac{2}{d_{\setminus\ell}}\|\left(\mathbf{L}^{(\ell)}\mathbf{e}_j\mathbf{e}_i^T\right)\|_{op} \\
&= \frac{2}{d_{\setminus\ell}}\|\mathbf{L}^{(\ell)}\|_{op} \\
\|\frac{2}{d_{\setminus\ell}}\left(\mathbf{L}^{(\ell)}\mathbf{e}_j\mathbf{e}_i^T\otimes\mathbf{I}_{d_{\setminus\ell}}\right)\|_F &= \frac{2}{\sqrt{d_{\setminus\ell}}}\|\left(\mathbf{L}^{(\ell)}\mathbf{e}_j\mathbf{e}_i^T\right)\|_F \\
&= \frac{2}{\sqrt{d_{\setminus\ell}}}\|\mathbf{L}^{(\ell)}\|_F
\end{aligned}$$

The second term is a sum over the other axes ℓ', with summands being $\sum_{\mathbf{c}}\mathbf{X}_{\mathbf{c}}^{(\ell;\ell')}\mathbf{L}^{(\ell'),T}\mathbf{X}_{\mathbf{c}}^{(\ell;\ell'),T}$.

$$\begin{aligned}
\sum_{\mathbf{c}}\mathbf{X}_{\mathbf{c}}^{(\ell;\ell')}\mathbf{L}^{(\ell'),T}\mathbf{X}_{\mathbf{c}}^{(\ell;\ell'),T} &= \mathbf{X}^{(\ell)}\left(\mathbf{L}^{(\ell'),T}\otimes\mathbf{I}_{d_{\setminus\ell\setminus\ell'}}\right)\mathbf{X}^{(\ell),T} \\
\mathbf{e}_i^T\mathbf{X}^{(\ell)}\left(\mathbf{L}^{(\ell'),T}\otimes\mathbf{I}_{d_{\setminus\ell\setminus\ell'}}\right)\mathbf{X}^{(\ell),T}\mathbf{e}_j &= \mathrm{tr}\left[\mathbf{X}^{(\ell)}\left(\mathbf{L}^{(\ell'),T}\otimes\mathbf{I}_{d_{\setminus\ell\setminus\ell'}}\right)\mathbf{X}^{(\ell),T}\mathbf{e}_j\mathbf{e}_i^T\right] \\
&= \mathrm{vec}\left[\mathbf{X}^{(\ell)}\right]^T\left(\mathbf{L}^{(\ell')}\otimes\mathbf{I}_{d_{\setminus\ell\setminus\ell'}}\otimes\mathbf{e}_j\mathbf{e}_i^T\right)\mathrm{vec}\left[\mathbf{X}^{(\ell)}\right] \\
\|\frac{2}{\sqrt{d_{\setminus\ell}d_{\setminus\ell'}}}\left(\mathbf{L}^{(\ell')}\otimes\mathbf{I}_{d_{\setminus\ell\setminus\ell'}}\otimes\mathbf{e}_j\mathbf{e}_i^T\right)\|_{op} &= \frac{2}{\sqrt{d_{\setminus\ell}d_{\setminus\ell'}}}\|\mathbf{L}^{(\ell')}\|_{op} \\
\|\frac{2}{\sqrt{d_{\setminus\ell}d_{\setminus\ell'}}}\left(\mathbf{L}^{(\ell')}\otimes\mathbf{I}_{d_{\setminus\ell\setminus\ell'}}\otimes\mathbf{e}_j\mathbf{e}_i^T\right)\|_F &= \frac{2d_{\setminus\ell}}{d_{\ell'}\sqrt{d_{\setminus\ell}d_{\setminus\ell'}}}\|\left(\mathbf{L}^{(\ell')}\otimes\mathbf{I}_{d_{\setminus\ell\setminus\ell'}}\right)\|_F \\
&= \frac{2}{\sqrt{d_\ell d_{\ell'}}}\|\mathbf{L}^{(\ell')}\|_F
\end{aligned}$$

Let $\mathbf{A}^{(i,j,\ell)}$ be the matrix representing the quadratic form for the gradient term $\mathbf{L}_{ij}^{(\ell)}$. Let R_{op}, R_F be the maximal operator and Frobenius norms of any factor. Then, by the triangle inequality, we have:

$$\|\mathbf{A}^{(i,j,\ell)}\|_{op} \leq \sum_{\ell'} \frac{2R_{op}}{\sqrt{d_{\setminus\ell}d_{\setminus\ell'}}}$$
$$\|\mathbf{A}^{(i,j,\ell)}\|_F \leq 2R_F\left(\frac{1}{\sqrt{d_{\setminus\ell}}} + \sum_{\ell'\neq\ell}\frac{1}{\sqrt{d_\ell d_{\ell'}}}\right)$$

□

We'll now move on to deriving the gradient of the log determinant term, which again requires some bespoke notation to express succinctly. Let $idx_\ell(j)$ be such that $\mathbf{L}^{(\ell)}_{idx_\ell(j)}$ that affects $\mathbf{L}_{jj}$. In other words, $\mathbf{L}_{jj} = \sum_\ell \mathbf{L}^{(\ell)}_{idx_\ell(j),idx_\ell(j)}$.

Gradient 2. *The gradient of* $-2\log\det\left[\operatorname{diag}\mathbf{L}\right]$ *w.r.t.* $\mathbf{L}^{(b)}_{aa}$ *is:*

$$-2\sum_{j|idx_b(j)=a}\mathbf{L}^{-1}_{jj}$$

Proof.

$$\begin{aligned}-\log\det\left[\mathbf{L}\mathbf{L}^T\right] &= -2\log\det\left[\operatorname{diag}\mathbf{L}\right]\\ &= -2\sum_j \log\mathbf{L}_{jj}\\ &= -2\sum_j \log\sum_\ell \frac{1}{\sqrt{d_{\setminus\ell}}}\mathbf{L}^{(\ell)}_{idx_\ell(j),idx_\ell(j)}\end{aligned}$$

The first derivative with respect to $\mathbf{L}^{(b)}_{aa}$ is:

$$\begin{aligned}&-2\sum_{j|idx_b(j)=a}\frac{1}{\sum_\ell \frac{1}{\sqrt{d_{\setminus\ell}}}\mathbf{L}^{(\ell)}_{idx_\ell(j),idx_\ell(j)}}\\ &= -2\sum_{j|idx_b(j)=a}\mathbf{L}^{-1}_{jj}\end{aligned}$$

□

An easy way to compute Gradient 2 is to construct the tensor $\mathcal{T}_{\mathbf{i}} = \frac{-1}{\sum_j \mathbf{L}^{(\ell)}_{\mathbf{i}_\ell\mathbf{i}_\ell}}$. The derivative with respect to $\operatorname{diag}\mathbf{L}^{(\ell)}$ is $-2\cdot\mathbf{1}^T\operatorname{mat}_\ell\left[\mathcal{T}\right]$.

From these results, it is easy to see that the derivative with respect to all K factor matrices can be computed in $O\left(\sum_\ell\left(d_\ell^3 + K\sum_{\ell'\neq\ell} d_{\ell'}^2\prod_{\ell''\neq\ell'} d_{\ell''}\right) + \prod_\ell d_\ell\right)$, which if all axes are the same size d can be simplified to $O(Kd^{K+1})$ for $K > 1$. This complexity is in line with other tensor-variate Kronecker-structured methods, such as TeraLasso for precision matrices.

When $K = 2$, the gradient can be simplified to:

Gradient 3. *The gradient for matrix-variate data, with respect to* $\mathbf{L}^{(1)}$*, is:*

$$\frac{2}{\sqrt{d_1}}\mathbf{X}\mathbf{X}^T\mathbf{L}^{(1)} - 2\operatorname{diag}\left[\mathbf{1}^T\operatorname{mat}_1\left[\mathcal{T}\right]\right] + \frac{2}{\sqrt{d_2}}\mathbf{X}\mathbf{L}^{(2),T}\mathbf{X}^T$$

With respect to $\mathbf{L}^{(2)}$*, it is:*

$$\frac{2}{\sqrt{d_2}}\mathbf{X}^T\mathbf{X}\mathbf{L}^{(2)} - 2\operatorname{diag}\left[\mathbf{1}^T\operatorname{mat}_2\left[\mathcal{T}\right]\right] + \frac{2}{\sqrt{d_1}}\mathbf{X}^T\mathbf{L}^{(1),T}\mathbf{X}$$

D.3 Proofs

In this section we'll prove Theorems 6-9 from the main paper. The first two concern solution existence and the second two concern statistical recoverability, so we have split this section into two subsections for organization.

Solution existence (lopsidedness)

Our proof of Theorem 7 revolves around Grassmann's formula.

Definitions (Grassmann's Formula)**.** *For any two linear spaces $\mathbb{B}, \mathbb{C}$ in an ambient space $\mathbb{A}$, we have the following:*

$$\dim[\mathbb{B} \cap \mathbb{C}] \geq \max\left(0, \dim[\mathbb{B}] + \dim[\mathbb{C}] - \dim[\mathbb{A}]\right) \qquad \text{(Grassmann's Formula)}$$

If one of $\mathbb{B}, \mathbb{C}$ are in general position, then with probability 1 the inequality becomes an equality.

Theorem 7 (Lopsidedness Theorem)**.** *Consider a dataset $\mathcal{X}_{d_1 \times \cdots \times d_K} \neq 0$. Under Assumption 6, if $\prod_\ell d_\ell + K \geq 1 + \sum_\ell \frac{d_\ell^2 - d_\ell}{2}$, then with probability 1 a solution exists to Equation 5.3. In particular, consider a matrix-variate dataset $\mathbf{X}_{d_1 \times d_2}$, for $d_2 = d_1 + \delta \geq d_1$. Then, a solution exists with probability 1 if $\delta \leq \frac{1+\sqrt{9+8d_1}}{2}$.*

Proof. Let us consider the following spaces:

- $\mathbb{L}^\oplus$, with dimension $\sum_\ell \frac{d_\ell^2 - d_\ell}{2} - K + 1$
- $\mathbb{X}$, the space of matrices $\mathbf{M}$ such that $\mathbf{M}\text{vec}[\mathcal{X}] = \mathbf{0}$.
- $\mathbb{A}$, some suitable ambient space containing $\mathbb{L}^\oplus$, such as the space of $d_\forall \times d_\forall$ lower triangular matrices.
- $\mathbb{X} \cap \mathbb{A}$, which has dimension $\dim[\mathbb{A}] - d_\forall$ as each of the $d_\forall$ rows of a matrix $\mathbf{M}$ must be orthogonal to $\text{vec}[\mathcal{X}]$ for $\mathbf{M} \in \mathbb{X}$.

By the definition of $\mathbb{X}$, any direction $\mathbf{\Delta} \in \mathbb{L}^\oplus \cap \mathbb{X}$ leaves the quadratic term of Equation 5.3 unchanged: $\|\mathbf{L}\text{vec}[\mathcal{X}]\|_2^2 = \|(\mathbf{L} + \mathbf{\Delta})\,\text{vec}[\mathcal{X}]\|_2^2$. With access to such a $\mathbf{\Delta}$, it is possible to independently alter the log determinant term, potentially driving it to $-\infty$. Without access, this is not possible: the quadratic term grows much faster than the logarithmic term, and hence any attempt to drive the objective to $-\infty$ will eventually be counterbalanced.

If we can establish that $\dim\left[\mathbb{X} \cap \mathbb{L}^\oplus\right] = 0$, then we know that there is no such $\mathbf{\Delta} \in \mathbb{X} \cap \mathbb{L}^\oplus$. Under Assumption 6, $\mathbb{X}$ is in general position. Thus, our objective reduces to proving that $\dim[\mathbb{X}] + \dim\left[\mathbb{L}^\oplus\right] - \dim[\mathbb{A}] \leq 0$.

We can immediately do this for the tensor-variate case:

$$\begin{aligned}
& & 0 &\geq \dim[\mathbb{X}] + \dim\left[\mathbb{L}^\oplus\right] - \dim[\mathbb{A}] \\
& & &= \sum_\ell \frac{d_\ell^2 - d_\ell}{2} - K + 1 - d_\forall \\
\iff & & d_\forall + K &\geq 1 + \sum_\ell \frac{d_\ell^2 - d_\ell}{2}
\end{aligned}$$

Intuitively, when $K \geq 3$, this should be easy to satisfy - the left hand is at least a cubic in $\{d_\ell\}$ whereas the right hand is a quadratic. This is sensible; each axis ℓ has effectively $\frac{d_\forall}{d_\ell} \geq d_{\min}^{K-1}$ samples, and hence a lot of information to work with when $K \geq 3$.

For matrix-variate data, the condition is more restrictive. Suppose we had a $d_1 \times d_1 + \delta$ dataset:

$$\begin{aligned}
& & 0 &\geq \sum_\ell \frac{d_\ell^2 - d_\ell}{2} - K + 1 - d_\forall \\
\iff & & 0 &\geq \sum_\ell d_\ell^2 - d_\ell - 2K + 2 - 2d_\forall \\
& & &= d_1^2 - d_1 + d_1^2 + 2d_1\delta + \delta^2 - d_1 - \delta - 2 - 2d_1^2 - 2d_1\delta && \text{(expand)} \\
& & &= \delta^2 - \delta - 2 - 2d_1 && \text{(simplify)} \\
\iff & & \delta &\leq \frac{1 + \sqrt{9 + 8d_1}}{2} && \text{(solve quadratic)}
\end{aligned}$$

This completes the proof. The theorem as given in the main paper mentions the stronger condition of 'strong convexity' rather than just solution existence; we defer the proof of strong convexity, which follows from a straightforward extension of our argument here, to Lemma 9. □

Corollary. *A sufficient condition for solution existence with probability 1 is that, for the largest axis* $d_{\max}$*, we have* $d_{\setminus\max} \geq \frac{K}{2} d_{\max}$*.*

Proof. First, note that $\sum_\ell d_\ell^2 \geq 1 - K + \sum_\ell \frac{d_\ell^2 - d_\ell}{2}$, so a sufficient condition is for $2\prod_\ell d_\ell \geq \sum_\ell d_\ell^2$. In other words, for the largest axis $d_{\max}$ a sufficient condition is that $2d_{\setminus\max} \geq d_{\max} + \sum_{\ell \neq \max} \frac{d_\ell^2}{d_{\max}} \geq K d_{\max}$. □

Theorem 7 guarantees the existence of solutions under certain conditions, but it does not demand non-existence when such conditions are not satisfied; this would only happen if for some $\boldsymbol{\Delta} \in \mathbb{L}^\oplus \cap \mathbb{X}$ we have that $\text{diag}\boldsymbol{\Delta} \geq 0$. Otherwise, we can only travel along $\boldsymbol{\Delta}$ a finite distance before some diagonal of $\mathbf{L} + \eta\boldsymbol{\Delta}$ snags on '0'. Since we require the diagonals to be strictly positive, we are not able to progress, saving us from non-existence. Using Cover's Theorem[1], we can establish the precise probability that a solution exists by expressing the problem as the probability that $\dim\left[\mathbb{L}^\oplus \cap \mathbb{X} \cap \mathbb{R}_+^d\right] > 0$.

Definitions (Cover's Theorem (Cover, 1965))**.** *A* D*-dimensional subspace of* $\mathbb{R}^N$ *in general position intersects* $C(N, D) = 2\sum_{i=0}^{D-1} \binom{N-1}{i}$ *orthants.*

Theorem 8 (Excess Lopsidedness Theorem)**.** *Let* $D = \sum_\ell \frac{d_\ell^2 - d_\ell}{2} - K + 1 - d_\forall$*, and suppose* $D > 0$*; in other words,* D *represents the 'excess lopsidedness' of the data. For* $(d_1, d_1 + \delta)$*-shaped (matrix-variate) data, this is equivalent to* $D = \frac{\delta^2 - \delta}{2} - (d_1 + 1)$*.*

The probability that a solution exists to Equation 5.3 is given by $1 - \frac{1}{2^{d_\forall - 1}} \sum_{i=0}^{\min(D, d_\forall))-1} \binom{d_\forall - 1}{i}$*.*

Proof. As mentioned above, the only way for existence of a solution to fail is if there exists a vector $\boldsymbol{\Delta} \in \mathbb{X} \cap \mathbb{L}^\oplus$ with all-positive diagonals. Theorem 7 tells us that $\dim\left[\mathbb{X} \cap \mathbb{L}^\oplus\right] = 0$ when the dataset is not too lopsided, trivially satisfying this requirement. However, if the dataset *is* too lopsided, then there do exist $\boldsymbol{\Delta} \in \mathbb{X} \cap \mathbb{L}^\oplus$; we need to find the probability that any such $\boldsymbol{\Delta}$ contains a negative element on the diagonal, which will depend on $\dim\left[\mathbb{X} \cap \mathbb{L}^\oplus\right]$.

First, let's understand how the size of $D = \dim\left[\mathbb{X} \cap \mathbb{L}^\oplus\right] > 0$ increases as the data becomes more lopsided. From Theorem 7, recall:

$$D = \sum_\ell \frac{d_\ell^2 - d_\ell}{2} - K + 1 - d_\forall \qquad \text{(almost surely)}$$

For matrix-variate data, we can simplify this to

$$\begin{aligned}
& D = \sum_\ell \frac{d_\ell^2 - d_\ell}{2} - K + 1 - d_\forall \\
\iff \quad & 2D = \sum_\ell d_\ell^2 - d_\ell - 2K + 2 - 2d_\forall \\
& = d_1^2 - d_1 + d_1^2 + 2d_1\delta + \delta^2 - d_1 - \delta - 2 - 2d_1^2 - 2d_1\delta && \text{(expand)} \\
& = \delta^2 - \delta - 2 - d_1 && \text{(simplify)} \\
\iff \quad & D = \frac{\delta^2 - \delta}{2} - (d_1 + 1)
\end{aligned}$$

As the lopsidedness increases, the dimensionality of $\mathbb{X} \cap \mathbb{L}^\oplus$ increases quadratically. The dimensionality of the diagonals is $\min(D, d_\forall)$, as there are $d_\forall$ diagonals. Using Cover's Theorem, we can observe that the probability that $\mathbb{X} \cap \mathbb{L}^\oplus$ intersects the positive orthant is given by $\frac{C(d_\forall, \min(D, d_\forall))}{2^{d_\forall}} = \frac{1}{2^{d_\forall - 1}} \sum_{i=0}^{\min(D, d_\forall))-1} \binom{d_\forall - 1}{i}$. This completes the proof.

□

[1] Cover's Theorem has many formulations. In particular, we refer to Theorem 2' from (Cover, 1965).

Statistical recovery rates

We'll now prove the statistical recovery rates of Theorem 6, which will require a few definitions and lemmas.

Definitions.

$\|\mathbf{L}\|_{\oplus} = \sum_{\ell} \|\mathbf{L}^{(\ell)}\|$ *for any norm.*

We can abbreviate our objective (Equation 5.4) as $f(\mathbf{L}; \mathcal{X}) = g(\mathbf{L}; \mathcal{X}) + \rho \|\mathbf{L}\|_{F,\oplus}^2 + \lambda \|\mathbf{L}\|_{od,\oplus}$*, assuming each axis gets the same level of regularization.*

$h(\mathbf{L}; \mathcal{X}) = g(\mathbf{L}; \mathcal{X}) + \rho \|\mathbf{L}\|_{F,\oplus}^2$

$\mathbf{L}^* = \underset{\mathbf{L}}{\operatorname{argmin}} \ \mathbb{E}_{\mathcal{X} \sim \mathcal{D}^*} \left[f\left(\mathbf{L}; \mathcal{X}\right) \right]$ *is the oracle solution.*

$\mathbf{L}_{\rho}^* = \underset{\mathbf{L}}{\operatorname{argmin}} \ \mathbb{E}_{\mathcal{X} \sim \mathcal{D}^*} \left[f\left(\mathbf{L}; \mathcal{X}\right) \right] + \rho \|\mathbf{L}\|_{F,\oplus}^2$ *is the regularized oracle solution.*

Definitions (Second-Order Characterization of Strong Convexity)**.** *For all* $\mathbf{L} \in \mathbb{L}_{++}^{\oplus}$ *and* $\mathbf{\Delta} \in \mathbb{L}^{\oplus}$*, with Euclidean geodesics* $\gamma_{\mathbf{L},\mathbf{\Delta}}(t) = \mathbf{L} + t\mathbf{\Delta}$*, a function* f *satisfies* μ*-strong convexity if:*

$$\frac{d^2}{dt^2} f(\gamma(t)) \bigg|_{t=0} \geq \mu \|\mathbf{\Delta}\|_F^2$$

Lemma 9 (Strong Convexity)**.** *Suppose our data is not too lopsided (Theorem 7). Then,* g *and* $\mathbb{E}\left[g\right]$ *are* μ*-strongly convex for some* $\mu > 0$*.*

Proof. Recall that $g(\mathbf{L}) = \operatorname{tr}\left[\mathbf{L}\mathbf{L}^T\mathbf{S}\right] - \log\det \mathbf{L}\mathbf{L}^T$. We'll show that $\operatorname{tr}\left[\mathbf{L}\mathbf{L}^T\mathbf{S}\right]$ is strongly convex, which will imply g itself is strongly convex. This should be intuitively true, as this term is quadratic in $\mathbf{L}$ and Theorem 7 rules out the possibility of any 'flat' directions. We just need to formalize this.

$$\begin{aligned}
\frac{d}{dt} \operatorname{tr}\left[\gamma(t)\gamma(t)^T\mathbf{S}\right] \bigg|_{t=0} &= 2\operatorname{tr}\left[\gamma''(0)\gamma(0)^T\mathbf{S}\right] + 2\operatorname{tr}\left[\gamma'(0)\gamma'(0)^T\mathbf{S}\right] \\
&= 2\operatorname{tr}\left[\Delta\Delta^T\mathbf{S}\right] \\
&= 2\operatorname{vec}\left[\mathbf{\Delta}\right]^T \left(\mathbf{S} \otimes \mathbf{I}_{d_\forall \times d_\forall}\right) \operatorname{vec}\left[\mathbf{\Delta}\right]
\end{aligned}$$

Typically, this would not be strongly convex - $\mathbf{S}$ is rank-1 ($\mathbf{S} \otimes \mathbf{I}_{d_\forall \times d_\forall}$ being rank-$d_\forall$), and hence contains many zero eigenvalues. However, $\mathbf{\Delta} \in \mathbb{L}^{\oplus}$, and hence has special structure. Let $\mathbf{P}_{\mathbb{L}^{\oplus}}$ be the matrix that projects an arbitrary vector onto $\mathbb{L}^{\oplus}$; $\mathbf{P}_{\mathbb{L}^{\oplus}} \operatorname{vec}\left[\mathbf{\Delta}\right] = \operatorname{vec}\left[\mathbf{\Delta}\right]$. Our second derivative then becomes equivalent to $2\operatorname{vec}\left[\mathbf{\Delta}\right]^T \mathbf{P}_{\mathbb{L}^{\oplus}}^T \left(\mathbf{S} \otimes \mathbf{I}_{d_\forall \times d_\forall}\right) \mathbf{P}_{\mathbb{L}^{\oplus}} \operatorname{vec}\left[\mathbf{\Delta}\right]$.

We know that $\mathbf{P}_{\mathbb{L}^{\oplus}}^T \left(\mathbf{S} \otimes \mathbf{I}_{d_\forall \times d_\forall}\right) \mathbf{P}_{\mathbb{L}^{\oplus}}$ has no zero eigenvalues, as otherwise that would mean there was a matrix $\mathbf{\Delta} \in \mathbb{L}^{\oplus} \cap \mathbb{X}$ - a direct contradiction to Theorem 7. Letting $\frac{\mu}{2}$ be the minimal eigenvalue of this matrix, we must have that:

$$\begin{aligned}
\frac{d}{dt} \operatorname{tr}\left[\gamma(t)\gamma(t)^T\mathbf{S}\right] \bigg|_{t=0} &= 2\operatorname{vec}\left[\mathbf{\Delta}\right]^T \left(\mathbf{S} \otimes \mathbf{I}_{d_\forall \times d_\forall}\right) \operatorname{vec}\left[\mathbf{\Delta}\right] \\
&\geq \mu \|\mathbf{\Delta}\|_F^2
\end{aligned}$$

This proves strong convexity of g. $\mathbb{E}$ preserves strong convexity, completing the theorem.

□

In theory, μ is simple to calculate. However, in practice $\mathbf{P}_{\mathbb{L}^{\oplus}}^T \left(\mathbf{S} \otimes \mathbf{I}_{d_\forall \times d_\forall}\right) \mathbf{P}_{\mathbb{L}^{\oplus}}$ has $O(d_{\max}^{4K})$ matrix entries - exact computation quickly becomes intractible. In contrast, the maximal eigenvalue (which governs the smoothness of the term) is easy to bound, being at most $\|\mathcal{X}\|_F^2$, the only nonzero eigenvector of $\mathbf{S}$.

We now move on to bounding the difference between our regularized and unregularized oracles. This difference expresses the bias introduced by our regularizer.

Definitions (Score Function Property)**.**

$\mathbf{L}^*$ *satisfies the property that* $\nabla \mathbb{E}\left[g\right]\left(\mathbf{L}^*; \mathcal{X}\right) = 0$

$\mathbf{L}_\rho^*$ *satisfies the property that* $\nabla_\ell \mathbb{E}\left[g\right]\left(\mathbf{L}_\rho^*; \mathcal{X}\right) + 2\rho_\ell \mathbf{L}_\rho^{*,(\ell)} = 0$

Definitions (First-Order Characterization of Strong Convexity)**.**

f is μ-strongly convex if and only if for all $\mathbf{A}, \mathbf{B}$:

$$f(\mathbf{B}) \geq f(\mathbf{A}) + \langle \nabla f(\mathbf{A}), \mathbf{B} - \mathbf{A} \rangle + \frac{\mu}{2}\|\mathbf{B} - \mathbf{A}\|_F^2$$

If f is μ-strongly convex, then for all $\mathbf{A}, \mathbf{B}$:

$$\mu\|\mathbf{A} - \mathbf{B}\|_F \leq \|\nabla f\left(\mathbf{A}\right) - \nabla f\left(\mathbf{B}\right)\|_F$$

Lemma 10 (Bounding L2 Regularization Bias)**.**

$$\|\mathbf{L}_\rho^{*,(\ell)} - \mathbf{L}^{*,(\ell)}\|_F \leq \frac{2\rho_\ell}{\mu}\|\mathbf{L}_\rho^{*,(\ell)}\|_{F,}$$

$$\|\mathbf{L}_\rho^* - \mathbf{L}^*\|_{F,\oplus} \leq \frac{2K\rho_{\max}}{\mu} R_F$$

Proof. By the score function property, $\nabla \mathbb{E}\left[g\right]\left(\mathbf{L}^{*,(\ell)}; \mathcal{X}\right) - \nabla \mathbb{E}\left[g\right]\left(\mathbf{L}_\rho^{*,(\ell)}; \mathcal{X}\right) = -2\rho_\ell \mathbf{L}_\rho^{*,(\ell)}$

$$\begin{aligned}\mu\|\mathbf{L}_\rho^{*,(\ell)} - \mathbf{L}^{*,(\ell)}\|_F &\leq \|\nabla \mathbb{E}\left[g\right]\left(\mathbf{L}^{*,(\ell)}; \mathcal{X}\right) - \nabla \mathbb{E}\left[g\right]\left(\mathbf{L}_\rho^{*,(\ell)}; \mathcal{X}\right)\|_F \\ &\leq 2\rho_\ell \|\mathbf{L}_\rho^{*,(\ell)}\|_F\end{aligned}$$

To achieve a non-factor-wise-bound, note:

$$\begin{aligned}\|\mathbf{L}_\rho^* - \mathbf{L}^*\|_{F,\oplus} &= \sum_\ell \|\mathbf{L}_\rho^{*,(\ell)} - \mathbf{L}^{*,(\ell)}\|_F \\ &\leq \sum_\ell \frac{2\rho_\ell}{\mu}\|\mathbf{L}_\rho^{*,(\ell)}\|_F \\ &\leq \frac{2K\rho_{\max}}{\mu} R_F\end{aligned}$$

□

$\|\mathbf{L}_\rho^{*,(\ell)} - \mathbf{L}^{*,(\ell)}\|_F$ represents the distance between the oracle parameters without regularization and the oracle parameters with regularization; it's the bias cost we pay for using regularization to reduce variance. Lemma 10 tells us that reducing ρ reduces the maximum bias, as expected. To bound the error, we need to know $\|\hat{\mathbf{L}}^{(\ell)} - \mathbf{L}_\rho^{*,(\ell)}\|_F$, and to know this we need another lemma.

Definitions (Hanson-Wright Inequality)**.** *Let $\mathbf{x}$ be a subgaussian random vector with subgaussian norm σ and covariance matrix $\mathbf{\Sigma}$. Let $B = \sigma\sqrt{\|\mathbf{\Sigma}\|_{op}}$ and let $\mathbf{A}$ be some matrix. Then:*

$$\mathbb{P}\left(\left|\mathbf{x}^T\mathbf{A}\mathbf{x} - \mathbb{E}\left[\mathbf{x}^T\mathbf{A}\mathbf{x}\right]\right| \geq t\right) \leq 2\exp\left(-c\min\left(\frac{t^2}{B^4\|\mathbf{A}\|_F^2}, \frac{t}{B^2\|\mathbf{A}\|_{op}}\right)\right)$$

In the above, c is a universal constant.

Definitions (Generalized Hanson-Wright Inequality (Sambale, 2023))**.** *Let $\mathbf{x}$ be an α-sub-exponential random vector ($\alpha \in (0, 2]$) with ψ_α-Orlicz norm σ and covariance matrix $\mathbf{\Sigma}$. Let $B = \sigma\sqrt{\|\mathbf{\Sigma}\|_{op}}$ and let $\mathbf{A}$ be some matrix. Then:*

$$\mathbb{P}\left(\left|\mathbf{x}^T\mathbf{A}\mathbf{x} - \mathbb{E}\left[\mathbf{x}^T\mathbf{A}\mathbf{x}\right]\right| \geq t\right) \leq 2\exp\left(-c_\alpha\min\left(\frac{t^2}{B^4\|\mathbf{A}\|_F^2}, \left[\frac{t}{B^2\|\mathbf{A}\|_{op}}\right]^{\frac{\alpha}{2}}\right)\right)$$

In the above, c_α depends only on α.

The Hanson-Wright inequality allows us to bound with high probability the deviation from the mean of a quadratic form of α-sub-exponential random vectors. We will express our gradient in exactly such a form, allowing us to bound its norm at the oracle, which we will then use to bound our distance from said oracle.

Lemma 11 (Bounding Oracle Gradients)**.** *Suppose $\mathcal{X}$ is an α-sub-exponential random vector (with parameter B), and $\mathbf{A}^{(\ell)}$ be the matrix corresponding to the quadratic form ∇f).*

$$\mathbb{P}\left(\|\nabla h(\mathbf{L}_\rho^*;\mathcal{X})^{(\ell)}\|_\infty > t\right) \leq 2d_\ell^2 \exp\left(-c_\alpha \min\left(\frac{t^2}{B^4\|\mathbf{A}^{(\ell)}\|_F^2}, \left[\frac{t}{B^2\|\mathbf{A}^{(\ell)}\|_{op}}\right]^{\frac{\alpha}{2}}\right)\right)$$

Proof. The gradient can be re-expressed as follows:

$$\begin{aligned}
\nabla h(\mathbf{L}_\rho^*;\mathcal{X}) &= \nabla f(\mathbf{L}_\rho^*;\mathcal{X}) + 2\rho\mathbf{L}_\rho^* && \text{(definition)}\\
&= \nabla f(\mathbf{L}_\rho^*;\mathcal{X}) + 2\rho\mathbf{L}_\rho^* - \underbrace{\mathbb{E}_{\mathcal{X}\sim\mathcal{D}^*}\left[\nabla h(\mathbf{L}_\rho^*;\mathcal{X})\right]}_{0} && \text{(score function property)}\\
&= \nabla f(\mathbf{L}_\rho^*;\mathcal{X}) - \mathbb{E}_{\mathcal{X}\sim\mathcal{D}^*}\left[\nabla f(\mathbf{L}_\rho^*;\mathcal{X})\right]
\end{aligned}$$

Recall from Lemma 8 that ∇f is quadratic in $\mathcal{X}$:

$$\nabla f(\mathbf{L}_\rho^*;\mathcal{X})_{i,j,\ell} = \text{vec}\left[\mathcal{X}\right]^T \mathbf{A}^{(i,j,\ell)}\text{vec}\left[\mathcal{X}\right]$$

The generalized Hanson-Wright inequality gives an expression for a probability bound for a single entry of $\nabla h(\mathbf{L}_\rho^*;\mathcal{X})^{(\ell)}$; there are d_ℓ^2 entries in total[2], so we apply a union bound:

$$\mathbb{P}\left(\|\nabla h(\mathbf{L}_\rho^*;\mathcal{X})^{(\ell)}\|_\infty > t\right) \leq 2d_\ell^2 \exp\left(-c_\alpha \min\left(\frac{t^2}{B^4\|\mathbf{A}^{(\ell)}\|_F^2}, \left[\frac{t}{B^2\|\mathbf{A}^{(\ell)}\|_{op}}\right]^{\frac{\alpha}{2}}\right)\right)$$

Above, we use the shorthand that, for each norm, $\|\mathbf{A}^{(\ell)}\| = \max_{ij}\|\mathbf{A}^{(i,j,\ell)}\|$.
This completes the proof. □

Lemma 12 (Choice of Regularizer)**.** *Suppose $\mathcal{X}$ is α-sub-exponential as in Lemma 11, and also let $r^{(\ell)} = \frac{\|\mathbf{A}^{(\ell)}\|_{op}}{\|\mathbf{A}^{(\ell)}\|_F}$. Let $\beta_\ell = 1$ if $2^{1-p\log d_\ell} \geq \frac{2}{d_\ell^2}\exp\left(c_\alpha r_\ell^{\frac{2\alpha}{\alpha-4}}\right)$, and otherwise $\beta_\ell = \frac{4}{\alpha}$. Suppose the following holds:*

$$\lambda_\ell \asymp \sqrt{\frac{(p\log d_\ell)^{\beta_\ell}}{d_{\setminus\ell}}} + \sum_{\ell'\neq\ell}\sqrt{\frac{(p\log d_\ell)^{\beta_\ell}}{d_\ell d_{\ell'}}}$$

Then, with probability $2^{1-p\log d_\ell}$, we have that $\|\nabla h(\mathbf{L}_\rho^*;\mathcal{X})^{(\ell)}\|_\infty \leq \frac{\lambda_\ell}{2}$

Proof. If we want to know the probability of $\|\nabla h(\mathbf{L}_\rho^*;\mathcal{X})^{(\ell)}\|_\infty < t$ by $2^{1-p\log d_\ell}$, we can see that:

[2]We could account for symmetry to decrease this and achieve a somewhat tighter bound, but the improvement is asymptotically irrelevant.

$$2^{1-p\log d_\ell} \geq 2d_\ell^2 \exp\left(-c_\alpha \min\left(\frac{t^2}{B^4\|\mathbf{A}^{(\ell)}\|_F^2}, \left[\frac{t}{B^2\|\mathbf{A}^{(\ell)}\|_{op}}\right]^{\frac{\alpha}{2}}\right)\right)$$

$$\iff \quad 2^{p\log d_\ell} d_\ell^2 \leq \exp\left(c_\alpha \min\left(\frac{t^2}{B^4\|\mathbf{A}^{(\ell)}\|_F^2}, \left[\frac{t}{B^2\|\mathbf{A}^{(\ell)}\|_{op}}\right]^{\frac{\alpha}{2}}\right)\right)$$

$$\iff \quad \frac{(p\log 2 + 2)\log d_\ell}{c_\alpha} \leq \min\left(\frac{t^2}{B^4\|\mathbf{A}^{(\ell)}\|_F^2}, \left[\frac{t}{B^2\|\mathbf{A}^{(\ell)}\|_{op}}\right]^{\frac{\alpha}{2}}\right)$$

Splitting over the minimum, we must have both of the following being satisfied:

$$t \geq B^2\|\mathbf{A}^{(\ell)}\|_F \left(\frac{(p\log 2 + 2)\log d_\ell}{c_\alpha}\right)^{\frac{1}{2}}$$

$$t \geq B^2\|\mathbf{A}^{(\ell)}\|_{op} \left(\frac{(p\log 2 + 2)\log d_\ell}{c_\alpha}\right)^{\frac{2}{\alpha}}$$

In general, the Frobenius-based lower-bound governs the central probabilities and the operator-norm-based lower-bound governs the tail. We'd like to find the cross-over point of the two. Let $r^{(\ell)} = \frac{\|\mathbf{A}^{(\ell)}\|_{op}}{\|\mathbf{A}^{(\ell)}\|_F}$. The Frobenius norm is the Euclidean norm of the singular values of $\mathbf{A}$, whereas the operator norm is just the largest singular value - thus, $r^{(\ell)} \in [\frac{1}{\sqrt{d}}, 1]$, and the closer to 1 it is the more one singular value dominates at the expense of the rest. The crossover point can then be expressed as:

$$r^{(\ell)} = \left(\frac{(p\log 2 + 2)\log d_\ell}{c_\alpha}\right)^{\frac{\alpha-4}{2\alpha}}$$

$$\iff \quad p\log d_\ell = \frac{c_\alpha r_\ell^{\frac{2\alpha}{\alpha-4}} - 2\log d_\ell}{\log 2}$$

For p larger than this, the operator-norm-based bound dominates. Recall that we originally bounded our probability by $2^{1-p\log d_\ell}$. Note that, at the crossover point:

$$2^{1-p\log d_\ell} = \frac{2}{d_\ell^2}\exp\left(c_\alpha r_\ell^{\frac{2\alpha}{\alpha-4}}\right)$$

If r_ℓ is large, then the crossover probability is larger (and then we get into the less favorable tail regime sooner). Likewise, for $\alpha \in (0, 2]$, $\frac{\alpha-4}{2\alpha}$ is monotonic increasing in α, which since $r \leq 1$ implies small α increase the crossover probability. This makes sense: large $r^{(\ell)}$ correspond to less-well-conditioned matrices $\mathbf{A}$, and small α correspond to heavier-tailed source distributions.

We will now consider what happens in the two regimes. Recall that, from Lemma 8, we have:

$$\|\mathbf{A}^{(\ell)}\|_{op} \leq \sum_{\ell'} \frac{2R_{op}}{\sqrt{d_{\setminus\ell} d_{\setminus\ell'}}}$$

$$\|\mathbf{A}^{(\ell)}\|_F \leq 2R_F\left(\frac{1}{\sqrt{d_{\setminus\ell}}} + \sum_{\ell'\neq\ell}\frac{1}{\sqrt{d_\ell d_{\ell'}}}\right)$$

If the following conditions are satisfied, then our probability bound on $\|\nabla h(\mathbf{L}_\rho^*; \mathcal{X})\|_\infty > \frac{\lambda_\ell}{2}$ holds:

$$\frac{\lambda_\ell}{2} \geq 2B^2 R_F \left(\frac{(p\log 2 + 2)\log d_\ell}{c_\alpha}\right)^{\frac{1}{2}} \left(\frac{1}{\sqrt{d_{\setminus \ell}}} + \sum_{\ell' \neq \ell} \frac{1}{\sqrt{d_\ell d_{\ell'}}}\right) \quad (\text{when } 2^{1-p\log d_\ell} \geq \tfrac{2}{d_\ell^2} \exp\left(c_\alpha r_\ell^{\frac{2\alpha}{\alpha-4}}\right))$$

$$\frac{\lambda_\ell}{2} \geq 2B^2 R_{op} \left(\frac{(p\log 2 + 2)\log d_\ell}{c_\alpha}\right)^{\frac{2}{\alpha}} \sum_{\ell'} \frac{1}{\sqrt{d_{\setminus \ell} d_{\setminus \ell'}}} \quad (\text{when } 2^{1-p\log d_\ell} < \tfrac{2}{d_\ell^2} \exp\left(c_\alpha r_\ell^{\frac{2\alpha}{\alpha-4}}\right))$$

Observe that $\|\mathbf{A}^{(\ell)}\|_{op} \leq \|\mathbf{A}^{(\ell)}\|_F$. For simplicity, we can instead use the conditions:

$$\frac{\lambda_\ell}{2} \geq 2B^2 R_F \left(\frac{(p\log 2 + 2)\log d_\ell}{c_\alpha}\right)^{\frac{1}{2}} \left(\frac{1}{\sqrt{d_{\setminus \ell}}} + \sum_{\ell' \neq \ell} \frac{1}{\sqrt{d_\ell d_{\ell'}}}\right) \quad (\text{when } 2^{1-p\log d_\ell} \geq \tfrac{2}{d_\ell^2} \exp\left(c_\alpha r_\ell^{\frac{2\alpha}{\alpha-4}}\right))$$

$$\frac{\lambda_\ell}{2} \geq 2B^2 R_F \left(\frac{(p\log 2 + 2)\log d_\ell}{c_\alpha}\right)^{\frac{2}{\alpha}} \left(\frac{1}{\sqrt{d_{\setminus \ell}}} + \sum_{\ell' \neq \ell} \frac{1}{\sqrt{d_\ell d_{\ell'}}}\right) \quad (\text{when } 2^{1-p\log d_\ell} < \tfrac{2}{d_\ell^2} \exp\left(c_\alpha r_\ell^{\frac{2\alpha}{\alpha-4}}\right))$$

This formulation is not as tight in the tail regime, but it enables us to express our bounds concisely (below) by letting $\beta_\ell \in \{1, \frac{4}{\alpha}\}$ depending on the regime. As the tail regime is less important to us, the decrease in tightness is a reasonable trade-off.

$$\frac{\lambda_\ell}{2} \geq 2B^2 R_F \left(\frac{(p\log 2 + 2)\log d_\ell}{c_\alpha}\right)^{\frac{\beta_\ell}{2}} \left(\frac{1}{\sqrt{d_{\setminus \ell}}} + \sum_{\ell' \neq \ell} \frac{1}{\sqrt{d_\ell d_{\ell'}}}\right)$$

We are interested in giving bounds asymptotic in d_ℓ, which allows us to drop many nuisance parameters. Asymptotically, the following choice for λ_ℓ suffices:

$$\lambda_\ell \asymp \sqrt{\frac{(p\log d_\ell)^{\beta_\ell}}{d_{\setminus \ell}}} + \sum_{\ell' \neq \ell} \sqrt{\frac{(p\log d_\ell)^{\beta_\ell}}{d_\ell d_{\ell'}}}$$

This completes the proof. □

Corollary. *Let β_ℓ be either 1 or $\frac{4}{\alpha}$, depending on whether $2^{1-p\log d_\ell} \geq \frac{2}{d_\ell^2} \exp\left(c_\alpha r_\ell^{\frac{2\alpha}{\alpha-4}}\right)$. Suppose that:*

$$\lambda \asymp \max_\ell \sqrt{\frac{(p\log d_\ell)^{\beta_\ell}}{d_{\setminus \ell}}} + \sum_{\ell' \neq \ell} \sqrt{\frac{(p\log d_\ell)^{\beta_\ell}}{d_\ell d_{\ell'}}}$$

Then with high probability (probability at least $1 - \sum_\ell 2^{1-p\log d_\ell}$) $\|\nabla h(\mathbf{L}_\rho^*; \mathcal{X})\|_\infty \leq \frac{\lambda}{2}$.

Proof. This follows from union-bounding over each of the K axes. □

It can be hard to interpret the scaling of λ with d_ℓ due to the $\sum_{\ell' \neq \ell} \sqrt{\frac{(p\log d_\ell)^{\beta_\ell}}{d_\ell d_{\ell'}}}$ term, so we'll specialize this lemma to the common matrix-variate case to aid our intuition.

Corollary. *Suppose we have matrix-variate data ($K = 2$) and the conditions of Lemma 11 hold. Let β be either 1 or $\frac{4}{\alpha}$ as in the previous corollary. If $\lambda \asymp \max\left(\sqrt{\frac{(p\log d_1)^{\beta_1}}{d_2}}, \sqrt{\frac{(p\log d_2)^{\beta_2}}{d_1}}\right)$, then with probability at least $1 - \sum_\ell 2^{2-p\log d_\ell}$ we have that $\|\nabla h(\mathbf{L}_\rho^*; \mathcal{X})\|_\infty \leq \frac{\lambda}{2}$*

Proof. From Lemma 12, we have that $\lambda_1 \asymp \sqrt{\frac{(p\log d_1)^{\beta_1}}{d_2}} + \sqrt{\frac{(p\log d_1)^{\beta_1}}{d_1 d_2}} = \frac{\sqrt{d_1}\sqrt{(p\log d_1)^{\beta_1}} + \sqrt{(p\log d_1)^{\beta_1}}}{\sqrt{d_1 d_2}}$. This can be written as $\frac{(\sqrt{d_1}+1)}{\sqrt{d_1}} \frac{\sqrt{(p\log d_1)^{\beta_1}}}{\sqrt{d_2}}$, which is asymptotically equivalent to $\sqrt{\frac{(p\log d_1)^{\beta_1}}{d_2}}$, completing the proof. Without loss of generality, the same holds for λ_2, completing the proof.

□

The above results allow us to bound our gradient with high probability. We say 'high probability' because, as p increases, the probability of the bound failing drops exponentially. With this gradient bound, we are now equipped to bound our error.

Let $\hat{\mathbf{L}} = \mathbf{L}^*_\rho + \mathbf{\Delta}$ for some $\mathbf{\Delta}$. Let $\mathcal{S}$ be the set of nonzero entries of $\mathbf{L}^*_\rho$, with size s, and let (for any matrix $\mathbf{M}$) $\mathbf{M}_\mathcal{S}$ and $\mathbf{M}_{\mathcal{S}^c}$ be $\mathbf{M}$ restricted to $\mathcal{S}$ and its complement, respectively, with zeros everywhere else. Note also that $\mathcal{S}$ always contains the diagonal, so $\|\mathbf{M}_{\mathcal{S}^c}\|_{od} = \|\mathbf{M}_{\mathcal{S}^c}\|_1$.

Observe that $\|\mathbf{M} + \mathbf{\Delta}\|_{od} = \|(\mathbf{M} + \mathbf{\Delta})_\mathcal{S}\|_{od} + \|(\mathbf{M} + \mathbf{\Delta})_{\mathcal{S}^c}\|_{od}$ and that by the definition of $\mathcal{S}^c$ we have $\left(\mathbf{L}^*_\rho\right)_{S^c} = \mathbf{0}$.

Lemma 13 (Finite Sample Error)**.** *Suppose* $\|\nabla h(\mathbf{L}^*_\rho; \mathcal{X})\|_\infty \leq \frac{\lambda}{2}$.

$$\|\hat{\mathbf{L}} - \mathbf{L}^*_\rho\|_{F,\oplus} \leq \frac{3\lambda\sqrt{s}}{\mu + \rho}$$

Proof. The strategy will be to upper- and lower-bound $h(\hat{\mathbf{L}}, \mathcal{X}) - h(\mathbf{L}^*_\rho, \mathcal{X})$ in terms of norms of $\mathbf{\Delta} = \hat{\mathbf{L}} - \mathbf{L}^*_\rho$, which we can then manipulate into the desired form.

$$\begin{aligned}
& h(\hat{\mathbf{L}}, \mathcal{X}) + \lambda\|\hat{\mathbf{L}}\|_{od,\oplus} \leq h(\mathbf{L}^*_\rho, \mathcal{X}) + \lambda\|\mathbf{L}^*_\rho\|_{od,\oplus} && \text{(optimality of } \hat{\mathbf{L}}\text{)} \\
\implies \quad & h(\hat{\mathbf{L}}, \mathcal{X}) - h(\mathbf{L}^*_\rho, \mathcal{X}) \leq \lambda\left(\|\mathbf{L}^*_\rho\|_{od,\oplus} - \|\hat{\mathbf{L}}\|_{od,\oplus}\right) \\
& \qquad = \lambda\left(\|\mathbf{L}^*_\rho\|_{od,\oplus} - \|\mathbf{L}^*_\rho + \mathbf{\Delta}\|_{od,\oplus}\right) \\
& \qquad = \lambda\left(\|\left(\mathbf{L}^*_\rho\right)_\mathcal{S}\|_{od,\oplus} - \|\left(\mathbf{L}^*_\rho + \mathbf{\Delta}\right)_\mathcal{S}\|_{od,\oplus} - \|\mathbf{\Delta}_{\mathcal{S}^c}\|_{od,\oplus}\right) \\
& \qquad \leq \lambda\left(\|\mathbf{\Delta}_\mathcal{S}\|_{od,\oplus} - \|\mathbf{\Delta}_{\mathcal{S}^c}\|_{od,\oplus}\right) && \text{(reverse triangle inequality)} \\
& \qquad \leq \lambda\left(\|\mathbf{\Delta}_\mathcal{S}\|_{1,\oplus} - \|\mathbf{\Delta}_{\mathcal{S}^c}\|_{1,\oplus}\right)
\end{aligned}$$

$$\begin{aligned}
\lambda\left(\|\mathbf{\Delta}_\mathcal{S}\|_{1,\oplus} - \|\mathbf{\Delta}_{\mathcal{S}^c}\|_{1,\oplus}\right) &\geq h(\hat{\mathbf{L}}, \mathcal{X}) - h(\mathbf{L}^*_\rho, \mathcal{X}) \\
&\geq -\langle\nabla h(\mathbf{L}^*_\rho, \mathcal{X}), \mathbf{\Delta}\rangle + \frac{\mu + \rho}{2}\|\mathbf{\Delta}\|^2_{F,\oplus} && \text{(strong convexity)} \\
&\geq -\|\nabla h(\mathbf{L}^*_\rho, \mathcal{X})\|_\infty\|\mathbf{\Delta}\|_{1,\oplus} + \frac{\mu + \rho}{2}\|\mathbf{\Delta}\|^2_{F,\oplus} && \text{(Holder's inequality)} \\
&\geq -\frac{\lambda}{2}\|\mathbf{\Delta}\|_{1,\oplus} + \frac{\mu + \rho}{2}\|\mathbf{\Delta}\|^2_{F,\oplus} \\
\implies \quad \frac{\mu + \rho}{2}\|\mathbf{\Delta}\|^2_{F,\oplus} &\leq \frac{3\lambda}{2}\|\mathbf{\Delta}_\mathcal{S}\|_{1,\oplus} - \frac{\lambda}{2}\|\mathbf{\Delta}_{\mathcal{S}^c}\|_{1,\oplus} \\
&\leq \frac{3\lambda}{2}\|\mathbf{\Delta}_\mathcal{S}\|_{1,\oplus} \\
&\leq \frac{3\lambda\sqrt{s}}{2}\|\mathbf{\Delta}\|_{F,\oplus} \\
\implies \quad \|\mathbf{\Delta}\|_{F,\oplus} &\leq \frac{3\lambda\sqrt{s}}{\mu + \rho}
\end{aligned}$$

□

We can now put all of our results together to prove Theorem 6.

Theorem 6 (Error Bounds)**.** *Suppose the largest Frobenius norm of any factor of the regularized oracle solution* $\mathbf{L}^*_\rho$ *is* R_F*, the total amount of nonzero elements of the factors of the regularized oracle solution is* s*, we use the same regularization parameters* ρ, λ *for each factor, and* μ *is the strong convexity strength (which depends on the data). Suppose our data is* α*-sub-exponential, with* β_ℓ *defined as in Lemma 12. Let* $\lambda \asymp \max_\ell \sqrt{\frac{(p\log d_\ell)^{\beta_\ell}}{d_{\setminus\ell}}} + \sum_{\ell' \neq \ell}\sqrt{\frac{(p\log d_\ell)^{\beta_\ell}}{d_\ell d_{\ell'}}}$*, then with probability at least* $1 - \sum_\ell 2^{1 - p\log d_\ell}$*:*

$$\|\hat{\mathbf{L}} - \mathbf{L}^*\|_{F,\oplus} \leq \frac{2K\rho}{\mu}R_F + \frac{3\lambda\sqrt{s}}{\mu + \rho}$$

Proof. This follows from Lemmas 10, 12, and 13 and a straightforward application of the triangle inequality. □

The final proof, of ordering recovery conditions, is easy.

Theorem 9 (Ordering Recovery)**.** *Let $\mathcal{X}$ come from the $\boxplus$-LGAM model. Under various conditions (Park, 2020; Peters & Bühlmann, 2014), we have that:*

$$i < i' \implies \mathbb{E}\left[\sum_{\mathbf{j}} \mathcal{X}_{i\mathbf{j}}^2\right] < \mathbb{E}\left[\sum_{\mathbf{j}} \mathcal{X}_{i'\mathbf{j}}^2\right]$$

One such condition is that the variances are homogeneous.

Proof. Note that $i < i' \implies (i, \mathbf{j}) < (i', \mathbf{j})$ due to properties of the Cartesian product. Due to Peters, Bühlmann, and Park, we have various conditions (such as homogenous variances) for LGAMs under which $i < i' \implies \operatorname{Var}[\mathcal{X}_i] < \operatorname{Var}[\mathcal{X}_{i'}]$; in this case, $(i, \mathbf{j}) < (i', \mathbf{j}) \implies \operatorname{Var}\left[\mathcal{X}_{i\mathbf{j}}\right] < \operatorname{Var}\left[\mathcal{X}_{i'\mathbf{j}}\right]$.

$$\begin{aligned}
\mathbb{E}\left[\mathcal{X}_{i\mathbf{j}}^2\right] &= \operatorname{Var}\left[\mathcal{X}_{i\mathbf{j}}\right] \\
&< \operatorname{Var}\left[\mathcal{X}_{i'\mathbf{j}}\right] && \text{(Under various conditions)} \\
&= \mathbb{E}\left[\mathcal{X}_{i'\mathbf{j}}^2\right]
\end{aligned}$$

Sum over all $\mathbf{j}$ to complete the proof. □

Bibliography for Paper 4 (appendix)

Cover, T. M. (1965). Geometrical and Statistical Properties of Systems of Linear Inequalities with Applications in Pattern Recognition. *IEEE Transactions on Electronic Computers*, *EC-14*(3), 326–334. https://doi.org/10.1109/PGEC.1965.264137

Park, G. (2020). Identifiability of Additive Noise Models Using Conditional Variances. *Journal of Machine Learning Research*, *21*(75), 1–34. Retrieved April 8, 2025, from http://jmlr.org/papers/v21/19-664.html

Peters, J., & Bühlmann, P. (2014). Identifiability of Gaussian structural equation models with equal error variances. *Biometrika*, *101*(1), 219–228. https://doi.org/10.1093/biomet/ast043

Sambale, H. (2023). Some Notes on Concentration for α-Subexponential Random Variables. In R. Adamczak, N. Gozlan, K. Lounici, & M. Madiman (Eds.), *High Dimensional Probability IX* (pp. 167–192). Springer International Publishing. https://doi.org/10.1007/978-3-031-26979-0_7

Appendix E

Making multi-axis Gaussian graphical models scalable to millions of samples and features

E.1 Evidence for the low-rank approximability assumption

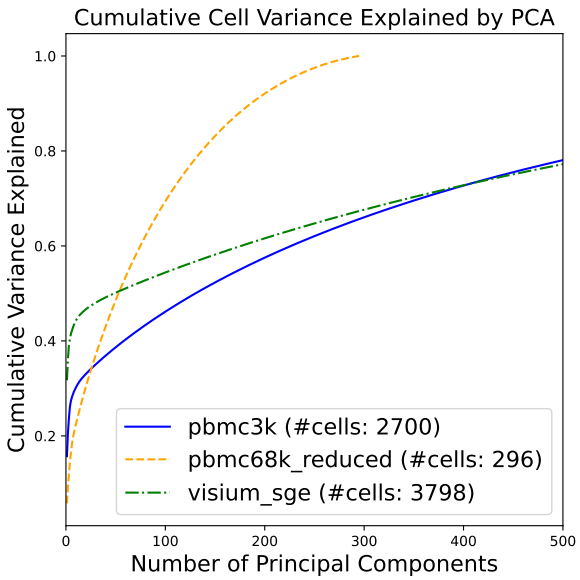


Figure E.1: The cumulative variance explained by low-rank approximations (PCA) for three scRNA-seq datasets.

The low-rank approximability assumption (Assumption 11) is frequently made in practice; whenever one performs PCA, this assumption is being made. This section aims to demonstrate the reasonability of Assumption 11. As PCA is likely to be most practitioner's primary encounter with low-rank approximations, we will frame our argument in terms of PCA.

The number of principal components to chose (and hence the rank of the low-rank approximation) is typically chosen by looking at the total explained variance of the components. An explained variance of 100% indicates perfect representation of the original matrix by the low-rank approximation. One method of picking the amount of principal components is to simply pick an amount that gives a satisfactory recovery rate, such as 60% or 80%. However, most components often represent random noise; we do not care about recovering them. An alternative selection procedure is to look for 'sharp corners' in the plot of cumulative variance, indicating the transition from informative to uninformative

components. The first approach is liable to pick too many components, whereas the second approach is liable to pick too few.

Figure E.1 displays the cumulative variance plot for three scRNA-seq datasets available in the ScanPy package and produced by 10X Genomics (Wolf et al., 2018). Figure E.1 shows that one does not need too many principal components to achieve good recovery; the explained variance approach requires a couple hundred components and the informative components approach requires less than a hundred for the larger datasets. This validates our use of low-rank approximations.

E.2 Derivation of the probability distribution

The space of all full-rank $d \times d$ matrices has dimension d^2. In contrast, the space of all rank-r $d \times d$ matrices has dimension $2dr - r^2$. Thus, any probability distribution over rank-r matrices must be degenerate, or 'singular', in the sense that the density is only positive on a subset of measure 0. First, let's consider a multivariate Gaussian of independent variables:

$$\begin{aligned} & \mathbf{x} \sim \mathcal{N}\left(\mathbf{0}, \operatorname{diag}\left[\boldsymbol{\sigma}\right]\right) \\ \iff \qquad & \operatorname{pdf}\left(\mathbf{x}\right) = \frac{\sqrt{\prod_i \frac{1}{\sigma_i}}}{(2\pi)^{\frac{r}{2}}} e^{-\frac{1}{2}\mathbf{x}^T \operatorname{diag}[\boldsymbol{\sigma}]^{-1}\mathbf{x}} \qquad (r \text{ Independent Normal Variables}) \end{aligned}$$

Suppose we then left-multiply $\mathbf{x}$ by an $r \times r$ orthonormal matrix $\mathbf{V}$ to introduce dependencies into our data while preserving Gaussianity. Letting $\mathbf{V}\operatorname{diag}\left[\boldsymbol{\sigma}\right]^{-1}\mathbf{V} = \boldsymbol{\Psi}$, we get:

$$\begin{aligned} \operatorname{pdf}\left(\mathbf{V}\mathbf{x}\right) &= \frac{\sqrt{\prod_i \frac{1}{\sigma_i}}}{(2\pi)^{\frac{r}{2}}} e^{-\frac{1}{2}\mathbf{x}^T \mathbf{V} \operatorname{diag}[\boldsymbol{\sigma}]^{-1} \mathbf{V}^T \mathbf{x}} \\ &= \frac{\sqrt{\det \boldsymbol{\Psi}}}{(2\pi)^{\frac{r}{2}}} e^{-\frac{1}{2}\mathbf{x}^T \boldsymbol{\Psi} \mathbf{x}} \end{aligned}$$

Notice that $\boldsymbol{\Psi}$ is fully general; $\mathbf{V}$ are the eigenvectors and $\boldsymbol{\sigma}^{-1}$ are the eigenvalues. To achieve a singular probability distribution, we multiply by a $d \times r$ orthonormal matrix $\mathbf{U}$ ($d > r$) instead. The only noticeable change in the distribution's specification is the use of the pseudo-determinant (the product of only the non-zero eigenvalues of $\boldsymbol{\Psi} = \mathbf{U}\operatorname{diag}\left[\boldsymbol{\sigma}\right]\mathbf{U}^T$).

$$\operatorname{pdf}\left(\mathbf{U}\mathbf{x}\right) = \frac{\sqrt{\det{}^{\dagger}\, \boldsymbol{\Psi}}}{(2\pi)^{\frac{r}{2}}} e^{-\frac{1}{2}\mathbf{x}^T \boldsymbol{\Psi} \mathbf{x}}$$

The above formulas were for vector-variate $\mathbf{x}$; for tensor-variate $\mathcal{X}$ we can use the identity $\operatorname{vec}\left[\mathcal{X} \times_1 \mathbf{V}_1 \times_2 \cdots \times_K \mathbf{V}_K\right] = \left(\bigoplus_{\ell=K}^{1} \mathbf{V}_\ell\right) \operatorname{vec}\left[\mathcal{X}\right]$. It is then straightforward to present a Kronecker-sum-structured version of this distribution, the 'singular Kronecker-sum-structured normal distribution'. We let r_ℓ be the rank chosen for each axis, and $r_\forall^\gamma$ be the product of ranks of all axes occurring in modality γ.

$$\begin{aligned} \operatorname{pdf}_{\text{normal}}(\mathcal{X}^\gamma) &= \frac{\sqrt{\det{}^{\dagger}\left(\bigoplus_{\ell \in \gamma} \boldsymbol{\Psi}_\ell\right)}}{(2\pi)^{\frac{r_\forall^\gamma}{2}}} e^{-\frac{1}{2}\operatorname{vec}[\mathcal{X}^\gamma]^T \left(\bigoplus_{\ell \in \gamma} \boldsymbol{\Psi}_\ell\right) \operatorname{vec}[\mathcal{X}^\gamma]} \\ &= \frac{\sqrt{\det{}^{\dagger}\left(\bigoplus_{\ell \in \gamma} \boldsymbol{\Psi}_\ell\right)}}{(2\pi)^{\frac{r_\forall^\gamma}{2}}} e^{-\frac{1}{2}\sum_{\ell \in \gamma} \operatorname{tr}\left[\mathbf{S}_\ell^\gamma \boldsymbol{\Psi}_\ell\right]} \end{aligned}$$

When given a multi-modal dateset, we have:

$$\text{pdf}_{\text{normal}}\left(\{\mathcal{X}^\gamma\}\right) = \prod_\gamma \frac{\sqrt{\det{}^\dagger\left(\bigoplus_{\ell\in\gamma}\mathbf{\Psi}_\ell\right)}}{(2\pi)^{\frac{r^\gamma_\forall}{2}}} e^{-\frac{1}{2}\sum_{\ell\in\gamma}\text{tr}\left[\mathbf{S}^\gamma_\ell\mathbf{\Psi}_\ell\right]}$$

E.3 Identifiability

The Kronecker sum decomposition is typically parameterized by a non-identifiable representation of $\bigoplus_{\ell\in\gamma}\mathbf{\Psi}_\ell$. Let $\{c_\ell\}_{\ell\in\gamma}$ be a set of constants such that $\sum_{\ell\in\gamma}c_\ell = 0$, then:

$$\bigoplus_{\ell\in\gamma}\left(\mathbf{\Psi}_\ell + c_\ell\mathbf{I}_{d_\ell}\right) = \bigoplus_{\ell\in\gamma}\mathbf{\Psi}_\ell + \left(\sum_{\ell\in\gamma}c_\ell\right)\mathbf{I}_{d^\gamma_\forall} = \bigoplus_{\ell\in\gamma}\mathbf{\Psi}_\ell$$

This is not unique to the Kronecker sum model; all Kronecker-separable models experience some form of non-identifiability. However, this non-identifiability is rather benign. Firstly, it only affects the diagonals of our estimates; the off-diagonals (which encode the graph structure) are unaffected. Secondly, Greenewald et al., 2019 derived the following identifiable parameterization for the single-modality case:

$$\begin{aligned}\mathbf{\Omega}^\gamma &= \left(\bigoplus_{\ell\in\gamma}\tilde{\mathbf{\Psi}}_\ell\right) + \tau^\gamma\mathbf{I}_{d^\gamma_\forall}\\ \textit{where}\ \ &\text{tr}\left[\tilde{\mathbf{\Psi}}_\ell\right] = 0\end{aligned}$$

We can map the standard parameterization to this one with ease. As a shorthand, let $\tau_\ell = \text{tr}\left[\mathbf{\Psi}_\ell\right]$:

$$\begin{aligned}\tau^\gamma &= \sum_{\ell\in\gamma} d^\gamma_{\setminus\ell}\tau_\ell\\ \tilde{\mathbf{\Psi}}_\ell &= \mathbf{\Psi}_\ell - \tau_\ell\mathbf{I}_{d_\ell}\end{aligned}$$

However, Greenewald et al.'s parameterization does not apply to the multi-modality case. Ideally, we would wish to parameterize our dataset by the parameters $\{\tau^\gamma\}_\gamma$ and $\{\tilde{\mathbf{\Psi}}_\ell\}_\ell$, but there are linear dependencies in our parameterization which may prevent this.

Example 26. *Consider the bi-modal dataset where $\gamma_1 = (\ell_1, \ell_2)$ and $\gamma_2 = (\ell_1, \ell_2, \ell_3)$. There is a linear dependency in our trace parameters:*

$$\begin{aligned}\tau^{\gamma_1} &= d_2\tau_1 + d_1\tau_2\\ \tau^{\gamma_2} &= d_2d_3\tau_1 + d_1d_3\tau_2 + d_2d_3\tau_3\\ &= d_3\tau^{\gamma_1} + d_2d_3\tau_3\end{aligned}$$

In Example 26, the parameterization is still valid - we need both τ^{γ_1} and τ^{γ_2} to identify the parameterization. However, we also get the trace of the third axis, τ_3, as a byproduct. Unlike in the single modality case, in this case $\mathbf{\Psi}_3$ is fully-specified.

Example 27. *Consider the tri-modal dataset where $\gamma_1 = (\ell_1, \ell_2)$, $\gamma_2 = (\ell_3, \ell_4)$, and $\gamma_3 = (\ell_1, \ell_2, \ell_3, \ell_4)$. Then $\tau^{\gamma_3} = d_3d_4\tau^{\gamma_1} + d_1d_2\tau^{\gamma_2}$.*

Example 27 shows that Greenewald et al.'s parameterization can be over-determined in the multi-modal case. GmGM can be used on arbitrarily complicated multi-tensor-variate data; how can we know when a parameterization is over-determined or not?

Note that $\{\tau^\gamma\}$ are always a linear combination of τ_ℓ. Letting $d^\gamma_{\setminus\ell} = 0$ if $\ell \notin \gamma$, we can express this linear relationship as:

$$\begin{bmatrix} \tau^{\gamma_1} \\ \vdots \\ \tau^{\gamma_\Gamma} \end{bmatrix} = \mathbf{M}_{\boldsymbol{\gamma}} \begin{bmatrix} \tau_{\ell_1} \\ \vdots \\ \tau_{\ell_L} \end{bmatrix}$$

$$\mathbf{M}_{\boldsymbol{\gamma}} = \begin{bmatrix} d^{\gamma_1}_{\backslash \ell_1} & \cdots & d^{\gamma_1}_{\backslash \ell_L} \\ \vdots & \ddots & \vdots \\ d^{\gamma_\Gamma}_{\backslash \ell_1} & \cdots & d^{\gamma_\Gamma}_{\backslash \ell_L} \end{bmatrix}$$

Depending on the modalities and the lengths of each axis, $\mathbf{M}_{\boldsymbol{\gamma}}$ could be practically any $\Gamma \times L$ matrix with nonnegative integer coefficients, so it is hard to say much about it in general. However, we can make a few observations. In particular, $\mathbf{M}_{\boldsymbol{\gamma}}$ describes how our identifiable quantities $\{\tau^\gamma\}_\gamma$ are linked through the latent factors $\{\tau_\ell\}_\ell$. Zero-eigenvectors $\mathbf{x}$ of $\mathbf{M}_{\boldsymbol{\gamma}}$ correspond to the non-identifiabilities, as $\mathbf{M}_{\boldsymbol{\gamma}}(\mathbf{x}+\mathbf{y}) = \mathbf{M}_{\boldsymbol{\gamma}}\mathbf{y}$. Our parameter vector $\begin{bmatrix} \tau^{\gamma_1} & \cdots & \tau^{\gamma_\Gamma} \end{bmatrix}^T$ must lie in the *rowspace* of $\mathbf{M}_{\boldsymbol{\gamma}}$.

For most datasets, $\mathbf{M}_{\boldsymbol{\gamma}}$ has maximal row rank, in which case the approach of Greenewald et al. is unchanged, and we can parameterize by $\{\tau^\gamma\}_\gamma$. However, there are sometimes linear dependencies in $\{\tau^\gamma\}_\gamma$, causing us to choose a new, smaller parameterization that functions as a basis for this rowspace. Table E.1 gives example parameterizations of the most common scenarios along with some unlikely-but-demonstrative ones.

In practice, the choice of parameterization is not too important; it does not have an effect on how we go about finding solutions, nor does it have an effect on the conditional dependency graph. However, it will be useful for hypothesis testing, which we will discuss in the next section.

Dataset	$\mathbf{M}_{\boldsymbol{\gamma}}$	Basis Vectors	Parameterization
$\gamma_1 = (\ell_1, \ell_2)$	$\begin{bmatrix} d_{\ell_2} & d_{\ell_1} \end{bmatrix}$	$\begin{bmatrix} \mathbf{e}_1 \end{bmatrix} = \begin{bmatrix} d_{\ell_2} & d_{\ell_1} \end{bmatrix}$	$\boldsymbol{\Omega}^{\gamma_1} = t_1 \mathbf{I}_{d_\forall^{\gamma_1}} + \bigotimes_{\ell \in \gamma_1} \tilde{\boldsymbol{\Psi}}_\ell$
$\gamma_1 = (\ell_1, ..., \ell_L)$	$\begin{bmatrix} d_{\setminus \ell_1} & ... & d_{\setminus \ell_L} \end{bmatrix}$	$\begin{bmatrix} \mathbf{e}_1 \end{bmatrix} = \begin{bmatrix} d_{\setminus \ell_1} & ... & d_{\setminus \ell_L} \end{bmatrix}$	$\boldsymbol{\Omega}^{\gamma_1} = t_1 \mathbf{I}_{d_\forall^{\gamma_1}} + \bigotimes_{\ell \in \gamma_1} \tilde{\boldsymbol{\Psi}}_\ell$
$\left\{ \gamma_i = (\ell_0, \ell_i) \right\}_{1 \le i \le \Gamma}$	$\begin{bmatrix} d_{\ell_1} & d_{\ell_0} & 0 & ... \\ d_{\ell_2} & 0 & d_{\ell_0} & ... \\ d_{\ell_3} & 0 & 0 & ... \\ \vdots & \vdots & \vdots & \ddots \end{bmatrix}$	$\begin{bmatrix} \mathbf{e}_1 \\ \vdots \end{bmatrix} = \begin{bmatrix} d_{\ell_1} & d_{\ell_0} & 0 & ... \\ d_{\ell_2} & 0 & d_{\ell_0} & ... \\ d_{\ell_3} & 0 & 0 & ... \\ \vdots & \vdots & \vdots & \ddots \end{bmatrix}$	$\begin{bmatrix} \boldsymbol{\Omega}^{\gamma_1} \\ \vdots \end{bmatrix} = \begin{bmatrix} t_1 \mathbf{I}_{d_\forall^{\gamma_1}} + \bigotimes_{\ell \in \gamma_1} \tilde{\boldsymbol{\Psi}}_\ell \\ \vdots \end{bmatrix}$
$\gamma_1 = (\ell_1, \ell_2)$ $\gamma_2 = (\ell_2, \ell_3)$ $\gamma_3 = (\ell_3, \ell_1)$	$\begin{bmatrix} d_{\ell_2} & d_{\ell_1} & 0 \\ 0 & d_{\ell_3} & d_{\ell_2} \\ d_{\ell_3} & 0 & d_{\ell_1} \end{bmatrix}$	$\begin{bmatrix} \mathbf{e}_1 \\ \mathbf{e}_2 \\ \mathbf{e}_3 \end{bmatrix} = \begin{bmatrix} d_{\ell_2} & d_{\ell_1} & 0 \\ 0 & d_{\ell_3} & d_{\ell_2} \\ d_{\ell_3} & 0 & d_{\ell_1} \end{bmatrix}$	$\begin{bmatrix} \boldsymbol{\Omega}^{\gamma_1} \\ \boldsymbol{\Omega}^{\gamma_2} \\ \boldsymbol{\Omega}^{\gamma_3} \end{bmatrix} = \begin{bmatrix} t_1 \mathbf{I}_{d_\forall^{\gamma_1}} + \bigotimes_{\ell \in \gamma_1} \tilde{\boldsymbol{\Psi}}_\ell \\ t_2 \mathbf{I}_{d_\forall^{\gamma_2}} + \bigotimes_{\ell \in \gamma_2} \tilde{\boldsymbol{\Psi}}_\ell \\ t_3 \mathbf{I}_{d_\forall^{\gamma_3}} + \bigotimes_{\ell \in \gamma_3} \tilde{\boldsymbol{\Psi}}_\ell \end{bmatrix}$
$\gamma_1 = (\ell_1, \ell_2)$ $\gamma_2 = (\ell_1, \ell_2, \ell_3)$ $\gamma_3 = (\ell_2, \ell_3, \ell_4)$	$\begin{bmatrix} d_{\ell_2} & d_{\ell_1} & 0 & 0 \\ d_{\ell_2} d_{\ell_3} & d_{\ell_1} d_{\ell_3} & d_{\ell_1} d_{\ell_2} & 0 \\ 0 & d_{\ell_3} d_{\ell_4} & d_{\ell_2} d_{\ell_4} & d_{\ell_2} d_{\ell_3} \end{bmatrix}$	$\begin{bmatrix} \mathbf{e}_1 \\ \mathbf{e}_2 \\ \mathbf{e}_3 \end{bmatrix} = \begin{bmatrix} d_{\ell_2} & d_{\ell_1} & 0 & 0 \\ d_{\ell_2} d_{\ell_3} & d_{\ell_1} d_{\ell_3} & d_{\ell_1} d_{\ell_2} & 0 \\ 0 & d_{\ell_3} d_{\ell_4} & d_{\ell_2} d_{\ell_4} & d_{\ell_2} d_{\ell_3} \end{bmatrix}$	$\begin{bmatrix} \boldsymbol{\Omega}^{\gamma_1} \\ \boldsymbol{\Omega}^{\gamma_2} \\ \boldsymbol{\Omega}^{\gamma_3} \end{bmatrix} = \begin{bmatrix} t_1 \mathbf{I}_{d_\forall^{\gamma_1}} + \bigotimes_{\ell \in \gamma_1} \tilde{\boldsymbol{\Psi}}_\ell \\ t_2 \mathbf{I}_{d_\forall^{\gamma_2}} + \bigotimes_{\ell \in \gamma_2} \tilde{\boldsymbol{\Psi}}_\ell \\ t_3 \mathbf{I}_{d_\forall^{\gamma_3}} + \bigotimes_{\ell \in \gamma_3} \tilde{\boldsymbol{\Psi}}_\ell \end{bmatrix}$
$\gamma_1 = (\ell_1, \ell_2)$ $\gamma_2 = (\ell_2, \ell_1)$	$\begin{bmatrix} d_{\ell_2} & d_{\ell_1} \\ d_{\ell_2} & d_{\ell_1} \end{bmatrix}$	$\begin{bmatrix} \mathbf{e}_1 \end{bmatrix} = \begin{bmatrix} d_{\ell_2} & d_{\ell_1} \end{bmatrix}$	$\begin{bmatrix} \boldsymbol{\Omega}^{\gamma_1} \\ \boldsymbol{\Omega}^{\gamma_2} \end{bmatrix} = \begin{bmatrix} t_1 \mathbf{I}_{d_\forall^{\gamma_1}} + \bigotimes_{\ell \in \gamma_1} \tilde{\boldsymbol{\Psi}}_\ell \\ t_1 \mathbf{I}_{d_\forall^{\gamma_2}} + \bigotimes_{\ell \in \gamma_2} \tilde{\boldsymbol{\Psi}}_\ell \end{bmatrix}$
$\gamma_1 = (\ell_1, \ell_2)$ $\gamma_2 = (\ell_3, \ell_4)$ $\gamma_3 = (\ell_1, \ell_2, \ell_3, \ell_4)$	$\begin{bmatrix} d_{\ell_2} & d_{\ell_1} & 0 & 0 \\ 0 & 0 & d_{\ell_4} & d_{\ell_3} \\ d_{\setminus \ell_1} & d_{\setminus \ell_2} & d_{\setminus \ell_3} & d_{\setminus \ell_4} \end{bmatrix}$	$\begin{bmatrix} \mathbf{e}_1 \\ \mathbf{e}_2 \end{bmatrix} = \begin{bmatrix} d_{\ell_2} & d_{\ell_1} & 0 & 0 \\ 0 & 0 & d_{\ell_4} & d_{\ell_3} \end{bmatrix}$	$\begin{bmatrix} t_1 \mathbf{I}_{d_\forall^{\gamma_1}} + \bigotimes_{\ell \in \gamma_1} \tilde{\boldsymbol{\Psi}}_\ell \\ t_2 \mathbf{I}_{d_\forall^{\gamma_2}} + \bigotimes_{\ell \in \gamma_2} \tilde{\boldsymbol{\Psi}}_\ell \\ (d_{\ell_3} d_{\ell_4} t_1 + d_{\ell_1} d_{\ell_2} t_2) \mathbf{I}_{d_\forall^{\gamma_2}} + \bigotimes_{\ell \in \gamma_2} \tilde{\boldsymbol{\Psi}}_\ell \end{bmatrix}$

Table E.1: Identifiable parameterizations for several datasets. Here, t_i is the coefficient corresponding to basis vector $\mathbf{e}_i$, i.e, t_i is one of the parameters, along with $\tilde{\boldsymbol{\Psi}}_\ell$. Often, $t_i = \tau^{\gamma_i}$. Note that many different bases are possible; our selection here is somewhat arbitrary. As can be seen, typically $\mathbf{M}_\gamma$ itself can be used for the basis vectors, but sometimes there can be linear dependencies in the data. The first three rows in this matrix correspond to the cases most likely to arise in practice; a matrix-variate dataset, a tensor-variate dataset, and a multi-modality matrix-variate dataset with a consistently shared axis. Cells with a grey background indicate a smaller parameterization than that of Greenewald et al.

E.4 The algorithm

Algorithm 3 The improved GmGM algorithm

Input: Dataset $\{\mathcal{D}^\gamma\}$, minimum eigenvalues $\{\epsilon_\ell\}$, sparsity $\{n_\ell\}$, principal components $\{r_\ell\}$
Output: $\{\mathbf{\Psi}_\ell\}$

for $1 \leq \ell \leq K$
  $\mathbf{D}_\ell \leftarrow \left[\text{mat}_\ell\left[\mathcal{D}^{\gamma_1}\right] \quad \cdots \quad \text{mat}_\ell\left[\mathcal{D}^{\gamma_\Gamma}\right]\right]$
  $\mathbf{V}_\ell^{(r_\ell)} \leftarrow$ top r_ℓ left singular vectors of $\mathbf{D}_\ell$ ▷ Theorem 10
  $\mathbf{E}_\ell^{(r_\ell)} \leftarrow$ (top r_ℓ singular values of $\mathbf{D}_\ell)^2$
end for
$\mathbf{\Lambda}_\ell^{0,(r_\ell)} \leftarrow \begin{bmatrix} \frac{1}{E_{\ell_{11}}^{(r_\ell)}} & \cdots & \frac{1}{E_{\ell_{r_\ell r_\ell}}^{(r_\ell)}} \end{bmatrix}^T$
$\mu \leftarrow 1$
while not converged
  while $\exists\gamma \sum_{\ell\in\gamma} \min \mathbf{\Lambda}_\ell^{t+1,(r_\ell)} < \epsilon_\ell$ ▷ Ensure smallest eigenvalue is at least ϵ_ℓ
    Decrease μ if needed
    for $1 \leq \ell \leq K$ ▷ Gradient descent on eigenvalues
      $\mathbf{\Lambda}_\ell^{t+1,(r_\ell)} \leftarrow \mathbf{\Lambda}_\ell^{t,(r_\ell)} - \frac{\mu}{2}\left(\mathbf{E}_\ell^{(r_\ell)} - \sum_{\gamma|\ell\in\gamma} \text{tr}_{k^\gamma_{>\ell}}^{k^\gamma_{\leq\ell}}\left[\left(\bigoplus_{\ell'\in\gamma} \mathbf{\Lambda}_{\ell'}^{t,(r_{\ell'})}\right)^{-1}\right]\right)$
    end for
  end while
end while
for $1 \leq \ell \leq K$
  $\mathbf{\Psi}_\ell \leftarrow \text{thresh}_{n_\ell}\left[\mathbf{V}_\ell^{(r_\ell)} \mathbf{\Lambda}_\ell^{t,(r_\ell)} \mathbf{V}_\ell^{T,(r_\ell)}\right]$ ▷ Threshold simultaneously with recomposition
end for

E.5 Computational complexity

Calculating the top r_ℓ singular vectors for each axis's $d_\ell \times d_{\forall\setminus\ell}$ matrix $\mathbf{D}_\ell$ takes $O(\sum_\ell r_\ell d_{\forall|\ell})$ time and $O(\sum_\ell r_\ell d_\ell)$ space. Per iteration, the convex optimization step requires computing $\left(\bigoplus_{\ell\in\gamma} \mathbf{\Lambda}_\ell^{(r_\ell)}\right)^{-1}$, which takes $O(\sum_\gamma \prod_{\ell\in\gamma} r_\ell)$ time and space. The eigen-recomposition step takes $O(\sum_\ell r_\ell d_\ell^2)$ time, with thresholding talking $O(\sum d_\ell^2)$ time - although our implementation when not thresholding by significance takes $O(\sum_\ell d_\ell^2 \log n_\ell)$ to minimize the number of times we pass through the data. As we threshold and eigen-recompose simultaneously, it requires $O(n_\ell)$ space.

The Nonparanormal Skeptic requires ranking the data, taking $O(\sum_\ell d_{\forall|\ell} \log d_\ell)$ time and $O(sd_\forall)$ space, where s is the sparsity of the input dataset. This leads to an algorithm with the following complexity:

$$O\left(\sum_\gamma \prod_{\ell\in\gamma} r_\ell + \sum_\ell \left(r_\ell d_{\forall|\ell} + r_\ell d_\ell^2 \log n_\ell + d_{\forall|\ell} \log d_\ell\right)\right) \quad \text{(time)}$$

$$O\left(\sum_\gamma \prod_{\ell\in\gamma} r_\ell + \sum_\ell \left(r_\ell d_\ell + n_\ell\right) + sd_\forall\right) \quad \text{(space)}$$

Blue terms arise only if using the Nonparanormal Skeptic, and the grey term is a product of our implementation; it could be dropped, but would in practice make the algorithm slower. As this formula is complicated, we will give an example of a special case. In the case of matrix-variate data with $n_\ell \in O(d_\ell)$, and constant r_ℓ (the case for many real-world datasets), as well as for simplicity assuming $d = d_{\ell_1} \approx d_{\ell_2}$ and thresholding by statistical significance, these become:

$$O\left(d^2 + d^2 \log d\right) \quad \text{(time)}$$

$$O\left(d + sd^2\right) \quad \text{(space)}$$

In such a scenario, the optimal runtime would be $\Omega\left(sd^2\right)$ as it must visit every element at least once. Use of the Nonparanormal Skeptic increases the asymptotics, but as long as the dataset is sparse such an increase is modest.

E.6 Kronecker product and stridewise-blockwise trace lemmas

This section contains the derivation of a few lemmas pertaining to Kronecker products. The stridewise-blockwise trace often appears in derivatives involving Kronecker products, and thus is integral to our method as well. The first two are known from prior work.

Lemma (Dahl et al. (2013) Lemma 1; Extraction Property)**.**

$$\operatorname{tr}\left[\left(\mathbf{I}_{a\times a} \otimes \mathbf{X} \otimes \mathbf{I}_{b\times b}\right)\mathbf{M}\right] = \operatorname{tr}\left[\mathbf{X}\operatorname{tr}_b^a\left[\mathbf{M}\right]\right]$$

Lemma (Cyclic Property (Andrew et al., 2024))**.**

$$\operatorname{tr}_b^a\left[\left(\mathbf{A}_{a\times a} \otimes I \otimes \mathbf{B}_{b\times b}\right)\mathbf{M}\right] = \operatorname{tr}_b^a\left[\mathbf{M}\left(\mathbf{A}_{a\times a} \otimes I \otimes \mathbf{B}_{b\times b}\right)\right]$$

The next lemma is a generalized version of lemmas from prior work; an 'extraction' property (a generalized Lemma 2 from Dahl et al. (2013)).

Lemma 14 (Extraction Property)**.**

$$\operatorname{tr}_b^a\left[\left(\mathbf{I}_a \otimes \mathbf{X} \otimes \mathbf{I}_b\right)\mathbf{M}\left(\mathbf{I}_a \otimes \mathbf{Y}^T \otimes \mathbf{I}_b\right)\right] = \mathbf{X}\operatorname{tr}_b^a\left[\mathbf{M}\right]\mathbf{Y}^T$$

Proof. This proof follows the original by Dahl et al. (2013) closely; the only differences are that it is proven in the tensor-variate case and that $\mathbf{X}$ and $\mathbf{Y}$ are not constrained to be the same. The differences do not significantly affect the proof.

$$\begin{aligned}
\operatorname{tr}_b^a\left[\left(\mathbf{I}_a \otimes \mathbf{X} \otimes \mathbf{I}_b\right)\mathbf{M}\left(\mathbf{I}_a \otimes \mathbf{Y}^T \otimes \mathbf{I}_b\right)\right] &= \operatorname{tr}\left[\left(\mathbf{I}_a \otimes \mathbf{X} \otimes \mathbf{I}_b\right)\mathbf{M}\left(\mathbf{I}_a \otimes \mathbf{Y}^T \otimes \mathbf{I}_b\right)\left(\mathbf{I}_a \otimes \mathbf{J}^{ij} \otimes \mathbf{I}_b\right)\right]_{ij} \\
&= \operatorname{tr}\left[\left(\mathbf{I}_a \otimes \mathbf{j}^j\mathbf{X} \otimes \mathbf{I}_b\right)\mathbf{M}\left(\mathbf{I}_a \otimes \mathbf{Y}^T\mathbf{j}^{i,T} \otimes \mathbf{I}_b\right)\right]_{ij} \\
&= \operatorname{tr}\left[\left(\mathbf{I}_a \otimes \mathbf{X}_j \otimes \mathbf{I}_b\right)\mathbf{M}\left(\mathbf{I}_a \otimes \mathbf{Y}_i^T \otimes \mathbf{I}_b\right)\right]_{ij} \\
&= \operatorname{tr}\left[\left(\mathbf{I}_a \otimes \otimes\mathbf{Y}_i^T\mathbf{X}_j \otimes \mathbf{I}_b\right)\mathbf{M}\right]_{ij} \\
&= \operatorname{tr}\left[\mathbf{Y}_i^T\mathbf{X}_j\operatorname{tr}_b^a\left[\mathbf{M}\right]\right]_{ij} \\
&= \operatorname{tr}\left[\mathbf{X}_j\operatorname{tr}_b^a\left[\mathbf{M}\right]\mathbf{Y}_i^T\right]_{ij}
\end{aligned}$$

Note now that the value inside the trace is a scalar value, and hence we can drop the outer trace.

$$\operatorname{tr}_b^a\left[\left(\mathbf{I}_a \otimes \mathbf{X} \otimes \mathbf{I}_b\right)\mathbf{M}\left(\mathbf{I}_a \otimes \mathbf{Y}^T \otimes \mathbf{I}_b\right)\right] = \left[\mathbf{X}_j\operatorname{tr}_b^a\left[\mathbf{M}\right]\mathbf{Y}_i^T\right]_{ij}$$

Which leads us to the final result.

$$\operatorname{tr}_b^a\left[\left(\mathbf{I}_a \otimes \mathbf{X} \otimes \mathbf{I}_b\right)\mathbf{M}\left(\mathbf{I}_a \otimes \mathbf{Y}^T \otimes \mathbf{I}_b\right)\right] = \mathbf{X}\operatorname{tr}_b^a\left[\mathbf{M}\right]\mathbf{Y}^T$$

□

Finally, we need a new property of the stridewise-blockwise trace, the 'Downsampling Property'. This is essential to be able to work with rectangular matrices in the stridewise-blockwise trace, such as those arising from partial eigendecomposition.

Lemma 15 (Downsampling Property)**.** *Suppose* $\mathbf{U}_{a\times x}$ *and* $\mathbf{V}_{a\times x}$ *are matrices such that* $\mathbf{V}^T\mathbf{U} = \mathbf{I}_{x\times x}$*, and* $\mathbf{W}_{b\times y}$ *and* $\mathbf{X}_{b\times y}$ *are matrices such that* $\mathbf{X}^T\mathbf{W} = \mathbf{I}_{y\times y}$*. Then we have the following.*

$$\mathrm{tr}_b^a\left[(\mathbf{U}\otimes\mathbf{I}\otimes\mathbf{W})\,\mathbf{M}\,(\mathbf{V}\otimes\mathbf{I}\otimes\mathbf{X})^T\right] = \mathrm{tr}_y^x\left[\mathbf{M}\right]$$

Proof.

$$\begin{aligned}
\mathrm{tr}_b^a\left[(\mathbf{U}\otimes\mathbf{I}\otimes\mathbf{W})\,\mathbf{M}\,(\mathbf{V}\otimes\mathbf{I}\otimes\mathbf{X})^T\right] &= \mathrm{tr}\left[(\mathbf{U}\otimes\mathbf{I}\otimes\mathbf{W})\,\mathbf{M}\,(\mathbf{V}\otimes\mathbf{I}\otimes\mathbf{X})^T\left(\mathbf{I}\otimes\mathbf{J}^{ij}\otimes\mathbf{I}\right)\right]_{ij}\\
&= \mathrm{tr}\left[\mathbf{M}\left(\mathbf{V}^T\mathbf{U}\otimes\mathbf{J}^{ij}\otimes\mathbf{X}^T\mathbf{W}\right)\right]_{ij}\\
&= \mathrm{tr}\left[\mathbf{M}\left(\mathbf{I}_{x\times x}\otimes\mathbf{J}^{ij}\otimes\mathbf{I}_{y\times y}\right)\right]_{ij}\\
&= \mathrm{tr}_y^x\left[\mathbf{M}\right]
\end{aligned}$$

□

E.7 Finding the maximum likelihood estimator

In this section, we will derive gradients for gradient descent and prove Theorem 10. For convenience, we repeat the definition of the NLL.

$$\mathrm{NLL} = \frac{-1}{2}\sum_\gamma\left(\log\det\left[\mathbf{t}^T\mathbf{a}^\gamma\mathbf{I}_{d_\forall^\gamma} + \bigoplus_{\ell\in\gamma}\mathbf{\Psi}_\ell\right] - \sum_{\ell\in\gamma}\mathrm{tr}\left[\mathbf{S}_\ell^\gamma\mathbf{\Psi}_\ell\right] - \sum_{\ell\in\gamma}\frac{\mathbf{t}^T\mathbf{a}^\gamma}{L^\gamma}\mathrm{tr}\left[\mathbf{S}_\ell^\gamma\right]\right)$$

To derive the MLE eigenvectors, we proceed in a manner similar to Andrew et al. (2024). A key difference is that we must now differentiate a log-pseudodeterminant, although it behaves analagously to the derivative of a full determinant. This is given by Theorem 2.15 of Holbrook (2018):

$$\frac{\partial}{\partial\mathbf{A}}\mathrm{logdet}^\dagger(\mathbf{A}) = \mathrm{tr}\left[\mathbf{A}^\dagger\right]$$

Lemma 16. *At the maximum likelihood of the singular Kronecker-sum-structured normal distribution, we have that* $\mathbf{S}_\ell = \sum_{\gamma|\ell\in\gamma}\mathrm{tr}_{d_{>\ell}^\gamma}^{d_{<\ell}^\gamma}\left[\left(\bigoplus_{\ell'\in\gamma}\mathbf{\Psi}_{\ell'}\right)^\dagger\right]$.

Proof. Note that $\mathbf{\Psi}_\ell$ is symmetric, and hence $\frac{\partial f(\mathbf{\Psi})}{\partial\mathbf{\Psi}} = \mathrm{sym}\left[\mathbf{X}\right] = \frac{\mathbf{X}+\mathbf{X}^T}{2}$, where $\mathbf{X}$ is the derivative found without taking into account symmetry (Srinivasan & Panda, 2023). It does not affect the location of the MLE (although we include it in the derivations for completeness), but it does affect the value of the gradient, and thus will be important to account for when the gradient itself is of interest.

$$\begin{aligned}
\frac{\partial}{\partial\mathbf{\Psi}_\ell}\mathrm{NLL} &= \mathrm{sym}\left[\frac{\partial}{\partial\mathbf{\Psi}_\ell}\sum_{\ell'}\frac{1}{2}\mathrm{tr}\left[\mathbf{S}_{\ell'}\mathbf{\Psi}_{\ell'}\right] - \sum_\gamma\left(\frac{\partial}{\partial\mathbf{\Psi}_{\ell_{ij}}}\frac{1}{2}\mathrm{logdet}^\dagger\left[\bigoplus_{\ell'\in\gamma}\mathbf{\Psi}_{\ell'}\right]_{ij}\right)\right]\\
&= \mathrm{sym}\left[\sum_\gamma\frac{1}{2}\mathbf{S}_\ell^\gamma - \frac{1}{2}\sum_{\gamma|\ell\in\gamma}\mathrm{tr}\left[\left(\bigoplus_{\ell'\in\gamma}\mathbf{\Psi}_{\ell'}\right)^\dagger\frac{\partial}{\partial\mathbf{\Psi}_{\ell_{ij}}}\bigoplus_{\ell'\in\gamma}\mathbf{\Psi}_{\ell'}\right]_{ij}\right]
\end{aligned}$$

It is also not too hard to show that $\frac{\partial}{\partial \boldsymbol{\Psi}_{\ell_{ij}}} \bigoplus_{\ell' \in \gamma} \boldsymbol{\Psi}_{\ell'} = \mathbf{I}_{d^\gamma_{<\ell}} \otimes \mathbf{J}^{ij}_{d_\ell \times d_\ell} \otimes \mathbf{I}_{d^\gamma_{>\ell}}$ (not accounting for symmetry).

$$\begin{aligned}
\frac{\partial}{\partial \boldsymbol{\Psi}_{\ell_{ij}}} \bigoplus_{\ell' \in \gamma} \boldsymbol{\Psi}_{\ell'} &= \frac{\partial}{\partial \boldsymbol{\Psi}_{\ell_{ij}}} \sum_{\ell' \in \gamma} \mathbf{I}_{d^\gamma_{<\ell'}} \otimes \boldsymbol{\Psi}_{\ell'} \otimes \mathbf{I}_{d^\gamma_{>\ell'}} \\
&= \mathbf{I}_{d^\gamma_{<\ell}} \otimes \frac{\partial}{\partial \boldsymbol{\Psi}_{\ell_{ij}}} \boldsymbol{\Psi}_{\ell} \otimes \mathbf{I}_{d^\gamma_{>\ell}} \\
&= \mathbf{I}_{d^\gamma_{<\ell}} \otimes \mathbf{J}^{ij}_{d_\ell \times d_\ell} \otimes \mathbf{I}_{d^\gamma_{>\ell}}
\end{aligned}$$

By the definition of the stridewise-blockwise trace, we have that:

$$\frac{\partial}{\partial \boldsymbol{\Psi}_\ell} \mathrm{NLL} = \mathrm{sym}\left[\sum_\gamma \frac{1}{2} \mathbf{S}^\gamma_\ell - \frac{1}{2} \sum_{\gamma | \ell \in \gamma} \mathrm{tr}^{d^\gamma_{<\ell}}_{d^\gamma_{>\ell}} \left[\left(\bigoplus_{\ell' \in \gamma} \boldsymbol{\Psi}_{\ell'} \right)^\dagger \right] \right]$$

Now, note that the maximum likelihood corresponds to the point where $\frac{\partial}{\partial \boldsymbol{\Psi}_\ell} \mathrm{NLL}$ is 0.

$$\mathbf{S}_\ell = \sum_{\gamma | \ell \in \gamma} \mathrm{tr}^{d^\gamma_{<\ell}}_{d^\gamma_{>\ell}} \left[\left(\bigoplus_{\ell' \in \gamma} \boldsymbol{\Psi}_{\ell'} \right)^\dagger \right]$$

□

Theorem (Theorem 10)**.** *Let the rank-r_ℓ partial eigendecomposition of $\mathbf{S}_\ell$ be $\mathbf{V}^{(k)}_\ell \mathbf{E}^{(r_\ell)}_\ell \mathbf{V}^{(r_\ell),T}_\ell$, with $r_\ell \geq \max\left(\mathrm{rank}\left[\mathbf{S}_\ell\right], \mathrm{rank}\left[\boldsymbol{\Psi}_\ell\right]\right)$. Then $\mathbf{V}^{(r_\ell)}_\ell$ are also eigenvectors of $\boldsymbol{\Psi}_\ell$.*

Proof. Suppose we had a rank-r_ℓ partial eigendecomposition yielding a $d_\ell \times r_\ell$ matrix of eigenvectors $\mathbf{V}^{(r_\ell)}_\ell$; in this case, $\mathbf{V}^{(r_\ell),T}_\ell \mathbf{V}^{(r_\ell)}_\ell$ is still the identity matrix (albeit of a smaller dimension); this will allow us to use Lemma 15 to 'downsample' the stridewise-blockwise trace.

We will use $\mathbf{V}^\gamma_{<\ell}$ as a shorthand for $\bigotimes_{\ell' \in \gamma | \ell' <^\gamma \ell} \mathbf{V}^{(r'_\ell)}_{\ell'}$ (with $\mathbf{V}^\gamma_{>\ell}$ defined analogously).

With this in mind, note that by Lemma 16 we have that:

$$\begin{aligned}
\mathbf{S}_\ell &= \sum_{\gamma | \ell \in \gamma} \mathrm{tr}^{d^\gamma_{<\ell}}_{d^\gamma_{>\ell}} \left[\left(\bigoplus_{\ell' \in \gamma} \boldsymbol{\Psi}_{\ell'} \right)^\dagger \right] \\
&= \sum_{\gamma | \ell \in \gamma} \mathrm{tr}^{d_{<\ell}}_{d^\gamma_{>\ell}} \left[\left(\mathbf{V}^\gamma_{<\ell} \otimes \mathbf{V}^{(r_\ell)}_\ell \otimes \mathbf{V}^\gamma_{>\ell} \right) \left(\bigoplus_{\ell' \in \gamma} \boldsymbol{\Lambda}^{(r_{\ell'})}_{\ell'} \right)^\dagger \left(\mathbf{V}^\gamma_{<\ell} \otimes \mathbf{V}^{(r_\ell)}_\ell \otimes \mathbf{V}^\gamma_{>\ell} \right)^T \right] \\
&= \mathbf{V}^{(r_\ell)}_\ell \sum_{\gamma | \ell \in \gamma} \mathrm{tr}^{k^\gamma_{<\ell}}_{k^\gamma_{>\ell}} \left[\left(\bigoplus_{\ell' \in \gamma} \boldsymbol{\Lambda}^{(r_{\ell'})}_{\ell'} \right)^\dagger \right] \mathbf{V}^{(r_\ell),T}_\ell \qquad \text{(Extraction and Downsampling Properties.)}
\end{aligned}$$

The $\mathbf{V}^\gamma_{<\ell}$ and $\mathbf{V}^\gamma_{>\ell}$ matrices depend on γ as the precise tensor they are in affects which ℓs are present and in what order. Thankfully, they disappear due to the downsampling property of sb-trace, allowing us to express our equation as the product of orthogonal matrices $\mathbf{V}^{(r_\ell)}_\ell, \mathbf{V}^{(r_\ell),T}_\ell$ and a diagonal matrix $\sum_{\gamma | \ell \in \gamma} \mathrm{tr}^{k^\gamma_{<\ell}}_{k^\gamma_{>\ell}} \left[\left(\bigoplus_{\ell' \in \gamma} \boldsymbol{\Lambda}^{(r_{\ell'})}_{\ell'} \right)^\dagger \right]$. Thus, this is clearly a partial eigendecomposition of $\mathbf{S}_\ell$, completing the proof. □

Theorem 10 establishes that we can solve for the eigenvectors of each $\mathbf{\Psi}_\ell$ analytically. No such expression for the eigenvalues exist, but it is not too hard to find them with gradient descent. The proportion of time this estimation takes is negligible, due to cost of partial eigendecomposition and the comparatively small dimension of the eigenvalues ($O(\sum_\ell d_\ell)$ parameters rather than $O(\sum_\ell d_\ell^2)$). We derive the gradient below:

Proposition. *The gradient of the* NLL *with respect to the eigenvalues is given by* $\frac{1}{2}\left(\mathbf{E}_\ell^{(r_\ell)} - \sum_{\gamma|\ell\in\gamma} \mathrm{tr}_{k^\gamma_{>\ell}}^{k^\gamma_{<\ell}}\left[\left(\bigoplus_{\ell'\in\gamma}\mathbf{\Lambda}_{\ell'}^{(r_{\ell'})}\right)^\dagger\right]\right)$.

Proof. As mentioned earlier, we will use the functional invariance of the MLE to change the problem from finding when $\frac{\partial}{\partial \mathbf{\Psi}_\ell}\mathrm{NLL} = 0$ to one of finding when:

$$\frac{\partial}{\partial \mathbf{V}_\ell^{(r_\ell)}}\mathrm{NLL} = 0$$
$$\frac{\partial}{\partial \mathbf{\Lambda}_\ell^{(r_\ell)}}\mathrm{NLL} = 0$$

We already know the conditions on $\mathbf{V}_\ell^{(r_\ell)}$ at the MLE, thanks to Theorem 10, so we will focus on $\mathbf{\Lambda}_\ell^{(r_\ell)}$. We follow a similar derivation to Lemma 16.

$$\begin{aligned}
\frac{\partial}{\partial \mathbf{\Lambda}_\ell}\mathrm{NLL} &= \frac{\partial}{\partial \mathbf{\Lambda}_\ell^{(r_\ell)}}\sum_{\ell'}\frac{1}{2}\mathrm{tr}\left[\mathbf{S}_{\ell'}\mathbf{\Psi}_{\ell'}\right] - \sum_\gamma\left(\frac{\partial}{\partial \mathbf{\Lambda}_{\ell_{ii}}^{(r_\ell)}}\frac{1}{2}\mathrm{logdet}^\dagger\left[\bigoplus_{\ell'\in\gamma}\mathbf{\Psi}_{\ell'}\right]_{ii}\right)\\
&= \frac{\partial}{\partial \mathbf{\Lambda}_\ell^{(r_\ell)}}\sum_{\ell'}\frac{1}{2}\mathrm{tr}\left[\mathbf{E}_{\ell'}^{(r_{\ell'})}\mathbf{\Lambda}_{\ell'}^{(r_{\ell'})}\right] - \sum_\gamma\left(\frac{\partial}{\partial \mathbf{\Lambda}_{\ell_{ii}}^{(r_\ell)}}\frac{1}{2}\mathrm{logdet}^\dagger\left[\bigoplus_{\ell'\in\gamma}\mathbf{\Lambda}_{\ell'}^{(r_{\ell'})}\right]_{ii}\right)\\
&= \sum_\gamma\frac{1}{2}\mathbf{E}_\ell^{(r_\ell),\gamma} - \frac{1}{2}\sum_{\gamma|\ell\in\gamma}\mathrm{tr}\left[\left(\bigoplus_{\ell'\in\gamma}\mathbf{\Lambda}_{\ell'}^{(r_{\ell'})}\right)^\dagger\frac{\partial}{\partial \mathbf{\Lambda}_{\ell_{ii}}^{(r_\ell)}}\bigoplus_{\ell'\in\gamma}\mathbf{\Lambda}_{\ell'}^{(r_{\ell'})}\right]_{ii}\\
&= \frac{1}{2}\left(\mathbf{E}_\ell^{(r_\ell)} - \sum_{\gamma|\ell\in\gamma}\mathrm{tr}_{k^\gamma_{>\ell}}^{k^\gamma_{<\ell}}\left[\left(\bigoplus_{\ell'\in\gamma}\mathbf{\Lambda}_{\ell'}^{(r_{\ell'})}\right)^\dagger\right]\right)
\end{aligned}$$

In the derivation, we made use of the fact that $\mathbf{S}_\ell$ and $\mathbf{\Psi}_\ell$ had the same eigenvectors at the MLE, the cyclic property of the trace, and the fact that pseudodeterminants only depend on the nonzero eigenvalues. □

E.8 Existence and uniqueness

It is easy to see that our objective is convex; unstructured Gaussian precision matrix estimation is strictly convex, and the space of Kronecker-sum-decomposable matrices is also convex. In this section, we will further prove the existence and uniqueness of solutions (Theorem 11).

Lemma 17 (Existence)**.** *There exists a minimizer of the negative log likelihood in the domain* $\mathbf{\Lambda}_\ell^{(r_\ell)} \geq \epsilon_\ell$ *(Assumption 10), where* ϵ_ℓ *is some strictly positive constant (for each* ℓ*), as long as* $\mathbf{E}_\ell^{(r_\ell)} \in \mathbb{R}_{r_\ell}^{++}$ *(i.e.* $r_\ell \leq \mathrm{rank}\left[\mathbf{S}_\ell\right]$*).*

Proof. First, note that $\bigtimes_\ell \bigtimes_{1\leq i\leq r_\ell}[\epsilon_\ell, \infty]$ is a compact subset of the extended reals, and thus a minimum to any continuous function must exist on this set. As it is convex, it will be a global minimum. However, the minimum could exist at ∞, i.e., only in the limit. To rule out this case, we will establish an upper bound on the solution.

Recall that any minimum must satisfy $\mathbf{E}_\ell^{(r_\ell)} = \sum_{\gamma|\ell\in\gamma} \mathrm{tr}_{k^\gamma_{>\ell}}^{k^\gamma_{<\ell}} \left[\left(\oplus_{\ell'\in\gamma}\mathbf{\Lambda}_{\ell'}^{(r_{\ell'})}\right)^\dagger\right]$ (for each ℓ). It will be helpful to see what this formula equates to elementwise:

$$\begin{aligned}
\mathbf{E}_{\ell_{ii}}^{(r_\ell)} &= \sum_{\gamma|\ell\in\gamma} \mathrm{tr}_{d^\gamma_{>\ell}}^{d^\gamma_{<\ell}} \left[\left(\oplus_{\ell'\in\gamma}\mathbf{\Lambda}_{\ell'}^{(r_{\ell'})}\right)^\dagger\right]_{ii} \\
&= \sum_{\gamma|\ell\in\gamma} \mathrm{tr}\left[\left(\oplus_{\ell'\in\gamma}\mathbf{\Lambda}_{\ell'}^{(r_{\ell'})}\right)^\dagger \left(\mathbf{I}_{r^\gamma_{<\ell}} \otimes \mathbf{J}^{ii} \otimes \mathbf{I}_{r^\gamma_{>\ell}}\right)\right] \\
&= \sum_{\gamma|\ell\in\gamma} \sum_{\mathbf{j} \text{ indexing all non-}\ell \text{ axes}} \frac{1}{\lambda_{\ell_i} + \sum_{\ell'} \lambda_{\ell'_{j_{\ell'}}}}
\end{aligned}$$

Thus, every element of $\mathbf{E}_\ell^{(r_\ell)}$ has a corresponding element of $\mathbf{\Lambda}_\ell^{(r_\ell)}$ that appears in all summands composing it, and vice versa. Suppose $\lambda_{\ell_i} = \frac{\sum_\gamma d^\gamma_{\setminus\ell}}{\mathbf{E}_{\ell_{ii}}}$ (for a given ℓ, and every i). Note that if for all other ℓ', $\lambda_{\ell'_{j_{\ell'}}}$ were 0, then the sum of these λ_ℓ terms would sum to exactly $\mathbf{E}_{\ell_{ii}}$. However, as $\mathbf{\Lambda}_\ell^{(r_\ell)} \geq \epsilon_\ell$, each reciprocal is smaller than the case where they are 0; thus, they cannot add up to $\mathbf{E}_{\ell_{ii}}$.

As λ_{ℓ_i} increases above this bound, the problem only becomes more severe. This is easy to see by checking the gradient, which indicates that the negative log-likelihood increases as one increases λ_{ℓ_i}.

$$\begin{aligned}
\frac{\partial}{\partial \mathbf{\Lambda}_{\ell_{ii}}^{(r_\ell)}} \mathrm{NLL} &= \frac{1}{2}\left(\mathbf{E}_{\ell_{ii}} - \sum_{\gamma|\ell\in\gamma} \sum_{\mathbf{j} \text{ indexing all non-}\ell \text{ axes}} \frac{1}{\lambda_{\ell_i} + \sum_{\ell'} \lambda_{\ell'_{j_{\ell'}}}}\right) \\
&> \frac{1}{2}\left(\mathbf{E}_{\ell_{ii}} - \mathbf{E}_{\ell_{ii}}\right) \\
&= 0
\end{aligned}$$

This implies that the minimum must occur between $\frac{\sum_\gamma d^\gamma_{\setminus\ell}}{\mathbf{E}_{\ell_{ii}}} \geq \lambda_{\ell_{ii}} \geq \epsilon_\ell$. Clearly, this rules out the unsavory case of the minimum occurring in the limit. This completes the proof. □

We already saw in Section E.3 that the diagonals of $\mathbf{\Psi}_\ell$ are not identifiable. However, this non-identifiability is only an effect of our parameterization of the full precision matrices $\{\mathbf{\Omega}^\gamma\}$; it is reasonable to ask if the solution is unique up to this non-identifiability; Lemma 18 shows that this is true.

Lemma 18 (Uniqueness up-to Identifiability)**.** *The solution of our optimization problem is unique, except for the non-identifiability in the diagonals.*

Proof. As mentioned earlier, we can frame our optimization problem as the strictly convex problem of unstructured precision matrix estimation (as in the graphical lasso problem), subject to a (convex) Kronecker-sum-decomposability constraint. As the problem is strictly convex, there is a unique solution. Thus, any non-uniqueness must come from our parameterization of Kronecker-sum-decomposable matrices. □

We now tie these two lemmas together to complete the statement of Theorem 11.

Theorem (Theorem 11)**.** *There exists a unique MLE (up to the non-identifiability of the diagonals of the Kronecker sum) that estimates the precision matrix of the singular Kronecker-sum-structured normal distribution, as long as* $r_\ell \leq \mathrm{rank}\,[\mathbf{S}_\ell]$*, subject to Assumption 10.*

Proof. This follows directly from Lemmas 17 and 18. □

E.9 Derivation of the Fisher information

This section contains the derivation of the Fisher Information, which is far too long and tedious to put in the main paper. For convenience, we repeat the definition of the NLL.

$$\text{NLL} = \frac{-1}{2}\sum_{\gamma}\left(\log\det\left[\mathbf{t}^T\mathbf{a}^\gamma\mathbf{I}_{d_\forall^\gamma} + \bigoplus_{\ell\in\gamma}\mathbf{\Psi}_\ell\right] - \sum_{\ell\in\gamma}\text{tr}\left[\mathbf{S}_\ell^\gamma\mathbf{\Psi}_\ell\right] - \sum_{\ell\in\gamma}\frac{\mathbf{t}^T\mathbf{a}^\gamma}{L^\gamma}\text{tr}\left[\mathbf{S}_\ell^\gamma\right]\right)$$

Lemmas 19, 20, and 21 contain the calculations that derive the Hessian of the NLL.

Lemma 19. $\frac{\partial}{\partial\psi_{\ell_{ij}}}\text{NLL} = -\frac{1}{2}\text{sym}\left[\sum_{\gamma|\ell\in\gamma}\text{tr}_{d_{>\ell}^\gamma}^{d_{<\ell}^\gamma}\left[\left(\mathbf{t}^T\mathbf{a}^\gamma\mathbf{I}_{\mathbf{d}_\forall^\gamma} + \bigoplus_{\ell\in\gamma}\mathbf{\Psi}_\ell\right)^{-1}\right]_{ij} - \mathbf{S}_{\ell_{ij}}\right]$

Proof. This follows from Lemma 16. □

Lemma 20.

$$\frac{\partial}{\partial t_i}\text{NLL} = \sum_{\gamma} -\frac{a_i^\gamma}{2}\text{tr}\left[\left(\mathbf{t}^T\mathbf{a}^\gamma\mathbf{I}_{d_\forall^\gamma} + \bigoplus_{\ell\in\gamma}\mathbf{\Psi}_\ell\right)^{-1}\right] + \frac{a_i^\gamma}{2}\text{tr}\left[\mathbf{S}^\gamma\right]$$

$$\frac{\partial}{\partial t_i\partial t_j}\text{NLL} = \sum_{\gamma}\frac{a_i^\gamma a_j^\gamma}{2}\text{tr}\left[\left(\mathbf{t}^T\mathbf{a}^\gamma\mathbf{I}_{d_\forall^\gamma} + \bigoplus_{\ell\in\gamma}\mathbf{\Psi}_\ell\right)^{-2}\right]$$

$$\frac{\partial}{\partial t_i\partial\psi_{\ell_{jk}}}\text{NLL} = \text{sym}\left[\sum_{\gamma|\ell\in\gamma}\frac{a_i^\gamma}{2}\text{tr}_{d_{>\ell}^\gamma}^{d_{<\ell}^\gamma}\left[\left(\mathbf{t}^T\mathbf{a}^\gamma\mathbf{I}_{d_\forall^\gamma} + \bigoplus_{\ell\in\gamma}\mathbf{\Psi}_\ell\right)^{-2}\right]\right]_{jk}$$

Proof. First, observe the following:

$$\begin{aligned}\frac{\partial}{\partial t_i}\log\det\left[\mathbf{t}^T\mathbf{a}^\gamma\mathbf{I}_{d_\forall^\gamma} + \bigoplus_{\ell\in\gamma}\mathbf{\Psi}_\ell\right] &= \text{tr}\left[\left(\mathbf{t}^T\mathbf{a}^\gamma\mathbf{I}_{d_\forall^\gamma} + \bigoplus_{\ell\in\gamma}\mathbf{\Psi}_\ell\right)^{-1}\frac{\partial}{\partial t_i}\left(\mathbf{t}^T\mathbf{a}^\gamma\mathbf{I}_{d_\forall^\gamma} + \bigoplus_{\ell\in\gamma}\mathbf{\Psi}_\ell\right)\right]\\ &= \text{tr}\left[\left(\mathbf{t}^T\mathbf{a}^\gamma\mathbf{I}_{d_\forall^\gamma} + \bigoplus_{\ell\in\gamma}\mathbf{\Psi}_\ell\right)^{-1}a_i^\gamma\mathbf{I}_{d_\forall^\gamma}\right]\\ &= a_i^\gamma\text{tr}\left[\left(\mathbf{t}^T\mathbf{a}^\gamma\mathbf{I}_{d_\forall^\gamma} + \bigoplus_{\ell\in\gamma}\mathbf{\Psi}_\ell\right)^{-1}\right]\end{aligned}$$

Furthermore, note that:

$$\begin{aligned}\frac{\partial}{\partial t_i}\sum_{\ell\in\gamma}\frac{\mathbf{t}^T\mathbf{a}^\gamma}{L^\gamma}\text{tr}\left[\mathbf{S}_\ell^\gamma\right] &= \sum_{\ell\in\gamma}\frac{a_i^\gamma}{L^\gamma}\text{tr}\left[\mathbf{S}_\ell^\gamma\right]\\ &= \frac{a_i^\gamma}{L^\gamma}\text{tr}\left[\mathbf{S}^\gamma\right]\end{aligned}$$

Putting these together yields the first claim:

$$\frac{\partial}{\partial t_i}\text{NLL} = \sum_{\gamma} -\frac{a_i^\gamma}{2}\text{tr}\left[\left(\mathbf{t}^T\mathbf{a}^\gamma\mathbf{I}_{d_\forall^\gamma} + \bigoplus_{\ell\in\gamma}\mathbf{\Psi}_\ell\right)^{-1}\right] + \frac{a_i^\gamma}{2}\text{tr}\left[\mathbf{S}^\gamma\right]$$

Differentiating a second time results in the following:

$$\frac{\partial}{\partial t_i \partial t_j}\text{NLL} = \sum_\gamma \frac{a_i^\gamma}{2}\text{tr}\left[\left(\mathbf{t}^T\mathbf{a}^\gamma\mathbf{I}_{d_\forall^\gamma} + \bigoplus_{\ell\in\gamma}\mathbf{\Psi}_\ell\right)^{-1}\left(a_i^\gamma\mathbf{I}_{d_\forall^\gamma}\right)\left(\mathbf{t}^T\mathbf{a}^\gamma\mathbf{I}_{d_\forall^\gamma} + \bigoplus_{\ell\in\gamma}\mathbf{\Psi}_\ell\right)^{-1}\right]$$
$$= \sum_\gamma \frac{a_i^\gamma a_j^\gamma}{2}\text{tr}\left[\left(\mathbf{t}^T\mathbf{a}^\gamma\mathbf{I}_{d_\forall^\gamma} + \bigoplus_{\ell\in\gamma}\mathbf{\Psi}_\ell\right)^{-2}\right]$$

Likewise, differentiating w.r.t. $\psi_{\ell_{jk}}$ gives us the final part of our result. We'll delay the symmetrization step until the end to prevent the equation running off the right margin of the page.

$$\frac{\partial^{\text{nosym}}}{\partial t_i \partial \psi_{\ell_{jk}}}\text{NLL} = \sum_{\gamma|\ell\in\gamma} \frac{a_i^\gamma}{2}\text{tr}\left[\left(\mathbf{t}^T\mathbf{a}^\gamma\mathbf{I}_{d_\forall^\gamma} + \bigoplus_{\ell\in\gamma}\mathbf{\Psi}_\ell\right)^{-1}\left(\mathbf{I}_{d_{<\ell}^\gamma}\otimes\mathbf{J}^{jk}\otimes\mathbf{I}_{d_{>\ell}^\gamma}\right)\left(\mathbf{t}^T\mathbf{a}^\gamma\mathbf{I}_{d_\forall^\gamma} + \bigoplus_{\ell\in\gamma}\mathbf{\Psi}_\ell\right)^{-1}\right]$$
$$= \sum_{\gamma|\ell\in\gamma} \frac{a_i^\gamma}{2}\text{tr}_{d_{>\ell}^\gamma}^{d_{<\ell}^\gamma}\left[\left(\mathbf{t}^T\mathbf{a}^\gamma\mathbf{I}_{d_\forall^\gamma} + \bigoplus_{\ell\in\gamma}\mathbf{\Psi}_\ell\right)^{-2}\right]_{jk}$$
$$\frac{\partial}{\partial t_i \partial \psi_{\ell_{jk}}}\text{NLL} = \text{sym}\left[\sum_{\gamma|\ell\in\gamma} \frac{a_i^\gamma}{2}\text{tr}_{d_{>\ell}^\gamma}^{d_{<\ell}^\gamma}\left[\left(\mathbf{t}^T\mathbf{a}^\gamma\mathbf{I}_{d_\forall^\gamma} + \bigoplus_{\ell\in\gamma}\mathbf{\Psi}_\ell\right)^{-2}\right]\right]_{jk}$$

This completes the proof. □

Lemma 21.

$$\frac{\partial^{\text{nosym}}}{\partial\psi_{\ell^1_{i^1j^1}}\partial\psi_{\ell^2_{i^2j^2}}}\text{NLL} = \frac{1}{2}\sum_{\gamma|\ell^1,\ell^2\in\gamma}\text{tr}_{d_{>\ell^1}^\gamma}^{d_{<\ell^1}^\gamma}\left[(\mathbf{\Omega}^\gamma)^{-1}\left(\mathbf{I}_{d_{<\ell^2}^\gamma}\otimes\mathbf{J}^{i^2j^2}\otimes\mathbf{I}_{d_{>\ell^2}^\gamma}\right)(\mathbf{\Omega}^\gamma)^{-1}\right]_{i^1j^1}$$

Proof. This follows from Lemma 19. First we will compute the unsymmetrized derivative.

$$\frac{\partial^{\text{nosym}}}{\partial\psi_{\ell^1_{i^1j^1}}\partial\psi_{\ell^2_{i^2j^2}}}\text{NLL} = -\frac{1}{2}\frac{\partial}{\partial\psi_{\ell^2_{i^2j^2}}}\sum_{\gamma|\ell^1\in\gamma}\text{tr}_{d_{>\ell^1}^\gamma}^{d_{<\ell^1}^\gamma}\left[\left(\mathbf{t}^T\mathbf{a}^\gamma\mathbf{I}_{\mathbf{d}_\forall^\gamma} + \bigoplus_{\ell\in\gamma}\mathbf{\Psi}_{\ell^1}\right)^{-1}\right]_{i^1j^1}$$
$$= -\frac{1}{2}\frac{\partial}{\partial\psi_{\ell^2_{i^2j^2}}}\sum_{\gamma|\ell^1\in\gamma}\text{tr}_{d_{>\ell^1}^\gamma}^{d_{<\ell^1}^\gamma}\left[(\mathbf{\Omega}^\gamma)^{-1}\right]_{i^1j^1}$$
$$= \frac{1}{2}\sum_{\gamma|\ell^1,\ell^2\in\gamma}\text{tr}_{d_{>\ell^1}^\gamma}^{d_{<\ell^1}^\gamma}\left[(\mathbf{\Omega}^\gamma)^{-1}\left(\mathbf{I}_{d_{<\ell^2}^\gamma}\otimes\mathbf{J}^{i^2j^2}\otimes\mathbf{I}_{d_{>\ell^2}^\gamma}\right)(\mathbf{\Omega}^\gamma)^{-1}\right]_{i^1j^1}$$

If we let the unsymmetrized derivative be $\mathbf{X}_{\ell^1_{i^1j^1}\ell^2_{i^2j^2}}$, the symmetrized version is:

$$\frac{\mathbf{X}_{\ell^1_{i^1j^1}\ell^2_{i^2j^2}} + \mathbf{X}_{\ell^1_{j^1i^1}\ell^2_{i^2j^2}} + \mathbf{X}_{\ell^1_{i^1j^1}\ell^2_{j^2i^2}} + \mathbf{X}_{\ell^1_{j^1i^1}\ell^2_{j^2i^2}}}{4}$$

□

With these, we can derive $\mathbf{F}$. As the derivation of inter-axis connections in $\mathbf{F}$ is complicated, and will be aided by the following lemma, which is just an application of the fact that the trace distributes over Kronecker products.

Lemma 22. *Let $\{\mathbf{M}_\ell\}_\ell$ be a subset of matrices where some subset X has the property that, for all $\ell \in X$, $\mathbf{M}_\ell = \mathbf{I}_{d_\ell \times d_\ell}$, where d_ℓ represents the size of the ℓth element, and, for all $\ell \notin X$, $\mathbf{M}_\ell = \mathbf{J}^{i_\ell j_\ell}_{d_\ell \times d_\ell}$ (the matrix of zeros except at with a 1 at position (i_ℓ, j_ℓ)).*

Then $\operatorname{tr}\left[\bigotimes_\ell \mathbf{M}_\ell\right] = \left(\prod_{\ell \in X} d_\ell\right)\left(\prod_{\ell \notin X} \delta_{i_\ell j_\ell}\right)$.

Proof.

$$\begin{aligned}
\operatorname{tr}\left[\bigotimes_\ell \mathbf{M}_\ell\right] &= \prod_\ell \operatorname{tr}\left[\mathbf{M}_\ell\right] \\
&= \prod_{\ell \in X} \operatorname{tr}\left[\mathbf{M}_\ell\right] \prod_{\ell \notin X} \operatorname{tr}\left[\mathbf{M}_\ell\right] \\
&= \left(\prod_{\ell \in X} d_\ell\right) \prod_{\ell \notin X} \operatorname{tr}\left[\mathbf{M}_\ell\right] \\
&= \left(\prod_{\ell \in X} d_\ell\right)\left(\prod_{\ell \notin X} \delta_{i_\ell j_\ell}\right)
\end{aligned}$$

□

We are now equipped with all the tools we need to prove Lemma 23, the values of $\mathbf{F}$ under the null hypothesis.

Lemma 23. *Under the null hypothesis where $\mathbf{\Omega}^\gamma = \frac{1}{(\sigma^\gamma)^2}\mathbf{I}_{d^\gamma_\forall}$, we have the following:*

$$\begin{aligned}
\mathbf{F}_{t_i t_j} &= \sum_\gamma \frac{d^\gamma_\forall \left(\sigma^\gamma\right)^4}{2}\left(\mathbf{a}^\gamma \mathbf{a}^{\gamma,T}\right)_{ij} \\
\mathbf{F}_{t_i \psi_{\ell_{jk}}} &= \sum_{\gamma | \ell \in \gamma} \frac{a^\gamma_i d^\gamma_\forall \left(\sigma^\gamma\right)^4}{2 d_\ell}\delta_{jk} \\
\mathbf{F}_{\psi_{\ell^1_{i^1 j^1}} \psi_{\ell^2_{i^2 j^2}}} &= \frac{1}{2}\sum_{\gamma | \ell^1, \ell^2 \in \gamma} \left(\sigma^\gamma\right)^4 \frac{d^\gamma_\forall}{d_{\ell^1} d_{\ell^2}}\delta_{i^2 j^2}\delta_{i^1 j^1} && (\ell^1 \neq \ell^2) \\
\mathbf{F}_{\psi_{\ell^1_{i^1 j^1}} \psi_{\ell^2_{i^2 j^2}}} &= \frac{1}{2}\sum_{\gamma | \ell^1, \ell^2 \in \gamma} \left(\sigma^\gamma\right)^4 \frac{d^\gamma_\forall}{d_\ell}\left(\delta_{i^1 j^1 i^2 j^2} + (1 - \delta_{i^1 j^1 i^2 j^2})\frac{\delta_{i^1 i^2}\delta_{j^1 j^2}}{2}\right) && (\ell^1 = \ell^2)
\end{aligned}$$

Proof.

$$\begin{aligned}
\frac{\partial}{\partial t_i \partial t_j}\text{NLL} &= \sum_\gamma \frac{a_i^\gamma a_j^\gamma}{2}\text{tr}\left[(\mathbf{\Omega}^\gamma)^{-2}\right] \\
&= \sum_\gamma \frac{a_i^\gamma a_j^\gamma d_\forall^\gamma (\sigma^\gamma)^4}{2} \\
&= \sum_\gamma \frac{d_\forall^\gamma (\sigma^\gamma)^4}{2}\left(\mathbf{a}^\gamma \mathbf{a}^{\gamma,T}\right)_{ij} \\
\frac{\partial}{\partial t_i \partial \psi_{\ell_{jk}}}\text{NLL} &= \text{sym}\left[\sum_\gamma \frac{a_i^\gamma}{2}\text{tr}_{d_{>\ell}^\gamma}^{d_{<\ell}^\gamma}\left[(\mathbf{\Omega}^\gamma)^{-2}\right]_{jk}\right] \\
&= \text{sym}\left[\sum_\gamma \frac{a_i^\gamma d_\forall^\gamma (\sigma^\gamma)^4}{2d_\ell}\left[\mathbf{I}_{d_{\setminus\ell}^\gamma}\right]_{jk}\right] \\
&= \frac{1}{2}\sum_\gamma \frac{a_i^\gamma d_\forall^\gamma (\sigma^\gamma)^4}{2d_\ell}\delta_{jk} + \frac{a_i^\gamma d_\forall^\gamma (\sigma^\gamma)^4}{2d_\ell}\delta_{kj} \\
&= \sum_\gamma \frac{a_i^\gamma d_\forall^\gamma (\sigma^\gamma)^4}{2d_\ell}\delta_{jk}
\end{aligned}$$

$$\begin{aligned}
\frac{\partial^{\text{nosym}}}{\partial \psi_{\ell^1_{i^1j^1}} \partial \psi_{\ell^2_{i^2j^2}}}\text{NLL} &= \frac{1}{2}\sum_{\gamma|\ell^1\in\gamma} \text{tr}_{d_{>\ell^1}^\gamma}^{d_{<\ell^1}^\gamma}\left[(\mathbf{\Omega}^\gamma)^{-1}\left(\mathbf{I}_{d_{<\ell^2}^\gamma}\otimes \mathbf{J}^{i^2j^2}\otimes \mathbf{I}_{d_{>\ell^2}^\gamma}\right)(\mathbf{\Omega}^\gamma)^{-1}\right]_{i^1j^1} \\
&= \frac{1}{2}\sum_{\gamma|\ell^1\in\gamma} (\sigma^\gamma)^4 \text{tr}_{d_{>\ell^1}^\gamma}^{d_{<\ell^1}^\gamma}\left[\left(\mathbf{I}_{d_{<\ell^2}^\gamma}\otimes \mathbf{J}^{i^2j^2}\otimes \mathbf{I}_{d_{>\ell^2}^\gamma}\right)\right]_{i^1j^1}
\end{aligned}$$

Let us focus on what happens to the stridewise-blockwise trace term under algebraic manipulations.

$$\text{tr}_{d_{>\ell^1}^\gamma}^{d_{<\ell^1}^\gamma}\left[\left(\mathbf{I}_{d_{<\ell^2}^\gamma}\otimes \mathbf{J}^{i^2j^2}+\mathbf{J}^{j^2i^2}\otimes \mathbf{I}_{d_{>\ell^2}^\gamma}\right)\right]_{i^1j^1} = \text{tr}\left[\left(\mathbf{I}_{d_{<\ell^2}^\gamma}\otimes \mathbf{J}^{i^2j^2}\otimes \mathbf{I}_{d_{>\ell^2}^\gamma}\right)\left(\mathbf{I}_{d_{<\ell^1}^\gamma}\otimes \mathbf{J}^{i^1j^1}\otimes \mathbf{I}_{d_{>\ell^1}^\gamma}\right)\right]$$

We now split the $(\psi_{\ell^1_{i^1j^1}}, \psi_{\ell^2_{i^2j^2}})$ computation into two cases, one in which $\ell = \ell^1 = \ell^2$ and another where $\ell^1 \neq \ell^2$. In both cases we use Lemma 22.

Case 1: $\ell^1 \neq \ell^2$

$$\text{tr}_{d_{>\ell^1}^\gamma}^{d_{<\ell^1}^\gamma}\left[\left(\mathbf{I}_{d_{<\ell^2}^\gamma}\otimes \mathbf{J}^{i^2j^2}\otimes \mathbf{I}_{d_{>\ell^2}^\gamma}\right)\right]_{i^1j^1} = \frac{d_\forall^\gamma}{d_{\ell^1}d_{\ell^2}}\delta_{i^2j^2}\delta_{i^1j^1}$$

Case 2: $\ell = \ell^1 = \ell^2$

$$\begin{aligned}
\text{tr}_{d_{>\ell}^\gamma}^{d_{<\ell}^\gamma}\left[\left(\mathbf{I}_{d_{<\ell}^\gamma}\otimes \mathbf{J}^{i^2j^2}\otimes \mathbf{I}_{d_{>\ell}^\gamma}\right)\right]_{i^1j^1} &= \text{tr}\left[\left(\mathbf{I}_{d_{<\ell}^\gamma}\otimes \mathbf{J}^{i^2j^2}\otimes \mathbf{I}_{d_{>\ell}^\gamma}\right)\left(\mathbf{I}_{d_{<\ell}^\gamma}\otimes \mathbf{J}^{i^1j^1}\otimes \mathbf{I}_{d_{>\ell}^\gamma}\right)\right] \\
&= \text{tr}\left[\left(\mathbf{I}_{d_{<\ell}^\gamma}\otimes \mathbf{J}^{i^2j^2}\mathbf{J}^{i^1j^1}\otimes \mathbf{I}_{d_{>\ell}^\gamma}\right)\right] \\
&= \frac{d_\forall^\gamma}{d_\ell}\left(\text{tr}\left[\mathbf{j}^{i^2}\mathbf{j}^{j^2,T}\mathbf{j}^{i^1}\mathbf{j}^{j^1,T}\right]\right) \\
&= \frac{d_\forall^\gamma}{d_\ell}\left(\text{tr}\left[\mathbf{j}^{j^2,T}\mathbf{j}^{i^1}\mathbf{j}^{j^1,T}\mathbf{j}^{i^2}\right]\right) \\
&= \frac{d_\forall^\gamma}{d_\ell}\left(\delta_{j^2i^1}\delta_{j^1i^2}\right)
\end{aligned}$$

This is the nonsymmetric derivative. Symmetricizing requires replacing $\delta_{j^2 i^1}\delta_{j^1 i^2}$ with the average over all symmetries, swapping i^1 and j^1 and swapping i^2 and j^2. These symmetries do not affect the different axis case, but do affect the same-axis case.

One of the symmetries of $\delta_{j^2 i^1}\delta_{j^1 i^2}$ is $\delta_{i^1 i^2}\delta_{j^1 j^2}$, which we will rewrite as $\delta_{(i^1,j^1)(i^2,j^2)}$ to make it more clear what is being compared.

Suppose $i^1 = j^1$ is diagonal. Then this is only true for $i^1 = i^2 = j^1 = j^2$, which if true under one symmetry then it is true under all symmetries.

Suppose now that neither are diagonal, and recall we required $i \leq j$. Then, any symmetries that swap the ordering of i^1 and j^1 will never be satisfied, unless they also swap i^2 and j^2. This lets us rewrite the same-axis case as:

$$\delta_{i^1 j^1 i^2 j^2} + (1 - \delta_{i^1 j^1 i^2 j^2})\frac{\delta_{i^1 i^2}\delta_{j^1 j^2}}{2}$$

This completes the proof.

□

Recall that the only independent parameters are $\mathbf{t}$ and $\psi_{\ell_{ij}}$ for $i \leq j$ and $(i,j) \neq (1,1)$. There are three types of parameters here; $\mathbf{t}$, diagonal $\psi_{\ell_{ii}}$, and off-diagonal $\psi_{\ell_{ij}}$. Of particular interest to us are the interactions between off-diagonal elements $\psi_{\ell_{ij}}$ (where $\delta_{ij} = 0$). It is clear they cannot interact with $\mathbf{t}$ terms due to the δ_{ij} term in its formula. They also cannot interact with any $\psi_{\ell'_{i'j'}}$ in different axes, since the $\delta_{ij}\delta_{i'j'}$ term is only nonzero for the diagonal elements. Finally, the δ terms governing interactions within the same axis are only nonzero when $(i,j) = (i',j')$ (self-interactions) or $(i,j) = (j',i')$, which for off-diagonal elements can never be true due to the $i \leq j$ requirement.

	$\mathbf{t}$	Diagonal ψ_{ii}	Off-diagonal ψ_{ij}
$\mathbf{t}$	Yes[a]	Yes	No
Diagonal ψ_{ii}	Yes	Only in different axes	No
Off-diagonal ψ_{ij}	No	No	No

Table E.2: Allowed interactions between parameters in $\mathbf{F}$.

[a] This depends on $\{\mathbf{a}^\gamma\}$. For most datasets, the most natural choice of identifiable parameterization results in the $\mathbf{t}$-$\mathbf{t}$ interactions forming a diagonal matrix (none of the $\mathbf{t}$ interact).

For all possible interactions, consult Table E.2 - these can be derived by investigating the δ terms in the formula for $\mathbf{F}$. The fact that off-diagonal elements do not interact with anything in $\mathbf{F}$ is quite significant, as seen by the following corollary to Lemma 23.

Corollary 7. *Suppose we order* $\mathbf{F}$ *such that the first rows/columns represent* $\mathbf{t}$ *terms, the next represent diagonal* ψ_{ii} *terms, and the last represent the off-diagonal terms. Then, we have the following:*

$$\mathbf{F} = \begin{bmatrix} \mathbf{A} & \mathbf{0} \\ \mathbf{0} & \mathbf{B} \end{bmatrix}$$

$\mathbf{B}$ *represents the off-diagonal terms, and* ***is a diagonal matrix****. This allows a convenient expression for* $\mathbf{F}^{-1}$*, by inverting blockwise.*

We will now give the example of a dataset $\gamma_1 = (\ell_1, \ell_2), \gamma_2 = (\ell_1, \ell_3)$ where $d_{\ell_1} = 3$ and $d_{\ell_2} = d_{\ell_3} = 2$. Note that $\mathbf{a}^{\gamma_i} = \begin{bmatrix}\delta_{1i} & \delta_{2i}\end{bmatrix}$ in this case (and most cases). This leads to $\mathbf{F}_{\mathbf{t},\mathbf{t}}$ being proportional to the identity matrix. If we let $A^\gamma = \frac{d_\forall^\gamma (\sigma^\gamma)^4}{2}$, then it becomes easy to express the matrix. Exact derivations are left as an exercise to the reader.

$$\Sigma_\gamma A^\gamma = \begin{array}{c} \begin{array}{ccccccccccc} t_1 & t_2 & \ell_1 (2,2) & \ell_1 (3,3) & \ell_2 (2,2) & \ell_3 (2,2) & \ell_1 (1,2) & \ell_1 (1,3) & \ell_1 (2,3) & \ell_2 (1,2) & \ell_3 (1,2) \end{array} \\ \begin{pmatrix} 1 & 0 & \frac{1}{d_{\ell_1}} & \frac{1}{d_{\ell_1}} & \frac{1}{d_{\ell_2}} & \frac{1}{d_{\ell_3}} & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & \frac{1}{d_{\ell_1}} & \frac{1}{d_{\ell_1}} & \frac{1}{d_{\ell_2}} & \frac{1}{d_{\ell_3}} & 0 & 0 & 0 & 0 & 0 \\ \frac{1}{d_{\ell_1}} & \frac{1}{d_{\ell_1}} & \frac{1}{d_{\ell_1}} & 0 & \frac{1}{d_{\ell_1 \ell_2}} & \frac{1}{d_{\ell_1 \ell_3}} & 0 & 0 & 0 & 0 & 0 \\ \frac{1}{d_{\ell_1}} & \frac{1}{d_{\ell_1}} & 0 & \frac{1}{d_{\ell_1}} & \frac{1}{d_{\ell_1 \ell_2}} & \frac{1}{d_{\ell_1 \ell_3}} & 0 & 0 & 0 & 0 & 0 \\ \frac{1}{d_{\ell_2}} & \frac{1}{d_{\ell_2}} & \frac{1}{d_{\ell_1} d_{\ell_2}} & \frac{1}{d_{\ell_1} d_{\ell_2}} & \frac{1}{d_{\ell_2}} & \frac{1}{d_{\ell_2 \ell_3}} & 0 & 0 & 0 & 0 & 0 \\ \frac{1}{d_{\ell_3}} & \frac{1}{d_{\ell_3}} & \frac{1}{d_{\ell_1} d_{\ell_3}} & \frac{1}{d_{\ell_1} d_{\ell_3}} & \frac{1}{d_{\ell_2} d_{\ell_3}} & \frac{1}{d_{\ell_3}} & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & \frac{1}{2d_{\ell_1}} & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & \frac{1}{2d_{\ell_1}} & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & \frac{1}{2d_{\ell_1}} & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & \frac{1}{2d_{\ell_2}} & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & \frac{1}{2d_{\ell_3}} \end{pmatrix} \end{array} \begin{array}{l} t_1 \\ t_2 \\ \ell_1 (2,2) \\ \ell_1 (3,3) \\ \ell_2 (2,2) \\ \ell_3 (2,2) \\ \ell_1 (1,2) \\ \ell_1 (1,3) \\ \ell_1 (2,3) \\ \ell_2 (1,2) \\ \ell_3 (1,2) \end{array}$$

With everything in place, we can now define the hypothesis test.

Theorem (Theorem 12)**.** *Under the null hypothesis* $\mathbf{\Omega}^\gamma = \frac{1}{(\sigma^\gamma)^2} \mathbf{I}_{d_\forall^\gamma}$*, we have the following distribution for each* $\psi_{\ell_{ij}}$ *independently:*

$$\sqrt{\frac{\sum_{\gamma | \ell \in \gamma} d_\forall^\gamma \left(\sigma^\gamma\right)^4}{d_\ell}} \frac{\psi_{\ell_{ij}}}{2} \dot{\sim} \mathcal{N}(0,1)$$

Proof. As noted in Corollary 7, the edge-edge interactions in $\mathbf{F}$ form a diagonal matrix, and there are no cross-terms connecting edges to diagonal and $\mathbf{t}$ parameters. Thus, inversion is simple.

By Lemma 23, each edge-edge precision term for axis ℓ in $\mathbf{F}$ can be expressed as $\sum_{\gamma | \ell \in \gamma} \frac{d_\forall^\gamma (\sigma^\gamma)^4}{4 d_\ell}$. In $\mathbf{F}^{-1}$, this becomes $\frac{4 d_\ell}{\sum_{\gamma | \ell \in \gamma} d_\forall^\gamma (\sigma^\gamma)^4}$.

Thus, under the null hypothesis, we have that:

$$\psi_{\ell_{ij}} \dot{\sim} \mathcal{N}\left(0, \frac{4 d_\ell}{\sum_{\gamma | \ell \in \gamma} d_\forall^\gamma \left(\sigma^\gamma\right)^4}\right)$$

$$\sqrt{\frac{\sum_{\gamma | \ell \in \gamma} d_\forall^\gamma \left(\sigma^\gamma\right)^4}{d_\ell}} \frac{\psi_{\ell_{ij}}}{2} \dot{\sim} \mathcal{N}(0,1)$$

□

This completes the derivations. While deriving such a test was tedious, it gives us an interpretable way to set the threshold for our recovered graphs, by choosing to only keep statistically significant edges!

E.10 Preserving input sparsity

The final practical issue to surmount is that of preserving input sparsity. $\text{mat}_\ell \left[\{\mathcal{D}^\gamma\}_\gamma\right]$ has $d_\forall$ elements, but in many contexts, such as transcriptomics, the dataset will be highly sparse; the majority of the

input is full of zeros. Ideally, we would never 'densify' the matrix. Using COCA, however, requires computing the matrix of ranks, which will be fully dense. If unaddressed, this would enforce an $\Omega(d_\forall)$ minimum memory requirement. We will demonstrate the problem with an example, where Φ represents the cumulative density function of the normal distribution.

Example 28 (Demonstration that ranking densifies data)**.**

$$\begin{bmatrix} 3 & 0 & -2 & 0 & 4 \\ 0 & -1 & -3 & 0 & -2 \end{bmatrix} \rightarrow \begin{bmatrix} 4 & 2 & 1 & 2 & 5 \\ 4 & 3 & 1 & 4 & 2 \end{bmatrix} \qquad \text{(rank data)}$$

$$\rightarrow \begin{bmatrix} \Phi^{-1}\left(\frac{4}{6}\right) & \Phi^{-1}\left(\frac{2}{6}\right) & \Phi^{-1}\left(\frac{1}{6}\right) & \Phi^{-1}\left(\frac{2}{6}\right) & \Phi^{-1}\left(\frac{5}{6}\right) \\ \Phi^{-1}\left(\frac{4}{6}\right) & \Phi^{-1}\left(\frac{3}{6}\right) & \Phi^{-1}\left(\frac{1}{6}\right) & \Phi^{-1}\left(\frac{4}{6}\right) & \Phi^{-1}\left(\frac{2}{6}\right) \end{bmatrix} \qquad \text{(map to normal)}$$

$$\approx \begin{bmatrix} 0.43 & -0.43 & -0.97 & -0.43 & 0.97 \\ 0.43 & 0 & -0.97 & 0.43 & -0.43 \end{bmatrix}$$

The exact values depend on how you handle ties when ranking data, but the general idea remains the same regardless. The trick to avoiding the densification is to realize that, in each row, all zero values get mapped to the same final value. We can rewrite our example as:

$$\begin{bmatrix} 0.43 + 0.43 & 0 & -0.97 + 0.43 & 0 & 0.97 + 0.43 \\ 0 & -0.43 & -0.97 - 0.43 & 0 & -0.43 - 0.43 \end{bmatrix} + \begin{bmatrix} -0.43 \\ 0.43 \end{bmatrix} \mathbf{1}_{1\times 5}$$

Let z_i be the normally-distributed value that the zeros in row i get mapped to. We can express our transformation more generally:

$$\begin{bmatrix} \mathbf{x}_1 \\ ... \\ \mathbf{x}_a \end{bmatrix} = \begin{bmatrix} \mathbf{x}_1 - z_1 \\ ... \\ \mathbf{x}_a - z_a \end{bmatrix} + \mathbf{z}\mathbf{1}^T$$

Crucially, we have expressed our transformed data as the sum of a matrix with the same sparsity structure as our original data, and a rank-one matrix. If we compute the SVD on the first matrix, we can then use a rank-one update algorithm to calculate the SVD of the sum of the two matrices (Brand, 2006). It is not difficult to work out $\mathbf{z}$, and hence we can perform this procedure without ever densifying our input matrix.

Algorithm	Cluster	Biological Process	p-value
GmGM (p=0.05)	1 (67 genes)	cell-cell adhesion	1.0×10^{-9}
		apoptotic process	3.1×10^{-9}
		positive regulation of cellular process	5.1×10^{-9}
	2 (65 genes)	protein maturation	5.6×10^{-7}
		response to endoplasmic reticulum stress	1.3×10^{-6}
		organonitrogen compound metabolic process	9.1×10^{-6}
	3 (60 genes)	immune response	9.5×10^{-15}
		leukocyte mediated immunity	3.8×10^{-14}
		immune system process	8.7×10^{-14}
	4 (43 genes)	GO:0048523	4.7×10^{-5}
		GO:0051172	6.8×10^{-5}
		GO:0031324	8.9×10^{-5}
	5 (27 genes)	positive regulation of immune system process	8.3×10^{-15}
		regulation of immune response	2.3×10^{-14}
		GO:0002504	2.4×10^{-14}
	6 (14 genes)	intracellular sequestering of iron ion	8.1×10^{-3}
		sequestering of iron ion	1.1×10^{-2}
		negative regulation of actin filament polymerization	1.3×10^{-2}

Table E.3: GO terms with long names have been replaced with their GO ID; most were of the form "negative regulation of cellular/nitrogen compound (metabolic) process". Immune-related GO terms have been colored blue. These were identified using the Python API of the GProfiler (Kolberg et al., 2023) functional enrichment tool.

E.11 Additional experiments

E.11.1 Synthetic data

In this section and in the main paper, we generated ground truth graphs from the Barabasi-Albert distribution. We chose this distribution because it is a scale-free distribution; many real-world networks, such as social networks and gene regulatory networks, are approximately scale-free. We then generated synthetic data using these graphs and the Kronecker-sum-structured normal distribution.

To test performance, and area-under-PR curves (AUPR) for GmGM and our proposed modifications to GmGM; when an algorithm correctly identifies an edge, it is considered a true positive. When comparing to prior work, we used their default convergence tolerance and max iterations.

We report our PR curves and their AUPRs in Figure E.2. It is clear that, for a fixed amount of components, the AUPR stays roughly the same as the axis size increases. This is true even as the percentage of variance explained decreases. There is a clear trade-off between the performance of the method and the number of principal components chosen. Keeping 500 components still has high performance even as the percent of variance explained decreases; this shows the method is suitable on problem sizes that full-rank methods such as GmGM simply cannot handle.

E.11.2 Sanity check of million-cell results

At the beginning of this paper, we promised scalability to a million cells. Here, we will use a 1,248,980 cell, 36,571 gene PBMC dataset by Yazar et al., 2022. For preprocessing, we removed any cells and genes that had 0 counts everywhere. Then, we kept only the top 2,200 highly variable genes. No cells had uniformly 0 counts, so the post-processing dataset size contained 1,249,980 cells and 2200 genes.

Due to the size of the dataset, we kept only the top 50 principal components ($\sim$50% of the explained variance). We keep only the statistically significant edges and use the Nonparanormal Skeptic. Overall, 5890 edges were statistically significant. We do not compare to prior work as this dataset is far too large for them to handle.

We then investigated the GO terms of the clusters of our graphs. If our graph were nonsense, we would expect there to be no significant GO terms associated with any clusters. However, in Table E.3 we can see that this is not the case; we find six clusters corresponding to meaningful groupings of genes, including immune-related groupings (which we would expect and hope to find in a PBMC dataset, which contain mostly immune cells).

This example ran in only a couple of minutes - faster than would be predicted from the runtimes in Figure 6.2, likely due to the high sparsity of the dataset[1].

[1] Figure 6.2 gives performance on dense datasets, whereas our algorithm, unlike prior work, benefits substantially from sparsity in the dataset as well. This is because there are many efficient partial eigendecomposition routines for sparse

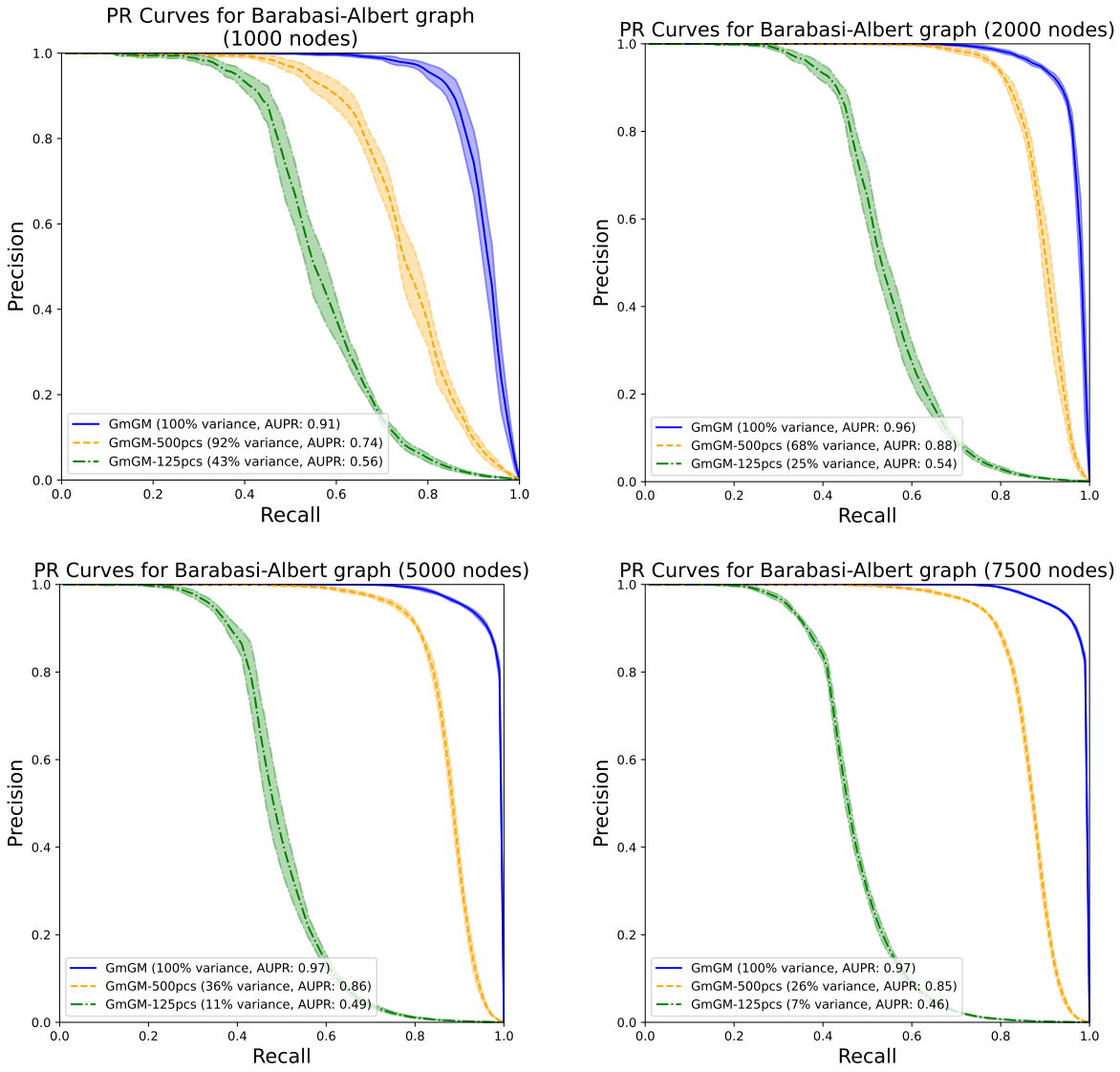


Figure E.2: Precision-recall curves, with AUPR reported, for $n \in \{1000, 2000, 5000, 7500\}$ datasets of shape $n \times n$ whose graphs ome from a Barabasi-Albert distribution. We compare the full-rank GmGM algorithm, along with our proposed algorithm with 500 principal components and a more extreme 125 principal components. We see that 500 principal components still yield good results as the dataset size increases.

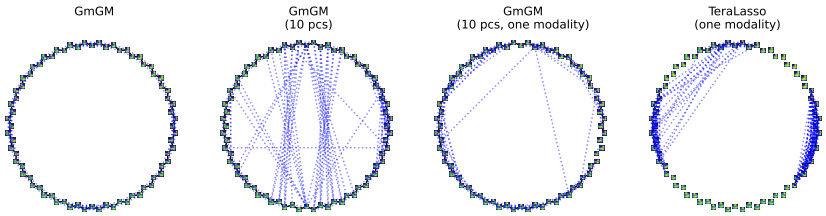


Figure E.3: A comparison of several methods to find the frame graph.

Connections	Adjacent	One off	Two off	Three off	More	Total	Method
GmGM	72	72	25	4	0	173	p=0.05
GmGM (10pcs)	72	64	13	0	51	200	p=0.05
GmGM (10pcs, one modality)	66	58	44	13	19	200	Thresholding
TeraLasso (one modality)	33	31	26	22	85	197[a]	Thresholding

[a] We were not able to get exactly 200 edges

Table E.4: The performance of several methods on reconstructing the frame graph of COIL-20 videos. The adjacent column denotes how many edges were between adjacent frames (the 'correct' edges), whereas the later columns denote how many edges exist between frames i and $i \pm (n+1)$ for various n. The More column summarizes all connections where $n > 3$.

E.11.3 COIL Dataset

It is difficult to test on real-world data, as ground truth graphs are often unknown. This is perhaps why the COIL-20 video dataset (Nene et al., 1996) has become a standard test for multi-axis models.

The dataset consists of 20 videos of 72 frames and 128x128 pixels. Each video consists of a simple object rotating 360 degrees, and thus the frame graph can reasonably expected to correspond to this circular structure. To test this, we considered the dataset to have the form $\left\{\gamma_i = (\ell_{\text{frame}}, \ell_{\text{row}(i)}, \ell_{\text{col}(i)})\right\}_i$, that is, that there is a singular frame graph describing the rotational structure of the images, and then individual row and column graphs for each object (since the structure of each object may be different). In other words, the dataset had 20 modalities, each of size 72x128x128. TeraLasso can only handle a single modality, so to compare against it we ran our algorithm another time, this time on only one modality (the duck object, "obj1"). We compared the results using all principal components and when using only the top 10.

In the aforementioned experiments, we kept only the edges that were statistically significant to the 5% level after applying the Bonferroni correction. No edges were statistically significant, so instead we kept the same amount of edges as were found to be significant when considering the whole dataset (200). We then did the same with TeraLasso.

The results can be compared visually in Figure E.3. These show that GmGM over all modalities does the best, which is to be expected - it uses the most information and does not make a low-rank assumption. TeraLasso cannot handle data of this form; however, even if we restrict our data to only a single modality and 10 principal components, our method performs best. Numerical values are given in Tables E.4 and E.5; we can see that, when considering statistically significant edges in the whole dataset, our method finds all the correct edges (even with only 10 principal components). The false positives it finds are typically close-to-correct, which is reasonable.

---

data.

Connections	Precision/Recall (exact)	Precision/Recall (one off)	Precision/Recall (two off)	Precision/Recall (three off)
GmGM	**41.6%**/**100%**	**83.2%**/**100%**	**97.6%**/**78.2%**	**100%**/60%
GmGM (10pcs)	36%/**100%**	60%/94.4%	74.5%/69%	74.5%/51.7%
GmGM (10pcs, one modality)	33%/91.7%	62%/86%	84%/77.8%	90.5%/**62.8%**
TeraLasso (one modality)	16.8%/45.8%	32.5%/44.4%	45.7%/41.7%	56.7%/38.9%

Table E.5: The precisions and recalls for each algorithm, when we consider correct edges to be those between edges i and $i \pm (n+1)$, for $n = 0$ ('exact'), $n = 1$ ('one off'), $n = 2$ ('two off'), and $n = 3$ ('three off'). In each column, the best performing model's results were given in bold for each metric. **Bold numbers** represent the best performance overall, and blue numbers represent the best performance when restricted to one modality.

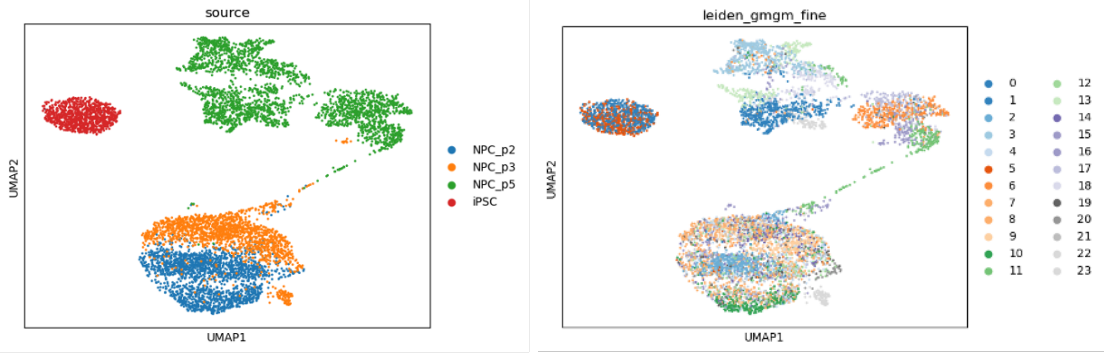


Figure E.4: (Left) The four time points of the data, displayed as a UMAP plot. (Right) The cell clusters after applying Leiden to GmGM's output graph.

E.12 Neuronal scRNA-seq dataset details

The dataset consists of four separate time points; the original iPSC cells, as well as cells taken from passages 2, 3, and 5 ('p2', 'p3', and 'p5'); these are shown graphically in Figure E.4. In total, there are 6876 cells and 29,324 genes. The dataset was generated in-house; the following subsections describe how this was done.

E.12.1 Culture of induced pluripotent stem cells (iPSCs)

STEMCELL Technologies's SCTi003-A line (STEMCELL Technologies; #200-0511) were cultured in complete mTeSR plus media (STEMCELL Technologies) on Matrigel® coated (Corning) 6-well plates (Helena Biosciences) and passaged at 1:40-1:60 split ratio with ReLeSR (STEMCELL Technologies) at approximately 70% confluency. Cultures were regularly tested for pluripotency by immunocytochemistry and mycoplasma using MycoStrip lateral flow tests (InvivoGen).

E.12.2 iPSC-derived neural progenitor cell (NPC) differentiation

Neural induction of iPSCs was performed using the STEMdiff SMADi Neural Induction Kit (STEMCELL Technologies) according to the manufacturer's monolayer protocol. iPSCs were dissociated with Accutase (Sigma) and resuspended as a single-cell suspension in DMEM/F12 (Thermo Fisher Scientific) for counting. 2.5×10^6 cells were pelleted at 300g for 5 minutes and resuspended in Neural Induction + SMADi Medium (NIM; STEMCELL Technologies) supplemented with 10 µM Y-27632 onto hESC-qualified Matrigel®-coated 6-well plates. Medium was replaced daily for 6–9 days, after which cells were passaged using Accutase and re-plated at 2×10^6 cells per well. This induction–passage cycle was repeated until passage 3, at which point cells were re-plated at 1.25×10^6 cells per well and subsequently maintained in STEMdiff Neural Progenitor Medium (STEMCELL Technologies) for all following passages.

Cell type	Marker genes
induced pluripotent stem cells	NANOG, POU5F1, LIN28A, DPPA4, TDGF1, ZSCAN10, UTF1
neural progenitors	PAX6, HES1, NES1, MSI1, VIM, FABP7
cortical neurons	TBR1, BCL11B, MAP2, RBFOX3, SLC17A7, SYT1, GRIN2B
astrocytes	NFIA, SOX9, GFAP, AQP4, S100B, ALDH1L1, GLUL, SLC1A3, SLC1A2
oligodendrocytes	SOX10, MYRF, MBP, MOG, PLP1, MAG, CNP
neural ectoderm	SOX1, ZIC1, OTX2, LHX2, EMX2, DACH1, SIX3

Table E.6: Marker genes for various neuronal cell types.

E.12.3 Cell fixation

iPSCs were dissociated with Accutase at low passage and approximately 70% confluency prior to fixation. NPCs were similarly lifted with Accutase on their designated passage day; for example, p2 NPCs were collected at passage 2 of differentiation. This procedure was performed for p2, p3, and p5 NPCs. Following cell counting, 1×10^6 to 2×10^6 cells were fixed using the Evercode Cell Fixation v2 kit (Parse Biosciences) according to the manufacturer's user manual (v2.1.1). Final post-fixation yields ranged from 7×10^5 to 1×10^6 cells, which were subsequently carried forward to barcoding.

E.12.4 Barcoding, cDNA library preparation and sequencing

Approximately 7×10^4 per sample were processed for barcoding and library preparation using the Evercode WT kit (v2.2.1) according to the manufacturers protocol (Parse Biosciences). Nine libraries were generated and assessed for fragment size and quality using an Agilent TapeStation prior to sequencing. The libraries were subsequently pooled and sequenced across two P4 flow cells on an Illumina NextSeq2000 using a 2x200bp paired end cycle. The resulting fastq files were processed using Trailmaker (Parse Biosciences) to generate gene read-counts per cell per time point. This count matrix is what was passed into GmGM.

E.13 Neuronal dataset cell marker genes

We used the marker genes given in Table E.6 to score each cluster. A cell cluster was considered statistically significantly associated with a gene if the gene expression was significantly different in that cluster compared to others; cell clusters were then assigned the cell type corresponding to the one whose marker genes were the most commonly significantly differentially expressed.

E.14 GO terms for network metrics

In Figure 6.3, we picked which GO terms in two groups - those which are important processes dominant in all cells, and those which are part of the development of a key component found in neurons. For each GO term category (such as 'DNA replication' or 'axon development'), we selected the GO term most relevant to the process, all GO terms that are annotated as *regulates*, *positively_ regulates*, *negatively_ regulates*, or *occurs_ in* the original term, and then any children and descendents of those GO terms via *part_ of* or *is_ a* relations. The precise GO terms are given in Table E.7.

E.15 Neuronal scRNA-seq dataset full results

In this section, we give a breakdown of our gene modules and cell clusters. The GmGM cell clusters (and their links to the GmGM gene modules) are given in Table 6.1, the hdWGCNA gene modules are given in Table E.8, and the GmGM gene modules are given in Table E.9; there are no hdWGCNA cell clusters as hdWGCNA does not create a cell network.

For the gene modules, we defined 'interesting' modules which satisfied one of the following criteria: high Moran's I (>0.3), high lncRNA percentage (>0.3), being related to transcription factors, being significantly expressed in a cell cluster, or being significantly expressed only in a specific passage.

Reason	Category	Fundamental GO terms	Regulators/Occurs-In
Fundamental Process	DNA Replication	GO:0006260	GO:0008156, GO:0045740, GO:0006275
Fundamental Process	Cell Cycle	GO:0007049	GO:0045786, GO:0045787, GO:0051726
Fundamental Process	Translation	GO:0006412	GO:0032062, GO:0017148, GO:0140244, GO:0140245, GO:0140243, GO:0099578, GO:0099577, GO:0032055, GO:0043557, GO:0043555, GO:0045727, GO:1902010, GO:0006417, GO:0032061, GO:0032056, GO:0099547, GO:0036490, GO:0036493, GO:0043556, GO:0032939
Fundamental Process	Splicing	GO:0008380	GO:0033119, GO:0033120, GO:0042484
Neurogenesis	Nervous System Development	GO:0007399	GO:0051961, GO:0051960, GO:0051962
Neurogenesis	Synapse Organization	GO:0050808, GO:0045202	GO:0140241, GO:0099574, GO:0140243, GO:1905809, GO:0050807
Neurogenesis	Dendrite Development	GO:0016358, GO:0030425	GO:0050773, GO:2000171, GO:1900006
Neurogenesis	Axon Development	GO:0061564, GO:0030424	N/A
Neurogenesis	Myelination	GO:0042552, GO:0043209	GO:0031642, GO:0031641, GO:0031643, GO:0019911

Table E.7: GO term choices for metric comparison in Figure 6.3. Does not show *is_a* or *part_of* descendants, which were also included.

Every module of hdWGCNA was 'interesting'; in contrast, only 39 GmGM modules were. Interesting modules are likely indicative of some true biological process; for example, GmGM modules m5, m7, and m117 are all interesting, and based on the GO terms represent the cell cycle, the ribosome, and mitochondrial genes, respectively.

Module	Top 3 genes by centrality	Select GO terms	Size	Annotation	Moran's I
Unassigned	N/A	N/A	14405	A catch-all module.	0.56
Grey	TSPAN6, CIBAR1, C3orf62	DNA-binding transcription factor activity, regulation of biological process	6399	Transcription factor module, only expressed in non-iPSCs.	0.99
Turquoise	LINC00678, LNC-PRESS2, ESRG	gene expression, biological regulation, RNA binding	6146	Only expressed in iPSCs.	1.04
Blue	SOX6, ZBTB20, ENSG00000258631	nervous system development, neuron projection	903	Only expressed in non-iPSCs.	1.01
Brown	DMD, MIR100HG, ENSG00000286757	anatomical structure development, cell differentiation, cilium organization	559	Expressed only in a subset of p5s.	0.96
Yellow	HMGCS1, SCD, FADS1	cell differentiation, system development	409	Only expressed in p2/p3s.	0.94
Green	ECT2, SMC4, KIF20B	cell cycle, chromosome segregation, cell division	262	Cell cycle module.	0.83
Red	CFTR, SORCS3, PDGFD	regulation of biological process, nervous system development, cell differentiation	241	Only expressed in a subset of p5s.	0.91

Table E.8: Every hdWGCNA module, in order of size. Moran's I is a statistic that expresses the tendency of 'nearby' cells to have a similar average expression of a given module. In this case, 'nearby' is defined by the Euclidean distance of gene expression vectors of each cell.

Module	Top 3 genes by centrality	Select GO terms	Size	Annotation	Moran's I
m0	DYNAP, DENRP3, ENSG00000250855	N/A	9047	A catch-all module.	0.15
m1	**FOXH1**, LAD1, FLT1	N/A	1756	Only expresses in iPSCs.	1.07

Table E.9 (*multi-page*)**.** Every GmGM module, in order of size. Bold genes either are, are antisense to, or differential transcripts of transcription factors. Select GO terms being N/A does not necessarily indicate a lack of GO terms, merely a lack of ones we felt worth highlighting. Bold modules were considered 'interesting'.

Module	Top 3 genes by centrality	Select GO terms	Size	Annotation	Moran's I
m2	CROCC2, ADGB, LMNTD1	cilium movement, cilium organization, axoneme assembly	600	Only expressed in a subset of p5s.	0.82
m3	SLC17A6, STMN2, **LHX9**	nervous system development, synaptic signalling, neuron projection morphogenesis	433	Only expresses in potential cortical neurons.	0.83
m4	ENSG00000288903, CARMN, CPED1	extracellular matrix organization, cell surface receptor signaling pathway	319	Only expressed in a subset of p5s.	0.71
m5	SGO2, KIF14, CCNB1	many cell-cycle related GO terms	315	Only expressed in a subset of p2/p3/p5s.	0.82
m6	PDZRN4, SGPP2, CT75	N/A	313	Only expresses in a subset of p4s, high amount of lncRNA, includes and is highly correlated with expression of GRM8.	0.94
m7	RPS6, RPL7A, RPS8	Ribosome-related GO terms	303	Ribosomal gene module.	0.77
m8	ANKRD1, SERPINE1, COL8A1	angiogenesis, vasculature development, DNA-binding transcription factor activity	300	Next three most central genes are all unnamed lncRNA; mostly expressed in p2s/p3s. One of three transcription-factor-related clusters.	0.58

Table E.9 (*multi-page*)**.** Every GmGM module, in order of size. Bold genes either are, are antisense to, or differential transcripts of transcription factors. Select GO terms being N/A does not necessarily indicate a lack of GO terms, merely a lack of ones we felt worth highlighting. Bold modules were considered 'interesting'.

Module	Top 3 genes by centrality	Select GO terms	Size	Annotation	Moran's I
m9	SPARCL1, SORCS3, CA12	branching morphogenesis of an epithelial tube, branching involved in ureteric bud morphogenesis, mesonephric tubule morphogenesis	276	Only expressed in a subset of p5 cells. Contains and correlates strongly with SORCS3.	0.89
m10	CDKN1A, INPP5D, ENSG00000251095	cell death/apoptosis GO terms	262	Not expressed in p5s.	0.57
m11	MALAT1, MT-RNR1, MT-RNR2	neurogenesis	251	Not expressed in iPSCs.	0.90
m12	BRIP1, DTL, ATAD2	DNA replication, DNA metabolic process, various cell-cycle terms	249	Only expressed in a subset of cells.	0.66
m13	**EBF2**, **NEUROD4**, **EBF3**	nervous system development, neurogenesis, DNA-binding transcription factor activity	239	One of three transcription-factor-related clusters, only expresses in potential cortical neurons.	0.70
m14	MIR9-2HG, **MEF2C-AS2**, WNT7A	nervous system development, cell differentiation, neurogenesis	212	Only expresses in a subset of p2/p3/p5 cells. Contains and correlates with he expression of MIR9-2HG. High lncRNA percentage.	0.76
m15	SEC24B, USB8, ZNF106	localization, nucleic acid biosynthetic process	204	N/A	0.04
m16	LINC02306, ENSG00000253693, LINC03000	nervous system development, neurogenesis	203	Only expressed in a subset of p5s.	0.74

Table E.9 (*multi-page*)**.** Every GmGM module, in order of size. Bold genes either are, are antisense to, or differential transcripts of transcription factors. Select GO terms being N/A does not necessarily indicate a lack of GO terms, merely a lack of ones we felt worth highlighting. Bold modules were considered 'interesting'.

Module	Top 3 genes by centrality	Select GO terms	Size	Annotation	Moran's I
m17	CCT2, CCDC25, WDR1	N/A	201	N/A	0.39
m18	CSNK2A1, DDR1, GDI1	N/A	199	N/A	0.08
m19	TXNL1, TMEM178A, FXYD6	N/A	198	Significantly overexpressed in cell cluster 4.	0.40
m20	PRANCR, C1QTNF7-AS1, ENSG00000250195	N/A	198	Not expressed in iPSCs. High lncRNA percentage.	0.42
m21	MED27, MBTD1, **MGA**	N/A	186	N/A	0.06
m22	CLCN3, PHF20L1, **ZNF532**	N/A	184	N/A	0.11
m23	GDF7, **PAX3**, SV2C	anatomical structure morphogenesis, glutamatergic synapse	181	Only expressed in a subset of non-iPSC cells.	0.64
m24	CDC20B, CCNO, **FOXN4**	centriole assembly, centriole replication, de novo centriole assembly involved in multiciliated epithelial cell differentiation	180	Only expressed in a subset of p5 cells.	0.53
m25	NOS2, AGBL1, **BARHL1**	N/A	178	Mostly expressed in p2/p3 cells, some in p5s.	0.85
m26	LENG8, 7SK, ATN1	N/A	177	N/A	0.18
m27	**TBR1**, ISLR2, **LHX1-DT**	neuron projection morphogenesis, telencephalon development, synapse organization, dendrite, neurogenesis	177	Only expresses in potential cortical neurons.	0.76

Table E.9 (*multi-page*)**.** Every GmGM module, in order of size. Bold genes either are, are antisense to, or differential transcripts of transcription factors. Select GO terms being N/A does not necessarily indicate a lack of GO terms, merely a lack of ones we felt worth highlighting. Bold modules were considered 'interesting'.

Module	Top 3 genes by centrality	Select GO terms	Size	Annotation	Moran's I
m28	MTOR, PIN4, INO80D	nucleic acid metabolic process, nucleobase-containing compound metabolic process	175	N/A	0.23
m29	**UBP1**, INPP4A, PRKCSH	N/A	174	N/A	0.12
m30	RSPO2, **LMX1A**, DYNLRB2-AS1	regulation of cell differentiation, regulation of signal transduction	174	Expressed in all non-iPSC cells, more expression in a subset of p5 cells.	0.86
m31	CAPN2, EXOC6, BDNF-AS	gene expression	172	N/A	0.07
m32	BMP2K, FARP2, TAF2	gene expression, RNA biosynthetic process	171	N/A	0.08
m33	**E2F5**, CCDC136, PREPL	gene xpression	171	N/A	0.08
m34	CPSF7, FGGY, SLC20A2	transcription by RNA polymerase II, regulation of transcription by RNA polymerase II	170	N/A	0.07
m35	ARHGAP35, MAPKAP1, ORC2	N/A	169	N/A	0.05
m36	ZNF84, KDM5C, EXOC3	positive regulation of post-translational protein modification	169	N/A	0.05
m37	RPAP3, SAFB, PEX14	N/A	168	N/A	0.10
m38	RAD50, SFSWAP, TBCA	MLL3/4 complex	167	N/A	0.06
m39	DDX21, ENSG00000258131, LARP4	N/A	167	N/A	0.06
m40	RHEB, LINC00632, PRDX3	N/A	166	N/A	0.07

Table E.9 (*multi-page*)**.** Every GmGM module, in order of size. Bold genes either are, are antisense to, or differential transcripts of transcription factors. Select GO terms being N/A does not necessarily indicate a lack of GO terms, merely a lack of ones we felt worth highlighting. Bold modules were considered 'interesting'.

Module	Top 3 genes by centrality	Select GO terms	Size	Annotation	Moran's I
m41	MTREX, IP6K1, MED4	gene expression, localization, translation	164	N/A	0.16
m42	CLEC2A, ENSG00000289578, NECTIN3-AS1	N/A	164	Expressed only in a subset of p2/p3 cells, high lncRNA percentage.	0.48
m43	PSMB7, KATNIP, SOS2	inositol phosphate metabolic process, "phospho-transferase activity, phosphate group as acceptor"	163	N/A	0.13
m44	MAPK8IP3, MED1, ANAPC1	Glycosylphosphatidylinositol (GPI)-anchor biosynthesis	163	N/A	0.09
m45	DZIP1, USP7, **AHDC1**	N/A	163	N/A	0.10
m46	GBF1, CERK, **MRTFA**	mitochondrion	161	N/A	0.08
m47	TARBP1, GCFC2, ESYT2	N/A	158	N/A	0.05
m48	SRP14, CEBPZ, CTTNBP2NL	RNA biosynthetic process, RNA metabolic process	158	N/A	0.14
m49	SLC38A6, MIER1, ICE2	proteolysis	157	N/A	0.11
m50	HMGCS1, MSMO1, HMGCR	sterol biosynthetic process, lipid biosynthetic process	157	Primarily expressed in p2/p3s, some expression in iPSCs and a subset of p5s.	0.61
m51	ENSG00000258312, MAMDC2, CCN1	hippo signaling, cellular response to stimulus	156	Only expressed in p2/p3s.	0.76
m52	SPG11, SSR1, SNX4	N/A	155	N/A	0.05
m53	UFL1-AS1, RGS12, MYH9	N/A	155	N/A	0.07
m54	TBC1D16, PRR14L, CMSS1	N/A	154	N/A	0.08

Table E.9 (*multi-page*)**.** Every GmGM module, in order of size. Bold genes either are, are antisense to, or differential transcripts of transcription factors. Select GO terms being N/A does not necessarily indicate a lack of GO terms, merely a lack of ones we felt worth highlighting. Bold modules were considered 'interesting'.

Module	Top 3 genes by centrality	Select GO terms	Size	Annotation	Moran's I
m55	WASL, DDAH2, PCGF3-AS1	gene expression	N/A	152	0.04
m56	XIST, LINC01515, HERC2P3	N/A	151	N/A	0.16
m57	RPS6KC1, PPFIA1, MCPH1	N/A	150	N/A	0.12
m58	CUL4B, QARS1, **PRDM4**	methylation	148	N/A	0.04
m59	LDAH, FUBP1, KLHDC10	N/A	147	N/A	0.05
m60	IFT81, GANAB, TNFAIP3	immune system response, immune system process	147	N/A	0.18
m61	ZNF782, ENSG00000263551, SLCO6A1	N/A	146	N/A	0.05
m62	CTNNA2, SMOC1, **SIM2**	neuron projection morphogenesis	145	Expressed in all non-iPSC cells.	0.80
m63	ATG5, ERCC5, PCBP1-AS1	RNA processing, RNA metabolic process, gene expression	144	Expressed in all cells, decreasing through passages.	0.34
m64	COPA, DHX57, MAPK1	N/A	144	N/A	0.06
m65	NDFIP1, SMURF1, TMEM245	gene expression, RNA metabolic processing, lncRNA processing	143	N/A	0.09
m66	RRAS2, FAM228B, IREB2	N/A	142	N/A	0.07
m67	CDK12, IARS1, ATP6V1D	tRNA metabolic process, RNA processing	141	N/A	0.10
m68	SENP5, AMOTL1, ATE1	N/A	140	N/A	0.16
m69	BMERB1, VPS4A, RIMKLB	N/A	140	N/A	0.12
m70	RICTOR, ZCCHC7, IGSF1	N/A	139	N/A	0.16

Table E.9 (*multi-page*)**.** Every GmGM module, in order of size. Bold genes either are, are antisense to, or differential transcripts of transcription factors. Select GO terms being N/A does not necessarily indicate a lack of GO terms, merely a lack of ones we felt worth highlighting. Bold modules were considered 'interesting'.

Module	Top 3 genes by centrality	Select GO terms	Size	Annotation	Moran's I
m71	ENSG00000197585, ENSG00000237461, KYAT3	N/A	139	High lncRNA percentage.	0.26
m72	SLC25A46, LUC7L2, COX16	N/A	139	N/A	0.08
m73	GARRE1, HSPA8, AKAP8L	N/A	138	N/A	0.17
m74	PPP2R2A, PGM3, MFSD8	N/A	138	N/A	0.05
m75	ZBED5, MRE11, TZNRD1	mitochondrial membrane organization	137	N/A	0.09
m76	RBM5, LAMP2, SCFD2	N/A	136	N/A	0.03
m77	NUMA1, NCK1, CFAP418-AS1	succinate dehydrogenase (quinone) activity	132	N/A	0.04
m78	UBE2D2, **ATF6**, SECISBP2L	N/A	132	N/A	0.09
m79	NEK9, HYCC1, **ZNF577**	N/A	132	N/A	0.09
m80	WDR70, CTBP2, **FOXO3**	serotonergic synapse, glutamatergic synapse	131	N/A	0.10
m81	RHBDD1, BDP1, CDC14A	transcription factor TFIIIB complex	129	N/A	0.07
m82	ABL1, RAPH1, PIGU	N/A	127	N/A	0.07
m83	ENSG00000285713, RNF214, AMBRA1	N/A	127	N/A	0.04
m84	HADHA, TMSB10, ACTB	N/A	126	N/A	0.33
m85	DNAJB14, HFM1, INTU	positive regulation of protein-containing complex disassembly	125	N/A	0.06
m86	ZSWIM5, UBL3, PPM1A	N/A	124	N/A	0.08
m87	ARMCX4, BSDC1, DCTN5	DNA-templated transcription, regulation of DNA-templated transcription	121	N/A	0.09

Table E.9 (*multi-page*)**.** Every GmGM module, in order of size. Bold genes either are, are antisense to, or differential transcripts of transcription factors. Select GO terms being N/A does not necessarily indicate a lack of GO terms, merely a lack of ones we felt worth highlighting. Bold modules were considered 'interesting'.

Module	Top 3 genes by centrality	Select GO terms	Size	Annotation	Moran's I
m88	GALNT13, TP53BP2, ANAPC10	N/A	121	N/A	0.06
m89	KDELR2, MEMO1, AIG1	N/A	120	N/A	0.06
m90	**NFIA**, ENSG00000243620, **NFIB**	nervous system development, neuron differentiation, neurogenesis, DNA-binding transcription factor activity	120	One of the three transcription factor modules, highly expressed in p5s.	0.93
m91	ADAMTS18, **GLIS3**, LGR5	cell adhesion, cell junction assembly, long-term synapse potentiation	119	Not expressed in iPSCs, **NR2F1** and related genes are very central to the module as well.	0.85
m92	ATG2B, ALG11, MICU3	N/A	116	N/A	0.05
m93	CDC42SE2, RAB2A, FAM222B	N/A	115	N/A	0.16
m94	AHCTF1, CHIC1, DIS3L2	N/A	115	N/A	0.11
m95	AGO4, PLEKHA8, PHF6	N/A	115	N/A	0.18
m96	NUMB, DLG3, UBE2B	regulation of primary metabolic process, regulation of macromolecule metabolic process	113	N/A	0.07
m97	ARL13B, IFRD1, DPH6	N/A	108	N/A	0.07
m98	EML4, ELMO2, ANKRD6	N/A	107	N/A	0.10
m99	PRKCI, PTTG1IP, CCDC57	"hydrolase activity, acting on esther bonds"	104	N/A	0.04
m100	**LMX1B**, C8orf34, FZD10	N/A	103	Only expressed in a subset of p5 cells.	0.68
m101	CALM3, PRKRA, ENSG00000285920	N/A	102	N/A	0.09

Table E.9 (*multi-page*)**.** Every GmGM module, in order of size. Bold genes either are, are antisense to, or differential transcripts of transcription factors. Select GO terms being N/A does not necessarily indicate a lack of GO terms, merely a lack of ones we felt worth highlighting. Bold modules were considered 'interesting'.

Module	Top 3 genes by centrality	Select GO terms	Size	Annotation	Moran's I
m102	DDX18, ENSG00000291130, **ZBTB25**	rRNA metabolic process, rRNA processing	101	N/A	0.22
m103	TMEM168, NUP50, CCDC93	N/A	100	N/A	0.11
m104	**ETV5**, LEMD1, TPRKB	negative regulation of lens fiber cell differentiation, regulation of lens fiber cell differentiation, negative regulation of ERK1 and ERK2 cascade	96	Expressed in all cells, but expression much higher in subset of p5 cells.	0.25
m105	H1-5, H1-2, H2AC8	nucleosome assembly, chromatin remodeling, protein-DNA complex organization	95	Present in all passages, but only in a subset of each.	0.55
m106	SUPT20H, ATP6V1C1, ENSG00000289349	N/A	95	High lncRNA percent.	0.06
m107	STX16, CTNND1, OBSCN	N/A	95	N/A	0.05
m108	SEC22A, TIAL1, SNHG29	nucleic acid biosynthetic process	94	N/A	0.08
m109	NUP107, DNAJB6, NBR1	N/A	91	N/A	0.08
m110	LINC02646, DIRC3, PGK1	pyruvate metabolic process, monosaccharide metabolic process, HIF1-signaling pathway, Glycolosis / Gluconeogenesis	91	Not expressed in p5s.	0.38
m111	TMTC1, ATXN1, KRAS	N/A	85	N/A	0.08

Table E.9 (*multi-page*)**.** Every GmGM module, in order of size. Bold genes either are, are antisense to, or differential transcripts of transcription factors. Select GO terms being N/A does not necessarily indicate a lack of GO terms, merely a lack of ones we felt worth highlighting. Bold modules were considered 'interesting'.

Module	Top 3 genes by centrality	Select GO terms	Size	Annotation	Moran's I
m112	KLHL7, CASTOR3P, AP2A2	N/A	85	N/A	0.09
m113	CDH18, NAV3, SIPA1L3	N/A	84	Expressed in all cells, but more so in certain subsets.	0.47
m114	MCPH1-AS1, BCL9, CRYZL1	N/A	78	N/A	0.09
m115	SMARCB1, CLEC2D, PRKD1	N/A	74	N/A	0.08
m116	SCP2, GSPT1, ZMAT1	N/A	63	N/A	0.04
m117	MT-CYB, MT-ND2, MT-ATP6	oxidative phosphorylation, ATP synthesis coupled electron transport, NADH dehydrogenase activity	48	Consists of mitochondrial genes. Expressed in all cells, less in p5s.	0.35
m118	ENSG00000227681, SAMD5, OSBPL1A	N/A	37	Expressed only in a subset of non-iPSC cells (all passages).	0.69

Table E.9 (*multi-page*)**.** Every GmGM module, in order of size. Bold genes either are, are antisense to, or differential transcripts of transcription factors (such as **LHX-DT**). Select GO terms being N/A does not necessarily indicate a lack of GO terms, merely a lack of ones we felt worth highlighting. Bold modules were considered 'interesting'.

Bibliography for Paper 5 (appendix)

Andrew, E. B., Westhead, D., & Cutillo, L. (2024). GmGM: A fast multi-axis Gaussian graphical model [ISSN: 2640-3498]. *Proceedings of The 27th International Conference on Artificial Intelligence and Statistics*, 2053–2061. Retrieved June 26, 2024, from https://proceedings.mlr.press/v238/b-andrew24a.html

Brand, M. (2006). Fast low-rank modifications of the thin singular value decomposition. *Linear Algebra and its Applications*, *415*(1), 20–30. https://doi.org/10.1016/j.laa.2005.07.021

Dahl, A., Hore, V., Iotchkova, V., & Marchini, J. (2013, December). Network inference in matrix-variate Gaussian models with non-independent noise [arXiv:1312.1622 [stat]]. https://doi.org/10.48550/arXiv.1312.1622

Greenewald, K., Zhou, S., & Hero, A., III. (2019). Tensor Graphical Lasso (TeraLasso). *Journal of the Royal Statistical Society Series B: Statistical Methodology*, *81*(5), 901–931. https://doi.org/10.1111/rssb.12339

Holbrook, A. (2018). Differentiating the pseudo determinant. *Linear Algebra and its Applications*, *548*, 293–304. https://doi.org/10.1016/j.laa.2018.03.018

Kolberg, L., Raudvere, U., Kuzmin, I., Adler, P., Vilo, J., & Peterson, H. (2023). G:Profiler—interoperable web service for functional enrichment analysis and gene identifier mapping (2023 update). *Nucleic Acids Research*, *51*(W1), W207–W212. https://doi.org/10.1093/nar/gkad347

Nene, S. A., Nayar, S. K., & Murase, H. (1996). Columbia Object Image Library (COIL-20).

Srinivasan, S., & Panda, N. (2023). What is the gradient of a scalar function of a symmetric matrix? *Indian Journal of Pure and Applied Mathematics*, *54*(3), 907–919. https://doi.org/10.1007/s13226-022-00313-x

Wolf, F. A., Angerer, P., & Theis, F. J. (2018). SCANPY: Large-scale single-cell gene expression data analysis. *Genome Biology*, *19*(1), 15. https://doi.org/10.1186/s13059-017-1382-0

Yazar, S., Alquicira-Hernandez, J., Wing, K., Senabouth, A., Gordon, M. G., Andersen, S., Lu, Q., Rowson, A., Taylor, T. R. P., Clarke, L., Maccora, K., Chen, C., Cook, A. L., Ye, C. J., Fairfax, K. A., Hewitt, A. W., & Powell, J. E. (2022). Single-cell eQTL mapping identifies cell type–specific genetic control of autoimmune disease [Publisher: American Association for the Advancement of Science]. *Science*, *376*(6589), eabf3041. https://doi.org/10.1126/science.abf3041

Appendix F

Making multi-axis models robust to multiplicative noise: how, and why?

Making Multi-Axis Models Robust to Multiplicative Noise: How, and Why?

Bailey Andrew, David R. Westhead, Luisa Cutillo

Indended for submission at NeurIPS

Abstract. In this paper we develop a graph-learning algorithm, MED-MAGMA, to fit multi-axis (Kronecker-sum-structured) models corrupted by multiplicative noise. This type of noise is natural in many application domains, such as that of single-cell RNA sequencing, in which it naturally captures technical biases of RNA sequencing platforms. Our work is evaluated against prior work on each and every public dataset in the Single Cell Expression Atlas under a certain size, demonstrating that our methodology learns networks with better local and global structure.

F.1 Introduction

This paper deals with the problem of how to make Kronecker-structured (or 'multi-axis') methods robust to a form of noise common in applied domains such as single-cell RNA sequencing data (scRNA-seq; matrix-variate data in which rows represent cells, columns represent genes, and entries represent the amount a given gene is expressed in a given cell). Multi-axis models aim to simultaneously learn matrices $\mathbf{\Psi}_{\text{rows}}, \mathbf{\Psi}_{\text{cols}}$ that encode dependency networks ('graphs') between the rows and the columns, respectively. For scRNA-seq, this would yield both cell and gene networks, useful for tasks such as cell subtyping and gene regulatory network inference.

In particular, we aim to fit models of the following form for matrix-variate data, where ζ is some function whose role is to mix the $O(d_{\text{rows}})$ row and $O(d_{\text{cols}})$ column dependencies into a network of $O(d_{\text{rows}}d_{\text{cols}})$ matrix entry dependencies.

$$\begin{aligned} \text{latent data} &\sim \mathcal{N}\left(\mathbf{0}, \zeta\left(\mathbf{\Psi}_{\text{rows}}, \mathbf{\Psi}_{\text{cols}}\right)^{-1}\right) \\ \text{observed data}_{ij} &= r_i^{\text{rows}} r_j^{\text{cols}} \left(\text{latent data}_{ij}\right) \qquad \text{(multiplicative noise)} \\ r_i^{\text{rows}}, r_j^{\text{cols}} &\sim \text{any strictly positive distribution} \end{aligned}$$

There are many methods to fit the above model without noise under a variety of choices of ζ, which we will discuss in Section F.2. However, no such method exists to fit the distributions under this noise regime. This is unfortunate, as not only is this type of noise common, but taking it into account also dramatically increases the flexibility of the model.

In scRNA-seq, this type of noise arises from two simple observations. Some genes are more likely to be measured than others due to biases such as GC-bias (Benjamini & Speed, 2012; Dohm et al., 2008), and cells have less measured gene expression than reality due to the incompleteness of the measuring process. If the true measurement of a gene g in cell c is x, but only $0 \leq r_c^{\text{cells}} \leq 1$ of the expression in that cell is collected, and gene g is r_g^{genes} times as likely to be collected than the other cells, then we should expect to observe a measurement of $r_g^{\text{genes}} r_c^{\text{cells}} x$ rather than x itself.

Even in cases where this noise type does not exist, the flexibility of models accounting for this noise is beneficial. This is because our model is a natural generalization of the class of elliptical distributions to the matrix-variate case. If a dataset follows the model below, then it has an elliptical distribution:

$$\begin{aligned} \text{latent data} &\sim \mathcal{N}\left(\mathbf{0}, \mathbf{\Omega}^{-1}\right) \\ \text{observed data} &= r \cdot (\text{latent data}) \\ r &\sim \text{any strictly positive distribution} \end{aligned}$$

Elliptical distributions encompass many other distributions, such as the normal distribution (when $r = 1$), the multivariate t distribution (when $r \sim \sqrt{\frac{\nu}{\chi^2_\nu}}$), and the multivariate Laplace distribution (when $r^2 \sim \text{Exponential}(1)$). In fact, any distribution whose density has *elliptical contours* can, under mild conditions, be expressed in this form.

Most multi-axis models either assume the dataset is Gaussian, or make the weaker 'Gaussian copula' assumption (i.e. arbitrary marginals but the variables 'interact Gaussian-ly'); the latter is often fit via the use of the nonparanormal skeptic (H. Liu et al., 2012). Gaussian copulas are restrictive in terms of the types of dependencies they allow. They do not allow 'tail dependence' - extreme events are always asymptotically uncorrelated. However, Figure F.1 implies that tail dependence *does* exist in scRNA-seq data. Elliptical distributions allow for such extremal dependencies.

Thus, it should be clear why our proposed model is useful: it is robust to a natural type of noise, and strongly increases the expressive power of multi-axis models. In this paper, we will produce an algorithm (MED-MAGMA: the **m**atrix **e**lliptical **d**istribution's **m**ulti-**a**xis **g**raphical **m**odelling **a**lgorithm) to learn $\mathbf{\Psi}_{\text{rows}}, \mathbf{\Psi}_{\text{cols}}$ under the presented model (Chapter F.3) and in Chapter F.4 comprehensively validate its performance across every sub-1000-cell dataset in the public Single Cell Expression Atlas database (Madrigal et al., 2026). In the next section we give the motivation and history behind such models.

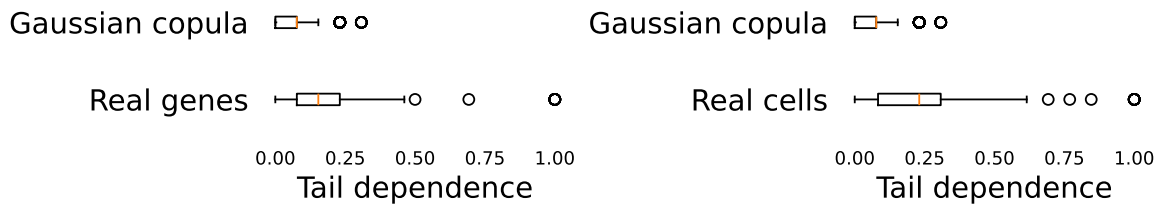


Figure F.1: Estimated tail dependence across 1000 randomly selected pairs of genes (left) and cells (right) from the E-MTAB-7249 scRNA-seq dataset discussed in Section F.4.3, compared to random data generated from a Gaussian copula.

F.2 Background

Multi-axis models were introduced to solve the problem of inference when our samples are not independent and identically distributed (i.i.d.). If one is willing to assume that the dataset $\{\mathbf{d}_i\}$ *is* i.i.d, then a simple formulation would be:

$$\mathbf{d}_i \sim \mathcal{N}\left(\mathbf{0}, \mathbf{\Psi}_{\text{cols}}^{-1}\right)$$

In this case, $\mathbf{\Psi}_{\text{cols}}$ encodes the graph of conditional dependencies:

$$\mathbf{\Psi}_{ij}^{\text{cols}} = 0 \iff \text{feature (column) } i \text{ is conditionally independent from feature } j$$

Two variables being conditionally independent means that, after conditioning out the rest of the dataset, they are statistically independent from one another. In other words, conditional dependencies intuitively capture the concept of 'direct correlations': if A is correlated with B, which in turn is correlated with C, then A and C will almost always also be correlated to some degree, via their indirect connection through B. However, A and C will not be *conditionally dependent*, as their indirect connection will have been conditioned out. This connection motivates the use of conditional dependencies: knowledge of direct effects is much more valuable than indirect effects.

For more complex datasets, the i.i.d. assumption breaks down. In particular, the 'samples' of a scRNA-seq dataset are cells of various different cell types, which had been interacting with other cells in the dataset for their entire life-cycle, until being destroyed and collected for data analysis. In this scenario, not only are the cells not independent, but the dependencies between the cells are also of interest in their own right. Compiling the dataset into a $d_{\text{rows}} \times d_{\text{cols}}$ matrix $\mathbf{D}$, we could address this by using the following distribution, where vec is the vectorization operation that stacks the columns of a matrix into a single vector:

$$\text{vec}\left[\mathbf{D}\right] \sim \mathcal{N}\left(\mathbf{0}, \mathbf{\Omega}^{-1}\right)$$

However, $\mathbf{\Omega}$ would be a $d_{\text{rows}}d_{\text{cols}} \times d_{\text{rows}}d_{\text{cols}}$ matrix capturing the dependencies between (cell, gene) pairs - it is computationally intractible to store and statistically intractible to learn. What we want is to decompose $\mathbf{\Omega}$ into tractable chunks directly representing the rows and columns: $\mathbf{\Omega} = \zeta\left(\mathbf{\Psi}_{\text{rows}}, \mathbf{\Psi}_{\text{cols}}\right)$. Rather than $O(d_{\text{rows}}^2 d_{\text{cols}}^2)$ parameters, this framework only requires $O(d_{\text{rows}}^2 + d_{\text{cols}}^2)$. There are many choices for ζ, but they are all based on the Kronecker product $\otimes$:

$$\begin{bmatrix} a_{11} & a_{12} & \cdots \\ a_{21} & \cdots & \\ \cdots & & \end{bmatrix} \otimes \mathbf{B} = \begin{bmatrix} a_{11}\mathbf{B} & a_{12}\mathbf{B} & \cdots \\ a_{21}\mathbf{B} & \cdots & \\ \cdots & & \end{bmatrix}$$

The following are the choices already considered in the literature:

$$\zeta(\mathbf{\Psi}_{\text{rows}}, \mathbf{\Psi}_{\text{cols}}) = \mathbf{\Psi}_{\text{cols}} \otimes \mathbf{I} \qquad \text{(Rows are i.i.d, same as before)}$$
$$\zeta(\mathbf{\Psi}_{\text{rows}}, \mathbf{\Psi}_{\text{cols}}) = \mathbf{I} \otimes \mathbf{\Psi}_{\text{rows}} \qquad \text{(Columns are i.i.d)}$$
$$\zeta(\mathbf{\Psi}_{\text{rows}}, \mathbf{\Psi}_{\text{cols}}) = \mathbf{\Psi}_{\text{cols}} \otimes \mathbf{\Psi}_{\text{rows}} \qquad \text{('Strong product' assumption)}$$
$$\zeta(\mathbf{\Psi}_{\text{rows}}, \mathbf{\Psi}_{\text{cols}}) = \mathbf{\Psi}_{\text{cols}} \otimes \mathbf{I} + \mathbf{I} \otimes \mathbf{\Psi}_{\text{rows}} \qquad \text{('Cartesian product' assumption)}$$
$$\zeta(\mathbf{\Psi}_{\text{rows}}, \mathbf{\Psi}_{\text{cols}}) = (\mathbf{\Psi}_{\text{cols}} \otimes \mathbf{I} + \mathbf{I} \otimes \mathbf{\Psi}_{\text{rows}})^2 \qquad \text{(Sylvester generative model)}$$

There are also models which break $\mathbf{\Omega}$ up into three axes, distinguishing between different *types* of row dependencies (B. Andrew et al., 2025). The strong product assumption is the oldest (DAWID, 1981), although it is often misreported as the 'tensor product' assumption. The Cartesian product assumption was introduced by Kalaitzis et al. (2013) and the Sylvester generative model by Y. Wang et al. (2020). For scRNA-seq data, the assumptions above correspond to the following statements:

Assumption (Strong product assumption)**.** *The strength of the dependency between the expression of a gene g_1 in a cell c_1 and the expression of a gene g_2 in a cell c_2 is equal to the product of the dependencies between c_1 and c_2, and g_1 and g_2.*

In other words, genes depend on the expression of other genes within the cell and genes in other cells.

Assumption (Cartesian product assumption)**.** *A gene g_1 may only be conditionally dependent ($\sim$) on other genes g_2 within the same cell c_1, or the same gene in other cells c_2: $(c_1, g_1) \sim (c_1, g_2)$, $(c_1, g_1) \sim (c_2, g_1)$, and $(c_1, g_1) \nsim (c_2, g_2)$. Just because they may be dependent does not mean the must be.*

In other words, genes can only directly affect the expression of other genes within the same cell (or the same gene in different cells).

Assumption (Sylvester generative model)**.**

$$\mathbf{\Psi}_{\text{rows}}\mathbf{D} + \mathbf{D}\mathbf{\Psi}_{\text{cols}} \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$$

This model naturally arises when the dataset is generated from certain partial differential equations.

For scRNA-seq data, the Cartesian product is the most natural. While genes can interact with genes in other cells in certain scenarios, such interactions will be much weaker than within-cell interactions; the strong product assumption would give equal weight to within- and between-cell interactions.

The operation $\mathbf{A} \otimes \mathbf{I} + \mathbf{I} \otimes \mathbf{B}$ is denoted the 'Kronecker sum': $\mathbf{A} \oplus \mathbf{B}$. Many algorithms exist to fit Kronecker-sum-structured normal distributions, the most scalable being GmGM E. B. Andrew et al., 2024. However, none fit such distributions under the scenario of multiplicative noise.

F.3 Methodology

To fit the model, we will perform maximum likelihood estimation (MLE) using expectation maximization (EM). Let us rephrase our model for a dataset $\mathbf{x} = \text{vec}\left[\mathbf{X}_{d_{\text{rows}} \times d_{\text{cols}}}\right]$ as follows:

$$\mathbf{x} = (\mathbf{r}_{\text{cols}} \otimes \mathbf{r}_{\text{rows}}) \circ \mathbf{z}$$
$$\mathbf{z} \sim \mathcal{N}\left(\mathbf{0}, (\mathbf{\Psi}_{\text{cols}} \oplus \mathbf{\Psi}_{\text{rows}})^{-1}\right)$$
$$\mathbf{r}_{\text{rows}}, \mathbf{r}_{\text{cols}} \sim \text{any strictly positive distribution}$$

As a shorthand, we will let $\mathbf{\Omega} = \mathbf{\Psi}_{\text{cols}} \oplus \mathbf{\Psi}_{\text{rows}}$. The $\mathbf{r}$ variables are allowed to have dependencies among themselves, but they must be independent of the latent variable $\mathbf{z}$. As further shorthand, we will implicitly let capital variables $\mathbf{V}$ and lowercase variables of the same name $\mathbf{v}$ follow the relationship $\mathbf{v} = \text{vec}[\mathbf{V}]$. Our final algorithm is summarized below. In the following sections, we will derive the algorithm step-by-step.

Algorithm 4 MED-MAGMA

Input: A $d_{\text{rows}} \times d_{\text{cols}}$ dataset $\mathbf{X}$ with vectorization $\mathbf{x}$
  $\mathbf{y} \leftarrow f(\mathbf{x})$ ▷ Remove the effect of noise (Section F.3.2)
  $\mathbf{\Psi}_{\text{rows}} \leftarrow \mathbf{I}$ ▷ Initialize estimates
  $\mathbf{\Psi}_{\text{cols}} \leftarrow \mathbf{I}$
  repeat
    Find $\mathbf{z}^*$ by flip-flopping axes while solving Equation F.3 ▷ Section F.3.5
    Approximate $\mathbf{S}_{\text{rows}}$ and $\mathbf{S}_{\text{cols}}$ using $\mathbf{\Psi}_{\text{rows}}, \mathbf{\Psi}_{\text{cols}},$ and $\mathbf{z}^*$ with Equation F.2 ▷ Section F.3.4
    $(\mathbf{\Psi}_{\text{rows}}, \mathbf{\Psi}_{\text{cols}}) \leftarrow \text{GmGM}(\mathbf{S}_{\text{rows}}, \mathbf{S}_{\text{cols}})$ ▷ Section F.3.3
  until $\mathbf{\Psi}_{\text{rows}}$ and $\mathbf{\Psi}_{\text{cols}}$ converge
Output: $\mathbf{\Psi}_{\text{rows}}$ and $\mathbf{\Psi}_{\text{cols}}$, which encode the conditional dependencies of the data.

F.3.1 Handling non-identifiability

Our model generalizes the class of elliptical distributions, and thus cannot discriminate between $\mathbf{\Omega}$ and $r\mathbf{\Omega}$ for $r > 0$. This is not a critical issue - we are only interested in the sparsity structure of $\mathbf{\Omega}$, not its scale, as it is the sparsity structure that encodes the conditional dependencies. However, it must be addressed to ensure proper behavior of our algorithm. Observe the following, for some measure of magnitude $\|\cdot\|$:

$$\mathbf{x} = (\mathbf{r}_{\text{cols}} \otimes \mathbf{r}_{\text{rows}}) \circ \mathbf{z} = r\left(\mathbf{r}'_{\text{cols}} \otimes \mathbf{r}'_{\text{rows}}\right) \circ \mathbf{z} \text{ where } \|\mathbf{r}'_{\text{cols}} \otimes \mathbf{r}'_{\text{rows}}\| = 1$$

This is why the non-identifiablilty arises: due to being free to choose r, $\mathbf{x}$ and $r\mathbf{x}$ are indistinguishable by the model, and hence so are $\mathbf{\Omega}$ and $r\mathbf{\Omega}$. As we don't care about this difference, we can assume w.l.o.g. that $r = 1$; in other words, w.l.o.g. $\|\mathbf{r}_{\text{cols}} \otimes \mathbf{r}_{\text{rows}}\| = 1$. It will be convenient later if we choose $\|\cdot\|$ to be a geometric mean constraint: $\left(\prod_i r_i^{\text{rows}}\right)^{\frac{1}{d_{\text{rows}}}} \left(\prod_j r_j^{\text{cols}}\right)^{\frac{1}{d_{\text{cols}}}} = 1$.

We also have to address the non-identifiability of Kronecker products: $\mathbf{a} \otimes \mathbf{b} = c\mathbf{a} \otimes \frac{1}{c}\mathbf{b}$. This non-identifiability is an artifact of the parameterization, and does not affect the extent to which $\mathbf{\Omega}$ can be learned. We can simply modify our constraint to get rid of this artifact: $\prod_i r_i^{\text{rows}} = 1 = \prod_j r_j^{\text{cols}}$. Thus, we aim to fit the following:

$$\begin{aligned}
\mathbf{x} &= (\mathbf{r}_{\text{cols}} \otimes \mathbf{r}_{\text{rows}}) \circ \mathbf{z} \\
\mathbf{z} &\sim \mathcal{N}\left(\mathbf{0}, (\mathbf{\Psi}_{\text{cols}} \oplus \mathbf{\Psi}_{\text{rows}})^{-1}\right) \\
\mathbf{r}_{\text{rows}}, \mathbf{r}_{\text{cols}} &\sim \text{any strictly positive distribution} \\
\text{such that } \prod_i r_i^{\text{rows}} &= 1 = \prod_j r_j^{\text{cols}}
\end{aligned}$$

F.3.2 Removing the effect of the noise

This model is a mixture of distributions (the normal distribution and any arbitrary strictly positive distribution), and thus does not directly yield a likelihood to estimate. In fact, no likelihood will be available as long as we have such a general assumption on the $\mathbf{r}$ variables. What we need is a function f such that $f(\mathbf{x}) = f(\mathbf{y}) \iff \mathbf{x} = (\mathbf{r}_{\text{cols}} \otimes \mathbf{r}_{\text{rows}}) \circ \mathbf{y}$. Such a function throws out all the information $\mathbf{x}$ provides that has become corrupted by the noise, leaving only the information upon which it is safe to do inference; we can then deduce a likelihood over f's image.

It is worth thinking about the class of d-dimensional elliptical distributions again, in which we have $\mathbf{x} = r\mathbf{z}$. The arbitrariness of r ensures we can never know the magnitude of $\mathbf{z}$, but we can know its orientation. In particular, $\frac{\mathbf{x}}{\|\mathbf{x}\|} = \frac{r\mathbf{z}}{\|r\mathbf{z}\|} = \frac{\mathbf{z}}{\|\mathbf{z}\|}$; $f(\mathbf{y}) = \frac{\mathbf{y}}{\|\mathbf{y}\|}$ maps the data to a circle. The probability distribution at any point $\mathbf{p}$ along this circle is obtained by integrating over every point $\mathbf{y}$ such that $f(\mathbf{y}) = f(\mathbf{p})$. This space of points, denoted generically $\mathbb{F}$ (and denoted $f^{-1}(\mathbf{p})$ when considering a specific circle point $\mathbf{p}$), is called the 'fiber space', and the remaining space $\mathbb{Q}$ (the circle) is the 'quotient space'; $\mathbb{R}^d \cong \mathbb{F} \times \mathbb{Q}$. We obtain the likelihood over $\mathbb{Q}$ by integrating out $\mathbb{F}$:

$$\text{pdf}\left(\mathbf{x}|\mathbf{\Omega}\right) = \int_{\mathbf{y}\in f^{-1}(\mathbf{x})} \text{pdf}_{\mathcal{N}}\left(\mathbf{y}|\mathbf{\Omega}\right) d\mathbf{y}$$

For the standard class of elliptical distributions, this integral is actually tractable, yielding the angular Gaussian distribution, whose MLE turns out to be Tyler's M estimator (Tyler, 1987). For our distribution, the scenario is significantly more complicated, as normalizing to the unit sphere does not remove the noise. Yet the procedure stays the same: we will produce a function f that throws away unreliable information while preserving everything else, and then integrate out the fibers induced by f to produce a formula for the likelihood over $\mathbb{Q}$.

$$f(\mathbf{X})_{ij} = \frac{x_{ij}\left|\prod_{i'j'} x_{i'j'}\right|^{\frac{1}{d_{\text{rows}}d_{\text{cols}}}}}{\left|\prod_{j'} x_{ij'}\right|^{\frac{1}{d_{\text{cols}}}}\left|\prod_{i'} x_{i'j}\right|^{\frac{1}{d_{\text{rows}}}}} \tag{F.1}$$

Proposition (Goodness of f)**.** *The function f described in Equation F.1 has the following property: $f(\mathbf{X}) = f(\mathbf{Y})$ if and only if one can find strictly positive $\mathbf{r}_{\text{rows}}, \mathbf{r}_{\text{cols}}$ such that we have $\text{vec}\left[\mathbf{X}\right] = \left(\mathbf{r}_{\text{cols}} \otimes \mathbf{r}_{\text{rows}}\right) \circ \text{vec}\left[\mathbf{Y}\right]$.*

Proof. See Appendix F.6. □

Handling zeros Many datasets, such as most scRNA-seq datasets, contain zeros. Equation F.1 as written is undefined for such zeros. We present a zero-handling modification of f in Appendix F.7. From here onwards, we will assume that f works with zeros.

F.3.3 Performing expectation maximization

Now that we have f, we can express the likelihood of our distribution as:

$$\text{pdf}\left(\mathbf{x}|\mathbf{\Omega}\right) = \int_{\mathbf{z}\in f^{-1}(\mathbf{x})} \text{pdf}_{\mathcal{N}}\left(\mathbf{z}|\mathbf{\Omega}\right) d\mathbf{z} \;\propto\; \sqrt{\det\mathbf{\Omega}} \int_{\mathbf{z}\in f^{-1}(\mathbf{x})} e^{-\frac{1}{2}\mathbf{z}^T\mathbf{\Omega}\mathbf{z}} d\mathbf{z}$$

Sadly, this integral is likely intractable due to the complicated structures of both f^{-1} and $\mathbf{\Omega}$. This motivates the use of the EM algorithm, frequently used in settings with latent variables, such as this. Letting $\mathcal{K}^{\oplus}_{++}$ be the space of positive definite Kronecker-sum-separable matrices (i.e. the space of valid $\mathbf{\Omega}$), the EM algorithm is as follows:

$$\begin{aligned}
\text{let } \mathbf{y} = f(\mathbf{x}) = f(\mathbf{z}) \text{ be our observed variable} \\
\text{let } Q\left(\mathbf{\Omega}\middle|\mathbf{\Omega}_{(t)}\right) &= \mathbb{E}_{\mathbf{z}\sim\mathcal{N}\left(\mathbf{0},\mathbf{\Omega}_{(t)}^{-1}\right) \text{ and } \mathbf{z}\in f^{-1}(\mathbf{y})}\left[\log \text{pdf}_{\mathcal{N}}(\mathbf{z}|\mathbf{\Omega})\right] \\
\mathbf{\Omega}_{(t+1)} &= \underset{\mathbf{\Omega}\in\mathcal{K}^{\oplus}_{++}}{\text{argmax}}\, Q\left(\mathbf{\Omega}\middle|\mathbf{\Omega}_{(t)}\right) \qquad \text{(iterate until convergence)}
\end{aligned}$$

Thus we have reduced our problem to the (admittedly intimidating) problem of maximizing Q.

Proposition (Maximizing Q)**.** *Optimizing Q corresponds to **the same optimization problem as GmGM**.*

GmGM uses the sufficient statistics $\mathbf{X}\mathbf{X}^T$ and $\mathbf{X}\mathbf{X}^T$. To maximize Q, we replace these with:

$$\mathbf{S}^{\text{rows}} = \mathbb{E}_{\substack{\mathbf{z}\sim\mathcal{N}\left(\mathbf{0},\mathbf{\Omega}_{(t)}^{-1}\right) \\ \mathbf{z}\in f^{-1}(\mathbf{y})}}\left[\mathbf{Z}\mathbf{Z}^T\right] \text{ and } \mathbf{S}^{\text{cols}} = \mathbb{E}_{\substack{\mathbf{z}\sim\mathcal{N}\left(\mathbf{0},\mathbf{\Omega}_{(t)}^{-1}\right) \\ \mathbf{z}\in f^{-1}(\mathbf{y})}}\left[\mathbf{Z}^T\mathbf{Z}\right]$$

*More generally, any method to fit unregularized Kronecker-sum-structured models **without noise** that operates on the sufficient statistics can be used in place of GmGM.*

Proof. The proof is given in Appendix F.8. □

This is excellent: GmGM is a fast algorithm. All that remains is to evaluate the expectations.

F.3.4 Evaluating $\mathbf{S}^{\text{rows}}$ and $\mathbf{S}^{\text{cols}}$ with Laplace's approximation

Let $g(\mathbf{z})$ be either $\mathbf{Z}\mathbf{Z}^T$ or $\mathbf{Z}^T\mathbf{Z}$; the logic behind evaluating the expectation is the same. We can express the expectation as follows:

$$\mathop{\mathbb{E}}_{\substack{\mathbf{z}\sim\mathcal{N}\left(\mathbf{0},\mathbf{\Omega}_{(t)}^{-1}\right)\\ \mathbf{z}\in f^{-1}(\mathbf{y})}}\left[g(\mathbf{z})\right] = \frac{\int_{\mathbf{z}\in f^{-1}(\mathbf{y})} g(\mathbf{z}) e^{-\frac{1}{2}\mathbf{z}^T\mathbf{\Omega}_{(t)}\mathbf{z}} d\mathbf{z}}{\int_{\mathbf{z}\in f^{-1}(\mathbf{y})} e^{-\frac{1}{2}\mathbf{z}^T\mathbf{\Omega}_{(t)}\mathbf{z}} d\mathbf{z}} \tag{F.2}$$

It may appear that we have swapped one hard problem for another - we were trying to avoid a nasty integral-over-fiber in Section F.3.3, yet now we have a ratio of them! However, this ratio can be well-approximated by Laplace's method (Tierney & Kadane, 1986). The intuition behind Laplace's method is that the integral is dominated by the exponential term, whose maximum occurs at $\mathbf{z}^*$ (in other words, the minimum of $\mathbf{z}^T\mathbf{\Omega}_{(t)}\mathbf{z}$ occurs at $\mathbf{z}^*$). By focusing on local approximations about $\mathbf{z}^*$, we can well-approximate the integral in the region where it has the most mass. Letting $\mathbf{P}$ be the matrix that projects onto the tangent space about $\mathbf{z}^*$ in the fiber ($\mathcal{T}_{\mathbf{z}^*} f^{-1}(\mathbf{y})$), Laplace's method tells us that:

$$\frac{\int_{\mathbf{z}\in f^{-1}(\mathbf{y})} g(\mathbf{z}) e^{-\frac{1}{2}\mathbf{z}^T\mathbf{\Omega}_{(t)}\mathbf{z}} d\mathbf{z}}{\int_{\mathbf{z}\in f^{-1}(\mathbf{y})} e^{-\frac{1}{2}\mathbf{z}^T\mathbf{\Omega}_{(t)}\mathbf{z}} d\mathbf{z}} \approx g\left(\mathbf{z}^*\right) + \frac{1}{2}\text{tr}\left[\left(\mathbf{P}\mathbf{\Omega}_{(t)}\mathbf{P}^T\right)^{-1}\mathbf{P}\left(\nabla^2 g(\mathbf{z}^*)\right)\mathbf{P}^T\right]$$

The trace term above looks intimidating, but in practice it can be evaluated efficiently; we show how in Appendix F.9. Thus, all that remains is to find the point $\mathbf{z}^* \in f^{-1}(\mathbf{y})$ that minimizes $\mathbf{z}^T\mathbf{\Omega}_{(t)}\mathbf{z}$.

F.3.5 Finding $\mathbf{z}^*$ with quadratic programming

In order to approximate $\mathbf{S}^{\text{rows}}$ and $\mathbf{S}^{\text{cols}}$ statistics, we need to solve the following optimization problem:

$$\begin{aligned}\mathbf{z}^* &= \mathop{\text{argmin}}_{\mathbf{z}\in f^{-1}(\mathbf{y})} \mathbf{z}^T\mathbf{\Omega}_{(t)}\mathbf{z} \\ &= \mathop{\text{argmin}}_{\substack{\mathbf{r}_{\text{rows}},\mathbf{r}_{\text{cols}}>0\\ \prod_i r_i^{\text{rows}} = 1 = \prod_j r_j^{\text{cols}}}} \sum_{ik} r_i^{\text{rows}}\left(\sum_{j\ell} r_j^{\text{cols}} Y_{ij}\Omega^{(t)}_{(ij),(k\ell)} r_\ell^{\text{cols}} Y_{k\ell}\right) r_k^{\text{rows}}\end{aligned}$$

This is equivalent to minimizing an order-4 polynomial with constraints - not a trivial task at all. However, our problem naturally lends itself to a flip-flop algorithm over two quadratic sub-problems; the terms in the parentheses may be aggregated into a $d_{\text{rows}} \times d_{\text{rows}}$ matrix $\tilde{\mathbf{\Omega}}$ not depending on $\mathbf{r}_{\text{rows}}$. This is a quadratic form in $\mathbf{r}_{\text{rows}}$, and so for fixed $\mathbf{r}_{\text{cols}}$ we can reduce our problem:

$$\text{argmin}\, \mathbf{r}_{\text{rows}}^T\tilde{\mathbf{\Omega}}\mathbf{r}_{\text{rows}} \text{ such that } \mathbf{r}_{\text{rows}} > 0 \text{ and } \prod_i r_i^{\text{rows}} = 1 \tag{F.3}$$

This is still non-convex due to the nonlinear multiply-to-1 constraint, but in practice is efficiently optimizable. An analogous formulation holds for $\mathbf{r}_{\text{cols}}$; we can flip-flop between them to find the optimal $\mathbf{r}_{\text{rows}}, \mathbf{r}_{\text{cols}}$, and hence $\mathbf{z}^*$.

F.4 Results

In this section, we evaluate our method, MED-MAGMA, in three ways: by comparing it to GmGM on synthetic data (Section F.4.1), comparing it to GmGM systematically across all public datasets of a certain type (Section F.4.2), and finally by exploring a specific example (Section F.4.3). The code to run these experiments is available on GitHub (TODO) and our model is available on PyPI (TODO).

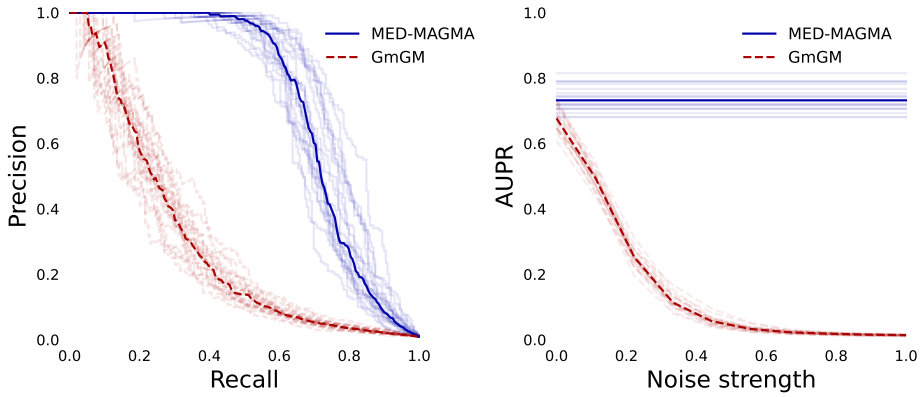


Figure F.2: (Both) Results across 20 synthetic Kronecker-sum-structured Gaussian datasets corrupted by varying levels of multiplicative noise. Median result is highlighted. (Left) Precision/recall (PR) curves for noise strength 0.2. (Right) Area under PR curves as noise strength varies.

F.4.1 Synthetic validation

To validate our method on synthetic data, we first generated ground truth graphs from the Barabasi-Albert distribution, and converted these to positive definite matrices $\mathbf{\Psi}^{\text{rows}}$ and $\mathbf{\Psi}^{\text{cols}}$. We then generated 20 samples from $\mathcal{N}\left(\mathbf{0}, (\mathbf{\Psi}_{\text{cols}} \oplus \mathbf{\Psi}_{\text{rows}})^{-1}\right)$. To corrupt the samples with multiplicative noise, we generated each $\mathbf{r}_{(\ell)}$ variable as chi-squared (df=1) and raised them to the power $\alpha \in [0, 1]$; α represents the noise strength ($\alpha = 0$ yields no noise).

We compare MED-MAGMA and GmGM on this synthetic dataset in Figure F.2. We can see that MED-MAGMA is unaffected by noise strength, whereas GmGM is quite severely affected.

F.4.2 Systematic validation

To validate MED-MAGMA on real data, we systematically downloaded every dataset in the Single Cell Expression Atlas less than a thousand cells which contained a categorical variable and referenced a source publication. There were 30 such datasets in total; we list these and the results of our algorithm on them in Appendix F.11. By evaluating our methodology in this manner, we ensure that we evaluate over a representative sample of all scRNA-seq datasets.

The datasets received minimal preprocessing, described in Appendix F.10. Each dataset contained a categorical variable, typically 'cell type'. To evaluate MED-MAGMA and GmGM, we pruned their output graphs to keep only the top 10 edges per vertex. For comparison, we used both assortativity and adjusted mutual information (AMI). Assortativity measures the tendency of cells of the same category to be connected in the learned graphs, whereas AMI measures the tendency of a clustering to align with the categorical variable. We used Leiden clustering (Traag et al., 2019), varying the resolution parameter from 0.02 to 2.0 to obtain clusterings of various sizes - we only report the best AMI across all clusterings. We compared against both standard GmGM and GmGM equipped with the nonparanormal skeptic; Figure F.3 demonstrates that assortativity and AMI are higher in MED-MAGMA than GmGM, with or without the nonparanormal skeptic.

F.4.3 Specific evaluation

Both prior sections serve to show that MED-MAGMA yields better results than GmGM. Thus, models that take into account multiplicative noise perform better on scRNA-seq data. In this section, we briefly pick one of the datasets from earlier and show that the cell and gene networks are robust, and that the gene network learned is also reasonable. In particular, we pick the 'median dataset' when ranking datasets by assortativity. By picking the median dataset, it gives us a generic view into what we can expect the average performance of our methodology to be.

The chosen dataseet, E-MTAB-7249, contains 185 cells from a patient with high-grade serous ovarian carcinoma (HGSOC) (Nelson et al., 2020). The source paper investigated the biology of HGSOC through several modalities of data from multiple patients. Here, we are using the portion of this dataset

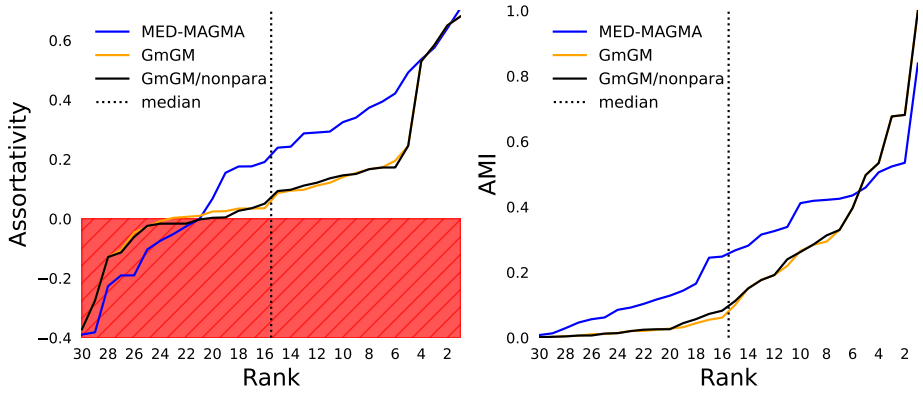


Figure F.3: (Both) The datasets are ordered by increasing performance along the x-axis. GmGM/nonpara refers to GmGM equipped with the nonparanormal skeptic. The median performance is highlighted. (Left) Assortativities below 0 represent a statistical tendency for cells of *different* categories to connect, which indicates the network does capture information about the dataset, but it is harder to interpret.

available on the Single Cell Expression Atlas, comprised only of the cells of patient 'OCM.38a'. The data from the atlas had 7 clusters representing cell subtypes: SA-a, SA-b, SB, TA-a, TA-b, TB, and a final small 'N/A' cluster of unlabeled cells. From the source paper, it is clear that these are subtypes of the following cell types: stromal (SA-a, SA-b, SB) and neoplastic (TA-a, TA-b, TB).

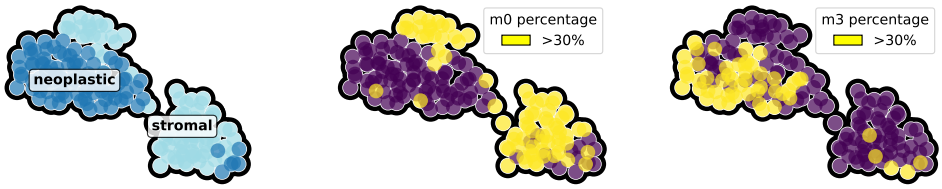


Figure F.4: (All) A UMAP plot of E-MTAB-7249 cells based on gene expression. (Left) Cell type. (Middle, Right) The percentage of gene expression from genes in modules m0 and m3, respectively.

We clustered the learned gene network (Leiden, resolution=1), yielding 8 'modules'. Figure F.4 shows that two of our gene modules strongly correspond to stromal and neoplastic cells. To quantify the relation between our gene modules and the cell subtypes, we performed a one-way ANOVA testing whether the percentage expression of each gene module differed across cell subtypes. All modules showed significant variation between clusters ($p = 2 \times 10^{-29}$ to 1×10^{-2}) except one ($p = 0.11$), indicating that MED-MAGMA learned gene modules capturing processes associated with specific cell subtypes.

For one final test, we evaluated the robustness of our cell and gene networks using Robin (Policastro et al., 2021). Robin measures the tendency for clusterings on a network to change after an increasing proportion of the edges are randomly perturbed: networks whose clusterings change significantly after a small clustering are not robust. Robin outputs Bayes factors; our cell network has factor of 108 and our gene network has a factor of 59, indicating very strong evidence that clusterings on MED-MAGMA's networks are robust to perturbations.

F.5 Discussion and limitations

We have developed a model that allows multi-axis estimation under multiplicative noise, and an algorithm to fit the model. Such a model is important; many datasets, such as scRNA-seq datasets, are affected by this type of noise. It also significantly increases the flexibility of multi-axis models; we did not make a Gaussian copula assumption, allowing our model to be suitable for datasets that contain more complicated modes of dependency (such as tail dependence). We demonstrated that our methodology outperforms prior work on synthetic data as well as through a systematic evaluation of all public scRNA-seq datasets below a certain size on a public database.

Despite the promising results, our model does have limitations. Currently, our method can handle matrices with thousands of rows/columns, but would struggle with tens of thousands. Both the approximation of $(\mathbf{S}_{\text{rows}}, \mathbf{S}_{\text{cols}})$ and the GmGM sub-problem dominate the runtime, so improvements in either of those would be significant. Additionally, elliptical-style distributions such as ours (despite their generality) still make some tangible assumptions about the dependencies in the dataset - namely that dependencies are linear and radially symmetric. It would be a worthwhile future direction to further weaken the assumptions made here.

F.6 Proof that f removes only the corrupted directions

In this section, we will demonstrate that our definition of f (Equation F.1) removes all directions corrupted by noise from our data, and nothing else.

Proposition (Goodness of f)**.** *The function f described in Equation F.1 has the following property: $f(\mathbf{X}) = f(\mathbf{Y})$ if and only if one can find strictly positive $\mathbf{r}_{\text{rows}}, \mathbf{r}_{\text{cols}}$ such that we have $\text{vec}[\mathbf{X}] = (\mathbf{r}_{\text{cols}} \otimes \mathbf{r}_{\text{rows}}) \circ \text{vec}[\mathbf{Y}]$.*

Proof. Recall the following direction of f:

$$f(\mathbf{X})_{ij} = \frac{x_{ij} \left| \prod_{i'j'} x_{i'j'} \right|^{\frac{1}{d_{\text{rows}} d_{\text{cols}}}}}{\left| \prod_{j'} x_{ij'} \right|^{\frac{1}{d_{\text{cols}}}} \left| \prod_{i'} x_{i'j} \right|^{\frac{1}{d_{\text{rows}}}}} \qquad \text{(Equation F.1)}$$

Note that this has the following equivalent, more complicated formulation:

$$f(\mathbf{X})_{ij} = \text{sgn}\left(x_{ij}\right) \exp\Bigg(\log\left|x_{ij}\right| + \frac{1}{d_{\text{rows}} d_{\text{cols}}} \sum_{i'j'} \log\left|x_{i'j'}\right| - \frac{1}{d_{\text{rows}}} \sum_{i'} \log\left|x_{i'j}\right| - \frac{1}{d_{\text{cols}}} \sum_{j'} \log\left|x_{ij'}\right| \Bigg)$$

Letting $\mathbf{L} = \log^{\circ}|\mathbf{X}|$ be the elementwise logarithm of the absolute values of $\mathbf{X}$, we can see that the core of our function is ultimately just a linear function on $\mathbf{L}$, $\mathcal{L}$:

$$\mathcal{L}(\mathbf{L})_{ij} = \ell_{ij} + \text{mean}(\mathbf{L}) - \text{row means}\left(\mathbf{L}\right)_i - \text{col means}\left(\mathbf{L}\right)_j$$

This is a well-known function (double-centering of a matrix) and it is easy to express it in matrix form:

$$\begin{aligned}
\mathcal{L}(\mathbf{L}) &= \mathbf{L} + \frac{\mathbf{1}^T\mathbf{L}\mathbf{1}}{d_{\text{cols}} d_{\text{rows}}} \mathbf{1}\mathbf{1}^T - \left(\frac{1}{d_{\text{cols}}} \mathbf{L}\mathbf{1}\right) \mathbf{1}^T - \mathbf{1}\left(\frac{1}{d_{\text{rows}}} \mathbf{1}^T\mathbf{L}\right) \\
&= \mathbf{L} - \mathbf{L}\left(\frac{1}{d_{\text{cols}}} \mathbf{1}\mathbf{1}^T\right) - \left(\frac{1}{d_{\text{rows}}} \mathbf{1}\mathbf{1}^T\right)\mathbf{L} + \left(\frac{1}{d_{\text{rows}}} \mathbf{1}\mathbf{1}^T\right)\mathbf{L}\left(\frac{1}{d_{\text{cols}}} \mathbf{1}\mathbf{1}^T\right) \\
&= \mathbf{L}\left(\mathbf{I} - \frac{1}{d_{\text{cols}}} \mathbf{1}\mathbf{1}^T\right) + \left(\frac{1}{d_{\text{rows}}} \mathbf{1}\mathbf{1}^T\right)\mathbf{L}\left(\mathbf{I} - \frac{1}{d_{\text{cols}}} \mathbf{1}\mathbf{1}^T\right) \\
&= \left(\mathbf{I} - \frac{1}{d_{\text{rows}}} \mathbf{1}\mathbf{1}^T\right)\mathbf{L}\left(\mathbf{I} - \frac{1}{d_{\text{cols}}} \mathbf{1}\mathbf{1}^T\right)
\end{aligned}$$

In other words:

$$f(\mathbf{X}) = \operatorname{sgn}(\mathbf{X}) \circ \exp^{\circ}\left(\left[\mathbf{I} - \frac{1}{d_{\text{rows}}}\mathbf{1}\mathbf{1}^T\right] \log^{\circ}|\mathbf{X}| \left[\mathbf{I} - \frac{1}{d_{\text{cols}}}\mathbf{1}\mathbf{1}^T\right]\right)$$

$$f(\mathbf{x}) = \operatorname{sgn}(\mathbf{x}) \circ \exp^{\circ}\left(\left(\left[\mathbf{I} - \frac{1}{d_{\text{cols}}}\mathbf{1}\mathbf{1}^T\right] \otimes \left[\mathbf{I} - \frac{1}{d_{\text{rows}}}\mathbf{1}\mathbf{1}^T\right]\right) \log^{\circ}|\mathbf{x}|\right) \qquad (\text{where } \mathbf{x} = \operatorname{vec}[\mathbf{X}])$$

The benefit of expressing our function in this form is that we can fully describe what information is lost and kept by the eigenvectors of $\left[\mathbf{I} - \frac{1}{d_{\text{cols}}}\mathbf{1}\mathbf{1}^T\right] \otimes \left[\mathbf{I} - \frac{1}{d_{\text{rows}}}\mathbf{1}\mathbf{1}^T\right]$. This is the Kronecker product of two centering matrices - the $n \times n$ centering matrix has a single zero eigenvalue, with the remaining $n-1$ eigenvalues being 1. The eigenvalues of a Kronecker product are all the products of the eigenvalues, giving $d_{\text{rows}} + d_{\text{cols}} - 1$ zero eigenvalues (the rest being one). If we can show that our noise gets mapped to zero and spans a $(d_{\text{rows}} + d_{\text{cols}} - 1)$-dimensional subspace, we will have proven that our function works.

$$\log^{\circ}|r(\mathbf{r}_{\text{cols}} \otimes \mathbf{r}_{\text{rows}}) \circ \mathbf{x}| = \log^{\circ}\mathbf{r}_{\text{cols}} \oplus \log^{\circ}\mathbf{r}_{\text{rows}} + \log^{\circ}|\mathbf{x}| + \log(r)\mathbf{1} \qquad (\text{Noise is linear in logspace})$$

$$\begin{aligned}\left(\left[\mathbf{I} - \frac{1}{d_{\text{cols}}}\mathbf{1}\mathbf{1}^T\right] \otimes \left[\mathbf{I} - \frac{1}{d_{\text{rows}}}\mathbf{1}\mathbf{1}^T\right]\right)&\left(\log^{\circ}\mathbf{r}_{\text{cols}} \oplus \log^{\circ}\mathbf{r}_{\text{rows}} + \log(r)\mathbf{1}\right)\\ &= \left(\left[\mathbf{I} - \frac{1}{d_{\text{cols}}}\mathbf{1}\mathbf{1}^T\right] \log^{\circ}\mathbf{r}_{\text{cols}}\right) \otimes \left[\mathbf{I} - \frac{1}{d_{\text{rows}}}\mathbf{1}\mathbf{1}^T\right]\\ &+ \left[\mathbf{I} - \frac{1}{d_{\text{cols}}}\mathbf{1}\mathbf{1}^T\right] \otimes \left(\left[\mathbf{I} - \frac{1}{d_{\text{rows}}}\mathbf{1}\mathbf{1}^T\right] \log^{\circ}\mathbf{r}_{\text{rows}}\right)\\ &+ \log(r)\left[\mathbf{I} - \frac{1}{d_{\text{cols}}}\mathbf{1}\mathbf{1}^T\right]\mathbf{1} \otimes \left[\mathbf{I} - \frac{1}{d_{\text{rows}}}\mathbf{1}\mathbf{1}^T\right]\mathbf{1}\end{aligned}$$

Our constraint that $\prod_i \mathbf{r}_i^{(\ell)} = 1$ is equivalent to $\sum_i \log \mathbf{r}_i^{(\ell)} = 0$. In other words, in logspace our vectors have mean zero, and are thus annihilated by the centering matrix. The whole equation then becomes zero: our noise is indeed removed by our function.

Finally, note that we parameterize $r, \mathbf{r}_{\text{rows}}$ and $\mathbf{r}_{\text{cols}}$ with $1 + d_{\text{rows}} + d_{\text{cols}}$ parameters. However, our two geometric mean constraints further decrease the dimensionality; our noise is parameterized by $d_{\text{rows}} + d_{\text{cols}} - 1$ parameters. This is exactly the size of the nullspace of our double-centering matrix, guaranteeing that only noise-corrupted directions are removed from $\mathbf{x}$. This completes the proof

□

F.7 A version of f that handles zero values

When the input $\mathbf{X}$ contains zeros, the function f breaks - either one must divide by zero or take the logarithm of it. However, zeros are unaffected by multiplicative noise; intuitively, they should not cause such a catastrophic failure. We use the obvious modification (with Figure F.5 demonstrating the performance): we will skip over the zero entries when doing products. Let nnz be a function that counts the number of nonzero entries in a vector, and $\prod^{\neq 0}$ be a product over only nonzero terms; we have the following:

$$f(\mathbf{X})_{ij} = \frac{x_{ij}\left|\prod_{i'j'}^{\neq 0} x_{i'j'}\right|^{\frac{1}{\text{nnz}(\mathbf{x})}}}{\left|\prod_{j'}^{\neq 0} x_{ij'}\right|^{\frac{1}{\text{nnz}(\mathbf{x}_{\text{row } i})}}\left|\prod_{i'}^{\neq 0} x_{i'j}\right|^{\frac{1}{\text{nnz}(\mathbf{x}_{\text{col } j})}}}$$

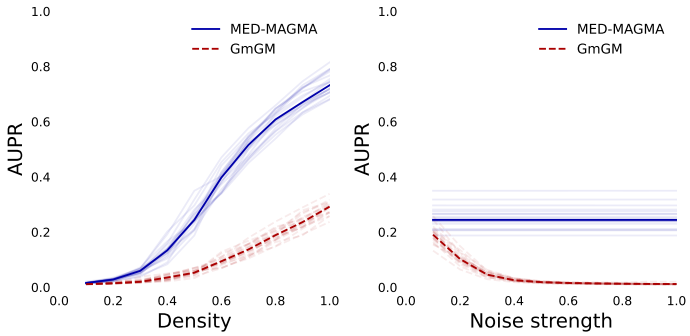


Figure F.5: (Both) Area under PR curves across 20 synthetic Kronecker-sum-structured Gaussian datasets corrupted by varying levels of multiplicative noise and sparsity. Median result is highlighted. (Left) As percentage of nonzero entries vary (noise strength 0.2) (Right) Area under PR curves as noise strength varies (density 0.5).

F.8 Proof that optimizing Q is equivalent to GmGM

Recall the problem of optimizing Q:

$$\text{let } Q\left(\boldsymbol{\Omega}\middle|\boldsymbol{\Omega}_{(t)}\right) = \mathbb{E}_{\mathbf{z}\sim\mathcal{N}\left(\mathbf{0},\boldsymbol{\Omega}_{(t)}^{-1}\right) \text{ and } \mathbf{z}\in f^{-1}(\mathbf{y})}\left[\log \text{pdf}_{\mathcal{N}}(\mathbf{z}|\boldsymbol{\Omega})\right]$$

$$\boldsymbol{\Omega}_{(t+1)} = \underset{\boldsymbol{\Omega}\in\mathcal{K}_{++}^{\oplus}}{\text{argmax}}\, Q\left(\boldsymbol{\Omega}\middle|\boldsymbol{\Omega}_{(t)}\right) \qquad \text{(iterate until convergence)}$$

As claimed in the main paper, maximization of Q requires solving the same optimization problem as GmGM. Below, let $\mathbf{J}^{ij}$ be the matrix of zeros except for the (i,j) entry being one.

Definition (GmGM (E. B. Andrew et al., 2024))**.** *GmGM solves the convex minimization problem of minimizing the negative log likelihood, with gradients:*

$$-\nabla_{\boldsymbol{\Psi}_{\text{rows}}} \log \text{pdf}_{\mathcal{N}}(\mathbf{z}|\boldsymbol{\Omega}) = \mathbf{X}\mathbf{X}^T - \text{tr}\left[\boldsymbol{\Omega}^{-1}\left(\mathbf{J}^{ij}\otimes\mathbf{I}\right)\right]_{ij}$$

$$-\nabla_{\boldsymbol{\Psi}_{\text{cols}}} \log \text{pdf}_{\mathcal{N}}(\mathbf{z}|\boldsymbol{\Omega}) = \mathbf{X}^T\mathbf{X} - \text{tr}\left[\boldsymbol{\Omega}^{-1}\left(\mathbf{I}\otimes\mathbf{J}^{ij}\right)\right]_{ij}$$

It only relies on the data through the terms $\mathbf{X}\mathbf{X}^T$ and $\mathbf{X}^T\mathbf{X}$; we will denote minimum point as GmGM $\left(\mathbf{X}\mathbf{X}^T, \mathbf{X}^T\mathbf{X}\right)$.

Proposition (Maximizing Q)**.** *Optimizing Q corresponds to* ***the same optimization problem as GmGM****.*

GmGM uses the sufficient statistics $\mathbf{X}\mathbf{X}^T$ and $\mathbf{X}\mathbf{X}^T$. To maximize Q, we replace these with:

$$\mathbf{S}^{\text{rows}} = \mathbb{E}_{\substack{\mathbf{z}\sim\mathcal{N}\left(\mathbf{0},\boldsymbol{\Omega}_{(t)}^{-1}\right)\\ \mathbf{z}\in f^{-1}(\mathbf{y})}}\left[\mathbf{Z}\mathbf{Z}^T\right] \text{ and } \mathbf{S}^{\text{cols}} = \mathbb{E}_{\substack{\mathbf{z}\sim\mathcal{N}\left(\mathbf{0},\boldsymbol{\Omega}_{(t)}^{-1}\right)\\ \mathbf{z}\in f^{-1}(\mathbf{y})}}\left[\mathbf{Z}^T\mathbf{Z}\right]$$

More generally, any method to fit unregularized Kronecker-sum-structured models ***without noise*** *that operates on the sufficient statistics can be used in place of GmGM.*

Proof. We will first show that the gradient w.r.t. $\boldsymbol{\Psi}_{\text{rows}}$ yields a very similar gradient to GmGM; the argument is analogous for $\boldsymbol{\Psi}_{\text{cols}}$. Note also that Q is being maximized, whereas GmGM frames the problem as a minimization problem, so we expect the gradients to have opposite signs.

$$\begin{aligned}
\nabla_{\boldsymbol{\Psi}_{\text{rows}}} Q\left(\boldsymbol{\Omega}\middle|\boldsymbol{\Omega}_{(t)}\right) &= \nabla_{\boldsymbol{\Psi}_{\text{rows}}} \mathbb{E}_{\mathbf{z}\sim\mathcal{N}\left(\mathbf{0},\boldsymbol{\Omega}_{(t)}^{-1}\right) \text{ and } \mathbf{z}\in f^{-1}(\mathbf{y})}\left[\log \text{pdf}_{\mathcal{N}}(\mathbf{z}|\boldsymbol{\Omega})\right] \\
&= \mathbb{E}_{\mathbf{z}\sim\mathcal{N}\left(\mathbf{0},\boldsymbol{\Omega}_{(t)}^{-1}\right) \text{ and } \mathbf{z}\in f^{-1}(\mathbf{y})}\left[\nabla_{\boldsymbol{\Psi}_{\text{rows}}} \log \text{pdf}_{\mathcal{N}}(\mathbf{z}|\boldsymbol{\Omega})\right] \\
&\qquad\qquad \text{(By dominated convergence theorem)} \\
&= \mathbb{E}_{\mathbf{z}\sim\mathcal{N}\left(\mathbf{0},\boldsymbol{\Omega}_{(t)}^{-1}\right) \text{ and } \mathbf{z}\in f^{-1}(\mathbf{y})}\left[\text{tr}\left[\boldsymbol{\Omega}^{-1}\left(\mathbf{J}^{ij}\otimes\mathbf{I}\right)\right]_{ij} - \mathbf{Z}\mathbf{Z}^T\right] \quad \text{(GmGM definition)} \\
&= \text{tr}\left[\boldsymbol{\Omega}^{-1}\left(\mathbf{J}^{ij}\otimes\mathbf{I}\right)\right]_{ij} - \mathbb{E}_{\mathbf{z}\sim\mathcal{N}\left(\mathbf{0},\boldsymbol{\Omega}_{(t)}^{-1}\right) \text{ and } \mathbf{z}\in f^{-1}(\mathbf{y})}\left[\mathbf{Z}\mathbf{Z}^T\right] \\
&= \text{tr}\left[\boldsymbol{\Omega}^{-1}\left(\mathbf{J}^{ij}\otimes\mathbf{I}\right)\right]_{ij} - \mathbf{S}_{\text{rows}} \qquad (\mathbf{S}_{\text{rows}} \text{ definition})
\end{aligned}$$

It should now be clear by inspection that the maximum of Q corresponds with $\text{GmGM}\left(\mathbf{S}_{\text{rows}}, \mathbf{S}_{\text{cols}}\right)$. This completes the proof.

□

F.9 Efficient evaluation of $\frac{1}{2}\mathrm{tr}\left[\left(\mathbf{P}\boldsymbol{\Omega}_{(t)}\mathbf{P}^T\right)^{-1}\mathbf{P}\left(\nabla^2 g(\mathbf{z}^*)\right)\mathbf{P}^T\right]$

We will assume $g(\mathbf{z}) = \mathbf{Z}\mathbf{Z}^T$, as $g(\mathbf{z}) = \mathbf{Z}^T\mathbf{Z}$ follows analogously. As in the previous section, let $\mathbf{J}_{(ij)}$ be the matrix of zeros except for the (i,j) entry being one. Let us start by deriving the Hessian of g, by investigating the directional derivative:

$$\begin{aligned}
\left(\nabla_{\mathbf{J}_{(k\ell)}}\nabla_{\mathbf{J}_{(ij)}}\right)\mathbf{Z}\mathbf{Z}^T &= \nabla_{\mathbf{J}_{(k\ell)}}\left(\mathbf{Z}\mathbf{J}_{(ij)}^T + \mathbf{J}_{(ij)}\mathbf{Z}^T\right) \\
&= \mathbf{J}_{(k\ell)}\mathbf{J}_{(ij)}^T + \mathbf{J}_{(ij)}\mathbf{J}_{(k\ell)}^T \\
&= \mathbf{J}_{(k\ell)}\mathbf{J}_{(ji)} + \mathbf{J}_{(ij)}\mathbf{J}_{(\ell k)} \\
&= \delta_{j\ell}\left(\mathbf{J}_{(ki)} + \mathbf{J}_{(ik)}\right) \\
\left[\left(\nabla_{\mathbf{J}_{(k\ell)}}\nabla_{\mathbf{J}_{(ij)}}\right)\mathbf{Z}\mathbf{Z}^T\right]_{ab} &= \delta_{j\ell}\left(\delta_{ak}\delta_{bi} + \delta_{ai}\delta_{bk}\right) \\
\nabla^2\text{vec}\left[\mathbf{Z}\mathbf{Z}^{\mathbf{T}}\right] &= \mathbf{I}_{d_{\text{cols}}}\otimes\left(\mathbf{J}_{d_{\text{rows}}}^{(ab)} + \mathbf{J}_{d_{\text{rows}}}^{(ba)}\right)
\end{aligned}$$

We will now work out the matrix $\mathbf{P}$ which projects onto the tangent space of the fiber. As we have seen in Section F.6, the fiber is linear-in-logspace with $d_{\text{rows}} + d_{\text{cols}} - 1$ basis vectors of the form $\mathbf{v}_{\text{cols}} \oplus \mathbf{v}_{\text{rows}}$ in which $\mathbf{1}^T\mathbf{v}_{\text{rows}} = 0 = \mathbf{1}^T\mathbf{v}_{\text{cols}}$. Note then that:

$$\begin{aligned}
\begin{bmatrix}\frac{1}{d_{\text{rows}}}\mathbf{I}\otimes\mathbf{1}^T \\ \frac{1}{d_{\text{cols}}}\mathbf{1}^T\otimes\mathbf{I}\end{bmatrix}\left(\mathbf{v}_{\text{cols}}\oplus\mathbf{v}_{\text{rows}}\right) &= \begin{bmatrix}\mathbf{v}_{\text{cols}} + \frac{\mathbf{1}^T\mathbf{v}_{\text{rows}}}{d_{\text{rows}}}\mathbf{1} \\ d_{\text{rows}}\mathbf{v}_{\text{rows}} + \frac{\mathbf{1}^T\mathbf{v}_{\text{cols}}}{d_{\text{cols}}}\mathbf{1}\end{bmatrix} \\
&= \begin{bmatrix}\mathbf{v}_{\text{cols}} \\ \mathbf{v}_{\text{rows}}\end{bmatrix}
\end{aligned}$$

Thus $\mathbf{P} = \begin{bmatrix}\frac{1}{d_{\text{rows}}}\mathbf{I}\otimes\mathbf{1}^T \\ \frac{1}{d_{\text{cols}}}\mathbf{1}^T\otimes\mathbf{I}\end{bmatrix}$ projects onto a $d_{\text{rows}} + d_{\text{cols}}$ space, from which the tangent space can be reached by just dropping the last row from the matrix. It should be clear that $\mathbf{P}\nabla^2 g(\mathbf{z})\mathbf{P}^T$ is easy, as all matrices involved are conformable block matrices of Kronecker products. Note that the $\frac{1}{d_{\text{cols}}}$ and $\frac{1}{d_{\text{rows}}}$ factors cancel out in the trace expression, so we will drop them: $\mathbf{P} = \begin{bmatrix}\mathbf{I}\otimes\mathbf{1}^T \\ \mathbf{1}^T\otimes\mathbf{I}\end{bmatrix}$.

All that remains is to efficiently find the inverse of $\mathbf{P}\boldsymbol{\Omega}_{(t)}\mathbf{P}^T$. Note the following:

$$\begin{aligned}\mathbf{P}\boldsymbol{\Omega}_{(t)}\mathbf{P}^T &= \begin{bmatrix}\boldsymbol{\Psi}_{\text{cols}} \otimes \mathbf{1}^T + \mathbf{I} \otimes \mathbf{1}^T\boldsymbol{\Psi}_{\text{rows}} \\ \mathbf{1}^T\boldsymbol{\Psi}_{\text{cols}} \otimes \mathbf{I} + \mathbf{1}^T \otimes \boldsymbol{\Psi}_{\text{rows}}\end{bmatrix}\mathbf{P}^T \\ &= \begin{bmatrix} d_{\text{rows}}\boldsymbol{\Psi}_{\text{cols}} + \left(\mathbf{1}^T\boldsymbol{\Psi}_{\text{rows}}\mathbf{1}\right)\mathbf{I} & \boldsymbol{\Psi}_{\text{cols}}\mathbf{1} \otimes \mathbf{1}^T + \mathbf{1} \otimes \mathbf{1}^T\boldsymbol{\Psi}_{\text{rows}} \\ \mathbf{1}^T\boldsymbol{\Psi}_{\text{cols}} \otimes \mathbf{1} + \mathbf{1}^T \otimes \boldsymbol{\Psi}_{\text{rows}}\mathbf{1} & d_{\text{cols}}\boldsymbol{\Psi}_{\text{rows}} + \left(\mathbf{1}^T\boldsymbol{\Psi}_{\text{cols}}\mathbf{1}\right)\mathbf{I}\end{bmatrix}\end{aligned}$$

Removing the last row of $\mathbf{P}$ amounts to removing the last row and column of this matrix. It can easily be inverted by the block matrix inversion formula, requiring inversions of only $d_\ell \times d_\ell$ matrices rather than a $d_{\text{rows}}d_{\text{cols}} \times d_{\text{rows}}d_{\text{cols}}$ matrix. Thus, our trace term can be evaluated efficiently. From this point on, all steps require standard (if tedious) linear algebra, which we leave to the reader. Our code contains a working implementation for those who wish to reference it.

F.10 Description of the systematic approach to preprocessing and evaluating datasets

As mentioned in the main paper, we systematically downloaded every dataset in the Single Cell Expression Atlas, subject to three criteria (below). One of the main difficulties in evaluating methods is the ease in which authors can (intentionally or unintentionally) end up cherry-picking examples that make their model look good. By being systematic, we can guarantee that our results represent the typical outputs of our model.

Criterion 1. The dataset must be at most 1000 cells large. We chose this threshold as it was large enough to include many datasets (40 before applying the remaining criteria) while still including only datasets small enough for our algorithm to run quickly on. It is also a nice round number. The next 'nice round number', in our opinion, is 5000 - which would be 63 datasets. As our algorithm scales cubically, the largest datasets would take 125 times as long as before.

Criterion 2. The dataset must contain at least one categorical variable. This is necessary to have a ground truth to compare against for AMI and assortativity. When there were multiple, we picked the first one we noticed; these are given in Appendix F.11.

Criterion 3. The dataset must clearly reference a source paper in the atlas. This rule was for our convenience; we did not want to waste time hunting down a paper to cite for the dataset.

30 datasets satisfied all criteria. We then had to design a standard preprocessing regime that worked for all datasets, choosing to do the minimal preprocessing necessary to get the algorithm to run. As such, our only requirement was to subset the total number of genes in each dataset; by default, they all had tens of thousands. We chose to pick the genes so that each dataset became roughly square, to avoid potentially introducing 'extreme lopsidedness' as a confounding factor. Our process was as follows:

Step 1. Select the top 2000 most variable genes. This is a very common default preprocessing step and threshold.

Step 2. Iterate from $s = 1\%$ to $s = 100\%$ in increments of 1%. For each value of s, select only genes with at least s nonzero values. Whichever value of s leads to the most square matrix, pick those genes. We chose to pick genes based on sparsity to mirror another standard preprocessing step (throw out 'low-quality' genes that do not appear much), and we quantized our values of s to ensure that not all of our matrices were *perfectly* square (which might also confound the results).

Dataset	Categorical Variable	Categories	Cells	Genes
E-MTAB-4850 (Eltahla et al., 2016)	Sample Characteristic[cell line]	2	60	63
E-GEOD-75367 (Jordan et al., 2016)	Sample Characteristic[facs marker]	4	72	70
E-MTAB-6142 (Karlsson et al., 2017)	Sample Characteristic[cell cycle phase]	3	96	91
E-GEOD-36552 (Yan et al., 2013)	Sample Characteristic[cell type]	6	115	127
E-CURD-10 (Kim et al., 2016)	Sample Characteristic[growth condition]	3	118	104
E-GEOD-99795 (Horning et al., 2018)	Factor Value[compound]	2	144	128
E-GEOD-110499 (Fan et al., 2018)	Sample Characteristic[organism part]	2	168	169
E-MTAB-7249 (Nelson et al., 2020)	Sample Characteristic[inferred cell type - authors labels]	7	185	185
E-MTAB-7606 (Koutsakos et al., 2019)	Sample Characteristic[phenotype]	2	209	203
E-GEOD-81383 (Gerber et al., 2017)	Sample Characteristic[cell line]	3	226	226
E-MTAB-6075 (Fischer et al., 2019)	Factor Value[compound]	3	236	231
E-GEOD-124858 (S. Liu et al., 2019)	Factor Value[passage]	5	241	239
E-MTAB-7381 (Fergusson et al., 2018)	Sample Characteristic[individual]	3	287	291
E-GEOD-111727 (Golumbeanu et al., 2018)	Sample Characteristic[disease]	2	320	324
E-GEOD-109979 (Lu et al., 2018)	Factor Value[time]	4	329	324
E-GEOD-100618 (Karamitros et al., 2018)	Sample Characteristic[cell type]	3	414	418
E-MTAB-8498 (Chen et al., 2020)	Sample Characteristic[cell type]	3	448	452
E-MTAB-9801 (Jardine et al., 2021)	Sample Characteristic[inferred cell type - authors labels]	12	484	501
E-GEOD-75688 (Chung et al., 2017)	Sample Characteristic[histology]	4	520	517
E-GEOD-86618 (Xu et al., 2016)	Sample Characteristic[disease]	2	537	538
E-GEOD-98556 (Phillips et al., 2018)	Sample Characteristic[cell type]	2	545	550
E-GEOD-83139 (Y. J. Wang et al., 2016)	Sample Characteristic[inferred cell type - ontology labels]	8	617	612
E-CURD-7 (Nguyen et al., 2018)	Sample Characteristic[sampling site]	2	631	627
E-ENAD-21 (Nguyen et al., 2018)	Sample Characteristic[cell type]	2	631	627
E-GEOD-70580 (Björklund et al., 2016)	Sample Characteristic[cell type]	4	648	643
E-ENAD-20 (Rambow et al., 2018)	Factor Value[time]	4	667	673
E-GEOD-75140 (Camp et al., 2015)	Sample Characteristic[cell type]	3	734	728
E-GEOD-89232 (Breton et al., 2016)	Sample Characteristic[cell type]	2	957	926
E-CURD-6 (Zhao et al., 2017)	Sample Characteristic[disease]	4	986	1031
E-MTAB-10596 (Hemeryck et al., 2022)	Sample Characteristic[growth condition]	3	993	984

Table F.1: All datasets analyzed during Section F.4.2.

F.11 Description of datasets used

A brief description and citation has been given in Table F.1 for every used dataset. Note that E-CURD-7 and E-ENAD-21 appear to be the same dataset; it is not a mistake. Both appear separately in the scRNA-seq database; a different categorical variable has been used for each.

Bibliography for Paper 6

Andrew, B., Westhead, D. R., & Cutillo, L. (2025). The Strong Product Model for Network Inference without Independence Assumptions. *Proceedings of The 28th International Conference on Artificial Intelligence and Statistics*, 5230–5238. Retrieved March 12, 2026, from https://proceedings.mlr.press/v258/andrew25a.html

Andrew, E. B., Westhead, D., & Cutillo, L. (2024). GmGM: A fast multi-axis Gaussian graphical model. *Proceedings of The 27th International Conference on Artificial Intelligence and Statistics*, 2053–2061. Retrieved March 12, 2026, from https://proceedings.mlr.press/v238/andrew24a.html

Benjamini, Y., & Speed, T. P. (2012). Summarizing and correcting the GC content bias in high-throughput sequencing. *Nucleic Acids Research*, *40*(10), e72. https://doi.org/10.1093/nar/gks001

Björklund, Å. K., Forkel, M., Picelli, S., Konya, V., Theorell, J., Friberg, D., Sandberg, R., & Mjösberg, J. (2016). The heterogeneity of human CD127(+) innate lymphoid cells revealed by single-cell RNA sequencing. *Nature immunology*, *17*(4), 451–460. https://doi.org/10.1038/ni.3368

Breton, G., Zheng, S., Valieris, R., Tojal da Silva, I., Satija, R., & Nussenzweig, M. C. (2016). Human dendritic cells (DCs) are derived from distinct circulating precursors that are precommitted to become CD1c+ or CD141+ DCs. *The Journal of experimental medicine*, *213*(13), 2861–2870. https://doi.org/10.1084/jem.20161135

Camp, J. G., Badsha, F., Florio, M., Kanton, S., Gerber, T., Wilsch-Bräuninger, M., Lewitus, E., Sykes, A., Hevers, W., Lancaster, M., Knoblich, J. A., Lachmann, R., Pääbo, S., Huttner, W. B., & Treutlein, B. (2015). Human cerebral organoids recapitulate gene expression programs of fetal neocortex development. *Proceedings of the National Academy of Sciences of the United States of America*, *112*(51), 15672–15677. https://doi.org/10.1073/pnas.1520760112

Chen, Y.-L., Gomes, T., Hardman, C. S., Vieira Braga, F. A., Gutowska-Owsiak, D., Salimi, M., Gray, N., Duncan, D. A., Reynolds, G., Johnson, D., Salio, M., Cerundolo, V., Barlow, J. L., McKenzie, A. N. J., Teichmann, S. A., Haniffa, M., & Ogg, G. (2020). Re-evaluation of human BDCA-2+ DC during acute sterile skin inflammation. *The Journal of experimental medicine*, *217*(3), e20190811. https://doi.org/10.1084/jem.20190811

Chung, W., Eum, H. H., Lee, H.-O., Lee, K.-M., Lee, H.-B., Kim, K.-T., Ryu, H. S., Kim, S., Lee, J. E., Park, Y. H., Kan, Z., Han, W., & Park, W.-Y. (2017). Single-cell RNA-seq enables comprehensive tumour and immune cell profiling in primary breast cancer. *Nature communications*, *8*, 15081. https://doi.org/10.1038/ncomms15081

DAWID, A. P. (1981). Some matrix-variate distribution theory: Notational considerations and a Bayesian application. *Biometrika*, *68*(1), 265–274. https://doi.org/10.1093/biomet/68.1.265

Dohm, J. C., Lottaz, C., Borodina, T., & Himmelbauer, H. (2008). Substantial biases in ultra-short read data sets from high-throughput DNA sequencing. *Nucleic Acids Research*, *36*(16), e105. https://doi.org/10.1093/nar/gkn425

Eltahla, A. A., Rizzetto, S., Pirozyan, M. R., Betz-Stablein, B. D., Venturi, V., Kedzierska, K., Lloyd, A. R., Bull, R. A., & Luciani, F. (2016). Linking the T cell receptor to the single cell transcriptome in antigen-specific human T cells. *Immunology and cell biology*, *94*(6), 604–611. https://doi.org/10.1038/icb.2016.16

Fan, J., Lee, H.-O., Lee, S., Ryu, D.-E., Lee, S., Xue, C., Kim, S. J., Kim, K., Barkas, N., Park, P. J., Park, W.-Y., & Kharchenko, P. V. (2018). Linking transcriptional and genetic tumor heterogeneity through allele analysis of single-cell RNA-seq data. *Genome research*, *28*(8), 1217–1227. https://doi.org/10.1101/gr.228080.117

Fergusson, J. R., Morgan, M. D., Bruchard, M., Huitema, L., Heesters, B. A., van Unen, V., van Hamburg, J. P., van der Wel, N. N., Picavet, D., Koning, F., Tas, S. W., Anderson, M. S., Marioni, J. C., Holländer, G. A., & Spits, H. (2018). Maturing Human CD127+ CCR7+ PDL1+ Dendritic Cells Express AIRE in the Absence of Tissue Restricted Antigens. *Frontiers in immunology*, *9*, 2902. https://doi.org/10.3389/fimmu.2018.02902

Fischer, C., Metsger, M., Bauch, S., Vidal, R., Böttcher, M., Grote, P., Kliem, M., & Sauer, S. (2019). Signals trigger state-specific transcriptional programs to support diversity and homeostasis in immune cells. *Science signaling*, *12*(581), eaao5820. https://doi.org/10.1126/scisignal.aao5820

Gerber, T., Willscher, E., Loeffler-Wirth, H., Hopp, L., Schadendorf, D., Schartl, M., Anderegg, U., Camp, G., Treutlein, B., Binder, H., & Kunz, M. (2017). Mapping heterogeneity in patient-derived melanoma cultures by single-cell RNA-seq. *Oncotarget*, *8*(1), 846–862. https://doi.org/10.18632/oncotarget.13666

Golumbeanu, M., Cristinelli, S., Rato, S., Munoz, M., Cavassini, M., Beerenwinkel, N., & Ciuffi, A. (2018). Single-Cell RNA-Seq Reveals Transcriptional Heterogeneity in Latent and Reactivated HIV-Infected Cells. *Cell reports*, *23*(4), 942–950. https://doi.org/10.1016/j.celrep.2018.03.102

Hemeryck, L., Hermans, F., Chappell, J., Kobayashi, H., Lambrechts, D., Lambrichts, I., Bronckaers, A., & Vankelecom, H. (2022). Organoids from human tooth showing epithelial stemness phenotype and differentiation potential. *Cellular and molecular life sciences*, *79*(3), 153. https://doi.org/10.1007/s00018-022-04183-8

Horning, A. M., Wang, Y., Lin, C.-K., Louie, A. D., Jadhav, R. R., Hung, C.-N., Wang, C.-M., Lin, C.-L., Kirma, N. B., Liss, M. A., Kumar, A. P., Sun, L., Liu, Z., Chao, W.-T., Wang, Q., Jin, V. X., Chen, C.-L., & Huang, T. H.-M. (2018). Single-Cell RNA-seq Reveals a Subpopulation of Prostate Cancer Cells with Enhanced Cell-Cycle-Related Transcription and Attenuated Androgen Response. *Cancer research*, *78*(4), 853–864. https://doi.org/10.1158/0008-5472.can-17-1924

Jardine, L., Webb, S., Goh, I., Quiroga Londoño, M., Reynolds, G., Mather, M., Olabi, B., Stephenson, E., Botting, R. A., Horsfall, D., Engelbert, J., Maunder, D., Mende, N., Murnane, C., Dann, E., McGrath, J., King, H., Kucinski, I., Queen, R., … Haniffa, M. (2021). Blood and immune development in human fetal bone marrow and Down syndrome. *Nature*, *598*(7880), 327–331. https://doi.org/10.1038/s41586-021-03929-x

Jordan, N. V., Bardia, A., Wittner, B. S., Benes, C., Ligorio, M., Zheng, Y., Yu, M., Sundaresan, T. K., Licausi, J. A., Desai, R., O'Keefe, R. M., Ebright, R. Y., Boukhali, M., Sil, S., Onozato, M. L., Iafrate, A. J., Kapur, R., Sgroi, D., Ting, D. T., … Haber, D. A. (2016). HER2 expression identifies dynamic functional states within circulating breast cancer cells. *Nature*, *537*(7618), 102–106. https://doi.org/10.1038/nature19328

Kalaitzis, A., Lafferty, J., Lawrence, N. D., & Zhou, S. (2013). The Bigraphical Lasso. *Proceedings of the 30th International Conference on Machine Learning*, 1229–1237. Retrieved March 12, 2026, from https://proceedings.mlr.press/v28/kalaitzis13.html

Karamitros, D., Stoilova, B., Aboukhalil, Z., Hamey, F., Reinisch, A., Samitsch, M., Quek, L., Otto, G., Repapi, E., Doondeea, J., Usukhbayar, B., Calvo, J., Taylor, S., Goardon, N., Six, E., Pflumio, F., Porcher, C., Majeti, R., Göttgens, B., & Vyas, P. (2018). Single-cell analysis reveals the continuum of human lympho-myeloid progenitor cells. *Nature immunology*, *19*(1), 85–97. https://doi.org/10.1038/s41590-017-0001-2

Karlsson, J., Kroneis, T., Jonasson, E., Larsson, E., & Ståhlberg, A. (2017). Transcriptomic Characterization of the Human Cell Cycle in Individual Unsynchronized Cells. *Journal of molecular biology*, *429*(24), 3909–3924. https://doi.org/10.1016/j.jmb.2017.10.011

Kim, K.-T., Lee, H. W., Lee, H.-O., Song, H. J., Jeong, D. E., Shin, S., Kim, H., Shin, Y., Nam, D.-H., Jeong, B. C., Kirsch, D. G., Joo, K. M., & Park, W.-Y. (2016). Application of single-cell RNA sequencing in optimizing a combinatorial therapeutic strategy in metastatic renal cell carcinoma. *Genome biology*, *17*, 80. https://doi.org/10.1186/s13059-016-0945-9

Koutsakos, M., Illing, P. T., Nguyen, T. H. O., Mifsud, N. A., Crawford, J. C., Rizzetto, S., Eltahla, A. A., Clemens, E. B., Sant, S., Chua, B. Y., Wong, C. Y., Allen, E. K., Teng, D., Dash, P., Boyd, D. F., Grzelak, L., Zeng, W., Hurt, A. C., Barr, I., … Kedzierska, K. (2019). Human $CD8^+$ T cell cross-reactivity across influenza A, B and C viruses. *Nature immunology*, *20*(5), 613–625. https://doi.org/10.1038/s41590-019-0320-6

Liu, H., Han, F., Yuan, M., Lafferty, J., & Wasserman, L. (2012). The nonparanormal skeptic. *Proceedings of the 29th International Coference on International Conference on Machine Learning*, 1731–1738.

Liu, S., Stroncek, D. F., Zhao, Y., Chen, V., Shi, R., Chen, J., Ren, J., Liu, H., Bae, H. J., Highfill, S. L., & Jin, P. (2019). Single cell sequencing reveals gene expression signatures associated with bone marrow stromal cell subpopulations and time in culture. *Journal of translational medicine*, *17*(1), 23. https://doi.org/10.1186/s12967-018-1766-2

Lu, J., Baccei, A., Lummertz da Rocha, E., Guillermier, C., McManus, S., Finney, L. A., Zhang, C., Steinhauser, M. L., Li, H., & Lerou, P. H. (2018). Single-cell RNA sequencing reveals metallothionein heterogeneity during hESC differentiation to definitive endoderm. *Stem cell research*, *28*, 48–55. https://doi.org/10.1016/j.scr.2018.01.015

Madrigal, P., Thanki, A. S., Fexova, S., Yu, I. D., Chatzigeorgiou, A., Zucchi, I., Marugan Calles, J. C., Vilmovsky, L., Khen, A., Zhao, L., Erdos, K., Kurri, S. R., Selvakumar, S., Kumbham, U., Prakash, A., Wang, S., Green, A., Ribas, C. E., Sweeney, B., … Ernst, C. (2026). Expression Atlas in 2026: Enabling FAIR and open expression data through community collaboration and integration. *Nucleic Acids Research*, *54*(D1), D147–D157. https://doi.org/10.1093/nar/gkaf1238

Nelson, L., Tighe, A., Golder, A., Littler, S., Bakker, B., Moralli, D., Murtuza Baker, S., Donaldson, I. J., Spierings, D. C. J., Wardenaar, R., Neale, B., Burghel, G. J., Winter-Roach, B., Edmondson, R., Clamp, A. R., Jayson, G. C., Desai, S., Green, C. M., Hayes, A., … Taylor, S. S. (2020). A living biobank of ovarian cancer ex vivo models reveals profound mitotic heterogeneity. *Nature Communications*, *11*(1), 822. https://doi.org/10.1038/s41467-020-14551-2

Nguyen, Q. H., Pervolarakis, N., Blake, K., Ma, D., Davis, R. T., James, N., Phung, A. T., Willey, E., Kumar, R., Jabart, E., Driver, I., Rock, J., Goga, A., Khan, S. A., Lawson, D. A., Werb, Z., & Kessenbrock, K. (2018). Profiling human breast epithelial cells using single cell RNA sequencing identifies cell diversity. *Nature communications*, *9*(1), 2028. https://doi.org/10.1038/s41467-018-04334-1

Phillips, M. J., Jiang, P., Howden, S., Barney, P., Min, J., York, N. W., Chu, L.-F., Capowski, E. E., Cash, A., Jain, S., Barlow, K., Tabassum, T., Stewart, R., Pattnaik, B. R., Thomson, J. A., & Gamm, D. M. (2018). A Novel Approach to Single Cell RNA-Sequence Analysis Facilitates In Silico Gene Reporting of Human Pluripotent Stem Cell-Derived Retinal Cell Types. *Stem cells (Dayton, Ohio)*, *36*(3), 313–324. https://doi.org/10.1002/stem.2755

Policastro, V., Righelli, D., Carissimo, A., Cutillo, L., & Feis, I. D. (2021). ROBustness In Network (robin): An R Package for Comparison and Validation of Communities. *The R Journal*, *13*(1), 276–293. https://doi.org/10.32614/RJ-2021-040

Rambow, F., Rogiers, A., Marin-Bejar, O., Aibar, S., Femel, J., Dewaele, M., Karras, P., Brown, D., Chang, Y. H., Debiec-Rychter, M., Adriaens, C., Radaelli, E., Wolter, P., Bechter, O., Dummer, R., Levesque, M., Piris, A., Frederick, D. T., Boland, G., … Marine, J.-C. (2018). Toward Minimal Residual Disease-Directed Therapy in Melanoma. *Cell*, *174*(4), 843–855.e19. https://doi.org/10.1016/j.cell.2018.06.025

Tierney, L., & Kadane, J. B. (1986). Accurate Approximations for Posterior Moments and Marginal Densities [_eprint: https://www.tandfonline.com/doi/pdf/10.1080/01621459.1986.10478240]. *Journal of the American Statistical Association*, *81*(393), 82–86. https://doi.org/10.1080/01621459.1986.10478240

Traag, V. A., Waltman, L., & van Eck, N. J. (2019). From Louvain to Leiden: Guaranteeing well-connected communities. *Scientific Reports*, *9*(1), 5233. https://doi.org/10.1038/s41598-019-41695-z

Tyler, D. E. (1987). A Distribution-Free M-Estimator of Multivariate Scatter. *The Annals of Statistics*, *15*(1), 234–251. https://doi.org/10.1214/aos/1176350263

Wang, Y., Jang, B., & Hero, A. (2020). The Sylvester Graphical Lasso (SyGlasso). *Proceedings of the Twenty Third International Conference on Artificial Intelligence and Statistics*, 1943–1953. Retrieved March 12, 2026, from https://proceedings.mlr.press/v108/wang20d.html

Wang, Y. J., Schug, J., Won, K.-J., Liu, C., Naji, A., Avrahami, D., Golson, M. L., & Kaestner, K. H. (2016). Single-Cell Transcriptomics of the Human Endocrine Pancreas. *Diabetes*, *65*(10), 3028–3038. https://doi.org/10.2337/db16-0405

Xu, Y., Mizuno, T., Sridharan, A., Du, Y., Guo, M., Tang, J., Wikenheiser-Brokamp, K. A., Perl, A.-K. T., Funari, V. A., Gokey, J. J., Stripp, B. R., & Whitsett, J. A. (2016). Single-cell RNA

sequencing identifies diverse roles of epithelial cells in idiopathic pulmonary fibrosis. *JCI insight*, *1*(20), e90558. https://doi.org/10.1172/jci.insight.90558

Yan, L., Yang, M., Guo, H., Yang, L., Wu, J., Li, R., Liu, P., Lian, Y., Zheng, X., Yan, J., Huang, J., Li, M., Wu, X., Wen, L., Lao, K., Li, R., Qiao, J., & Tang, F. (2013). Single-cell RNA-Seq profiling of human preimplantation embryos and embryonic stem cells. *Nature structural & molecular biology*, *20*(9), 1131–1139. https://doi.org/10.1038/nsmb.2660

Zhao, X., Zhao, X., Gao, S., Wu, Z., Kajigaya, S., Feng, X., Liu, Q., Townsley, D. M., Cooper, J., Chen, J., Keyvanfar, K., Fernandez Ibanez, M. D. P., Wang, X., & Young, N. S. (2017). Single-cell RNA-seq reveals a distinct transcriptome signature of aneuploid hematopoietic cells. *Blood*, *130*(25), 2762–2773. https://doi.org/10.1182/blood-2017-08-803353

Appendix G

Addendum to Chapter 2 ('GmGM: a fast multi-axis Gaussian graphical model')

We would like to clarify that, in Equation 2.1, the indices ij range over all entries of the output matrix:

$$\mathrm{tr}_b^a\left[\mathbf{M}\right] = \mathbf{X} \tag{G.1}$$

$$\iff \mathbf{X}_{ij} = \left[\mathrm{tr}\left[\mathbf{M}\left(\mathbf{I}_a \otimes \mathbf{J}^{ij} \otimes \mathbf{I}_b\right)\right]\right]_{ij} \text{ for all matrix entries } \mathbf{X}_{ij} \tag{G.2}$$

We would also like to clarify that, in Theorem 1, $\mathbf{S}_\ell$ is guaranteed to be a symmetric positive definite matrix (as its definition is of the form $\sum \mathbf{x}_i\mathbf{x}_i^T$). This is important, as it guarantees that the eigendecomposition the theorem refers to exists. For reference, we have repeated the theorem below.

Theorem (Theorem 1, Chapter 2)**.** *Let* $\mathbf{V}_\ell \mathrm{diag}\left[\mathbf{e}_\ell\right]\mathbf{V}_\ell^T$ *be the eigendecomposition of* $\mathbf{S}_\ell$ *(where* $\mathbf{V}_\ell \in \mathbb{R}^{d_\ell \times d_\ell}$ *and* $\mathrm{diag}\left[\mathbf{e}_\ell\right] \in \mathbb{R}^{d_\ell \times d_\ell}$ *is a diagonal matrix with diagonal* $\mathbf{e}_\ell$*). Then* $\mathbf{V}_\ell$ *are the eigenvectors of the maximum likelihood estimate of* $\mathbf{\Psi}_\ell$*.*

At the start of Section 2.4.2, we refer to 'a 10x Genomics dataset', but did not make clear which dataset, so we will do so here. The dataset was a paried scRNA-seq+scATAC-seq dataset of a lymph node with B cell lymphoma. Notably, the samples were flash-frozen before collection, and thus the RNA comes only from the nuclei ("single-nucleus RNA sequencing", or "snRNA-seq").

Our Figure 2.6 (comparing the assortativities of GmGM and ZiLN) was also hard to read, due to how much information we tried to cram into each axis. Here, we've split apart each subfigure of Figure 2.6 into the constituent phylogenic levels: phylum, class, order, family, and genus. This allows for an easier comparison of GmGM and ZiLN, as one can focus on a single level at a time.

In Figure G.1 (an updated Figure 2.6a), we compare the assortativities of our model (in terms of connecting nodes/'species' of the same phylum/class/order/family/genus) with that of ZiLN on the metagenomics data, showing broadly comparable results - albeit ZiLN does perform better at the order level and we perform better at the class level. Figure G.2 (an updated Figure 2.6b), in which we additionally give GmGM access to metabolomics data, tells a similar story; there is no evidence here that the addition of metabolomics data increases the ability of GmGM to discern species phylogenetics. In Figure G.3 (an updated Figure 2.6c), we give GmGM access to the phylum of the species as a prior, which greatly improves the results.

The colors in Figure 2.8b refer to the clusters learned by GmGM using louvain clustering. During the viva, the examiners pointed out that the experiment would be much better if we tested on a dataset that had cell labels, so that there was some way to validate our methodology. Thus, we have found a dataset of human PBMCs labeledfrom 10x Genomics[1]; this dataset consists of 2994 cells and 18132 genes, and has cell types derived from the measurement of cell surface proteins.

[1] In particular, we are using Sample C-A01 from this dataset

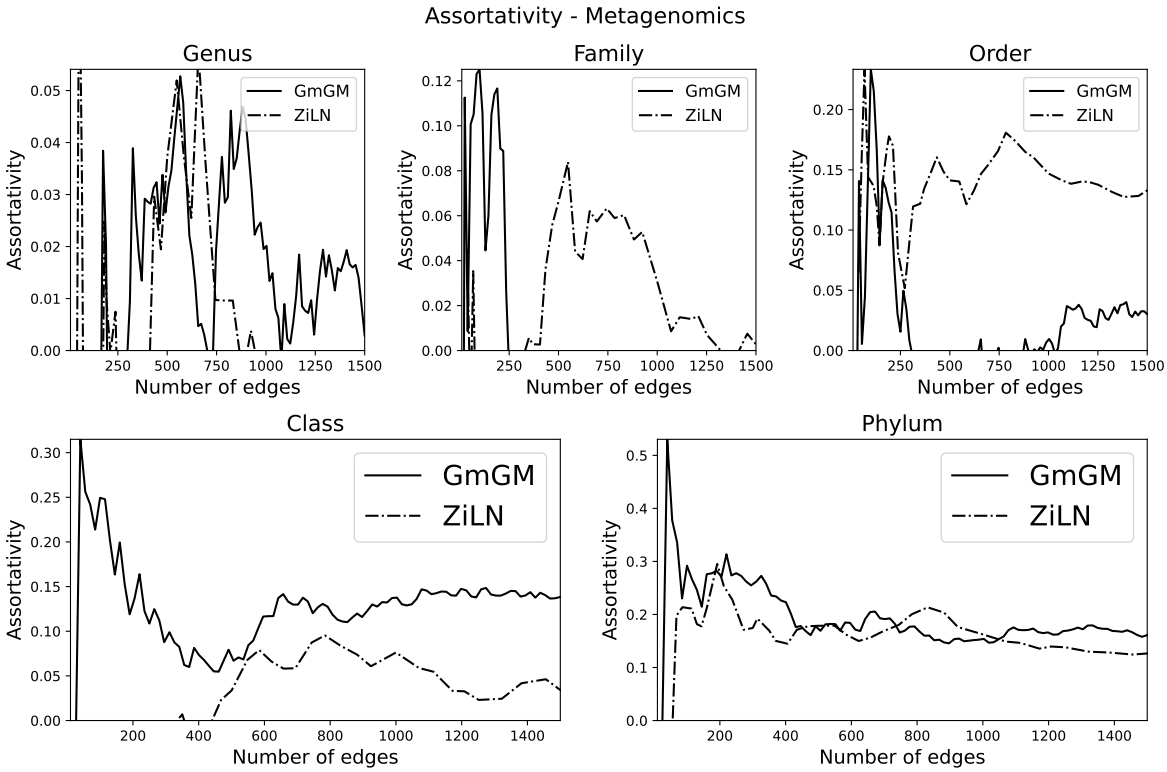


Figure G.1: A revised Figure 2.6a, splitting each level into its own plot for easy comparison between GmGM (solid lines) and ZiLN (dot-dashed).

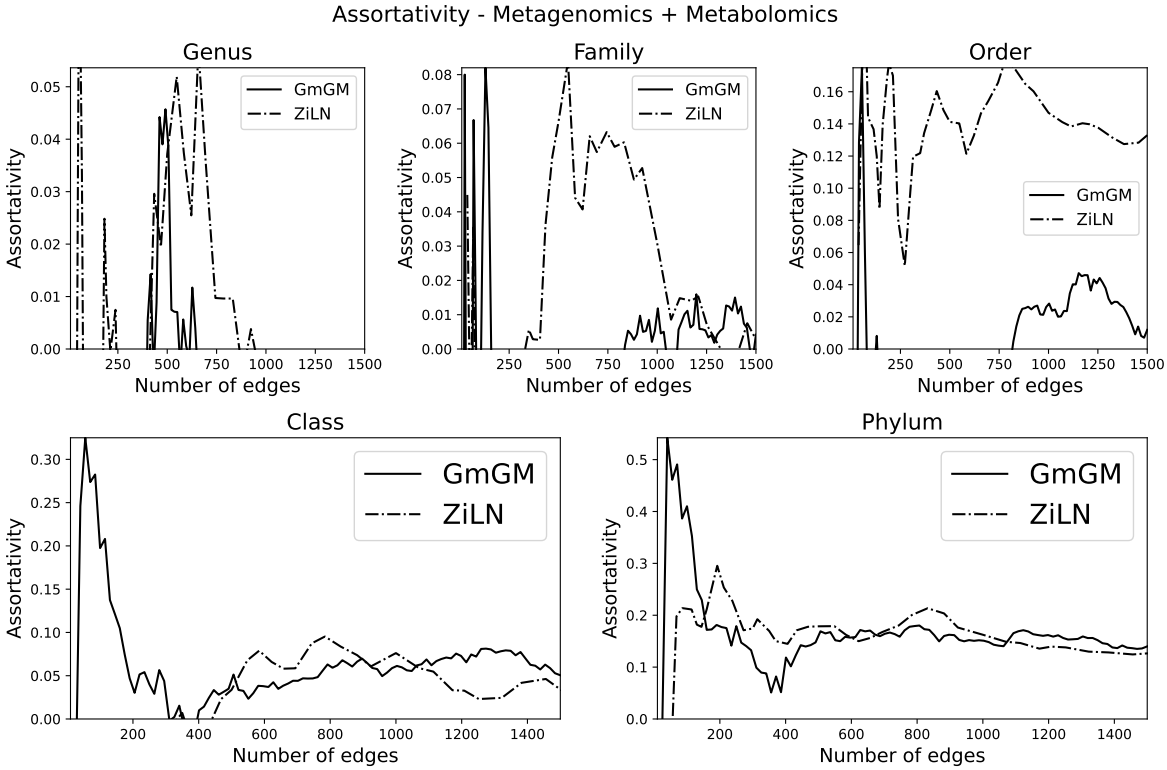


Figure G.2: A revised Figure 2.6b, splitting each level into its own plot for easy comparison between GmGM (solid lines) and ZiLN (dot-dashed).

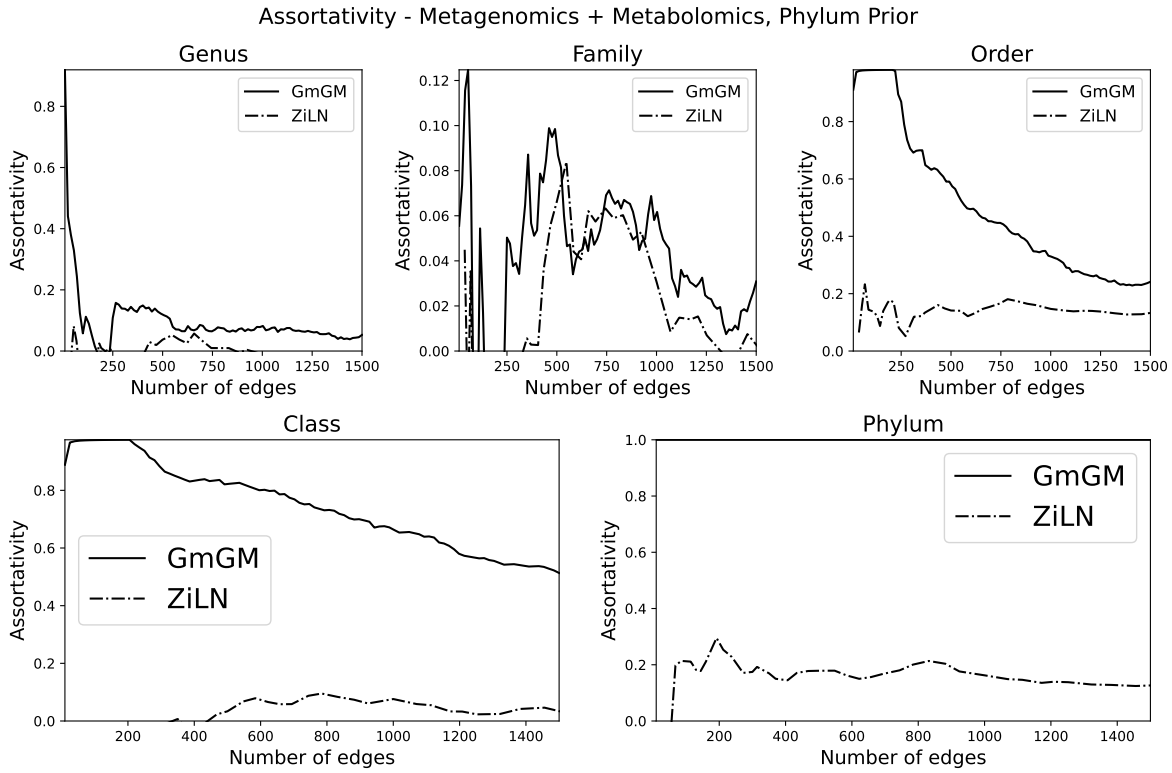


Figure G.3: A revised Figure 2.6c, splitting each level into its own plot for easy comparison between GmGM (solid lines) and ZiLN (dot-dashed). Note that, because the phylum is given as a prior, GmGM recovers it very well.

To analyze the dataset, we selected the top 5000 highly variable genes and ran GmGM on it, keeping the top 5 edges per cell and per gene. We then clustered using Leiden with a resolution of 1.2; Figure G.4 shows the result of our clustering, in comparison with their 10X Genomics' type labels derived from cell surface proteins. Visually, the clusters align well; in Figure G.5, we quantify how much overlap there is between our clusters and theirs. In particular, our clustering is able to identify major cell types (such as B and T cells). It provides a finer classification of monocytes, which could represent genuine biological heterogeneity or be a clustering artifact. We do not pick up clusters specific to granulocytes (2 cells), hematopoietic cells (10 cells), hematopoietic precursor cells (1 cell), or mast cells (20 cells). Such cell types are very rare, with the four making up a cumulative 1.1% of the total amount of cells (2994), most of which being mast cells. For contrast, there are 904 cells labelled by 10X as being monocytes.

Some of GmGM's clusters are split between two 10X types (our cluster 4 is split between T cells and lymphocytes, and our cluster 8 is split between monocytes and myeloid leukocytes). This is not a problem; T cells *are* lymphocytes, and monocytes *are* myeloid leukocytes - the lymphocyte and myeloid leukocyte 10X clusters represent very high-level cell type assignments that 10X were unable to refine further based on the cell surface markers they measured. As additional validation, we have confirmed that GmGM cluster 6 is enriched for traditional T cell markers (CD3D, CD3G, and CD3E are all within the top 20 most enriched genes in that population).

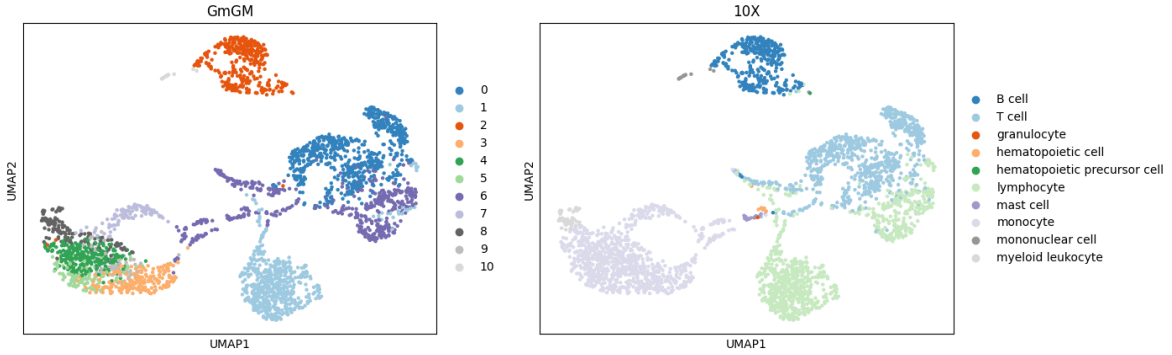


Figure G.4: GmGM (Leiden resolution=1.2) clusters compared to cell-surface-protein-based annotations from a 10X Genomics human PGMC dataset.

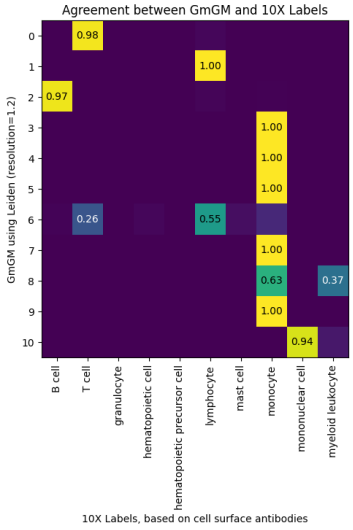


Figure G.5: Each row represents the composition of a single GmGM cluster in terms of the 10X cell type labels. Only when a 10X label comproses at least 20% of a cluster is the percentage displayed on the plot.

Appendix H

Addendum to Chapter 3 ('Gaussian graphical modelling without independence assumptions for uncentered data')

We would like to clarify our meaning behind our statement that 'estimating the mean of a single sample is a recipe for disaster; when one centers the dataset, they will be left with a constant zero vector.' The examiners pointed out it would be clearer if we had said 'In the single-sample multi-axis setting, we cannot estimate an unstructured mean vector because we have $d_\forall$ unknowns and only 1 observation. The maximum likelihood estimate of $\boldsymbol{\mu}$ would be the data vector $\mathbf{x}$ itself, which is equivalent to assuming $\boldsymbol{\mu} = \mathbf{x}$ and leaving zero residuals.' We agree with this - the main takeaway we hope readers have is that it is completely futile to attempt estimate the unstructured mean in a Kronecker-structured scenario; it is necessary to make some kind of assumption on the structure of the mean.

We would also like to provide axis labels to our various figures given in the paper (for Figures 3.1, 3.3, and 3.4); we give these, respectively, in Figures H.2, H.3, and H.4.

During the viva, the examiners asked whether we felt the E-MTAB-2805 mouse embryo stem cell experiment had been impacted by the selection of genes (as we had selected a small amount, all relating to mitosis). It is certainly possible. As all the genes are related to mitosis, it's possible that their mean expressions are similar, reducing the utility of our model (which will provide the largest improvement when the means are different and thus not accounted for by the simple 'subtract the global mean' technique). However, empirically the mitosis-expressing genes take on a wider range of mean values with greater frequency than the set of all 38,293 genes in the unfiltered dataset; we've added a new figure, Figure H.1, to show this[1]. This shows that the spread of mean values for mitosis genes, despite their relatedness, is actually quite large in the context of scRNA-seq data, motivating methods such as ours.

This question also came up in the context of the next paper (Chapter 4), in which a more critical issue was raised (and which I assume the intent of this being mentioned in the list of corrections is to address, rather than for this paper). The concern raised during the viva was that, as all genes were mitosis-related, they were unlikely to interact across different cells; thus it is no wonder that we did not find much improvement when using the strong product model. This is a valid concern; it is not the ideal dataset to test on to validate our method on; in this aspect, our COIL experiment (Section 4.5.2) is better as it is easy to imagine cross-dependencies in video data (the location of the rubber duck's head in one part of the video does give you information on the likely location of its tail, despite them being in completely different rows and columns).

In our description of the problem our optimization problem, given below, the meaning of $\theta_{\setminus \ell}$ was not clear:

[1] It should not be surprising that this occurs, given not all genes will be active in an embryo - and hence have zero or near-zero expression - but mitosis genes almost certainly will be active.

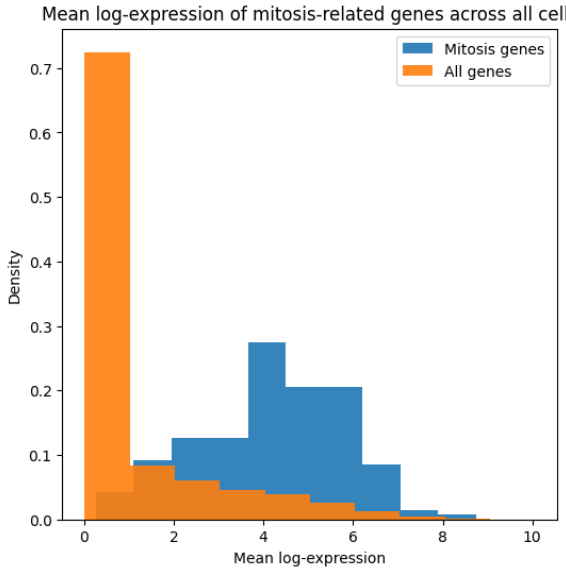


Figure H.1: A histogram of the mean log-expression across cells for all genes (orange) compared to just the mitosis-related genes that we used (blue). The spread of expression values of the two categories is roughly equal, but the mitosis-related genes tend to utilize more of that spread, whereas the expression values of all genes are much more concentrated on the low end.

$$
\begin{aligned}
&\underset{\widetilde{\boldsymbol{\mu}}_\ell}{\operatorname{argmin}}\ \widetilde{\boldsymbol{\mu}}_\ell^T \mathbf{A}_\ell \widetilde{\boldsymbol{\mu}}_\ell + \mathbf{b}_\ell^T \widetilde{\boldsymbol{\mu}}_\ell \\
&\text{where } \mathbf{A}_\ell = d_{\setminus\ell} \boldsymbol{\Psi}_\ell + \theta_{\setminus\ell} \mathbf{I}_{d_\ell} \\
&\qquad \boldsymbol{b}_\ell = m d_{\setminus\ell} \mathbf{1}_{d_\ell}^T \boldsymbol{\Psi}_\ell + m\theta_{\setminus\ell} \mathbf{1}_{d_\ell} + \left[\widetilde{\boldsymbol{\mu}}_{\setminus\ell}^T \boldsymbol{\Psi}_{\setminus\ell} \mathbf{1}_{d_{\setminus\ell}}\right] \mathbf{1}_{d_\ell} \\
&\qquad\quad - \mathbf{x}_\ell^T \left(\boldsymbol{\Psi}_\ell \otimes \mathbf{1}_{d_{\setminus\ell}}\right) - \mathbf{x}_\ell^T \left(\mathbf{1}_\ell \otimes \boldsymbol{\Psi}_{\setminus\ell} \mathbf{1}_{d_{\setminus\ell}}\right) \\
&\qquad \widetilde{\boldsymbol{\mu}}_\ell^T \mathbf{1}_{d_\ell} = \mathbf{0}
\end{aligned}
$$

It had been defined in Section 3.2, but was easy to miss as it had not been included in the notation section (Section 3.1.2). The definition is as follows; $\theta_{\setminus\ell} = \sum_{\ell' \neq \ell} \frac{d_\forall}{d_\ell d_{\ell'}} \theta_\ell$.

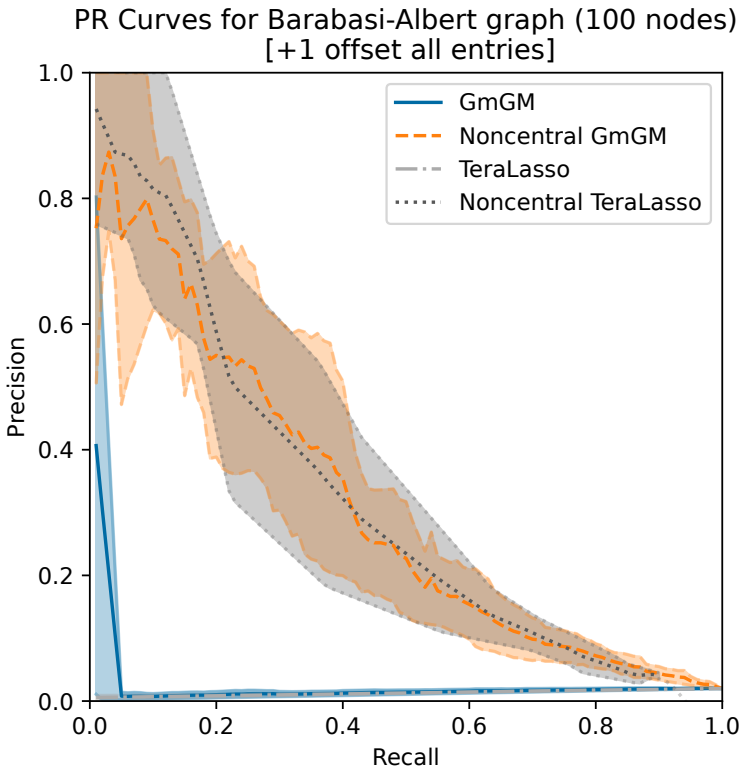


Figure H.2: (Figure 3.1) Precision and recall for a synthetic dataset in which data was generated from a Kronecker-sum normal distribution with a constant value of 1 added to every entry of the matrix. Error bars are the best/worst performance over 10 trials; the center line is average performance.

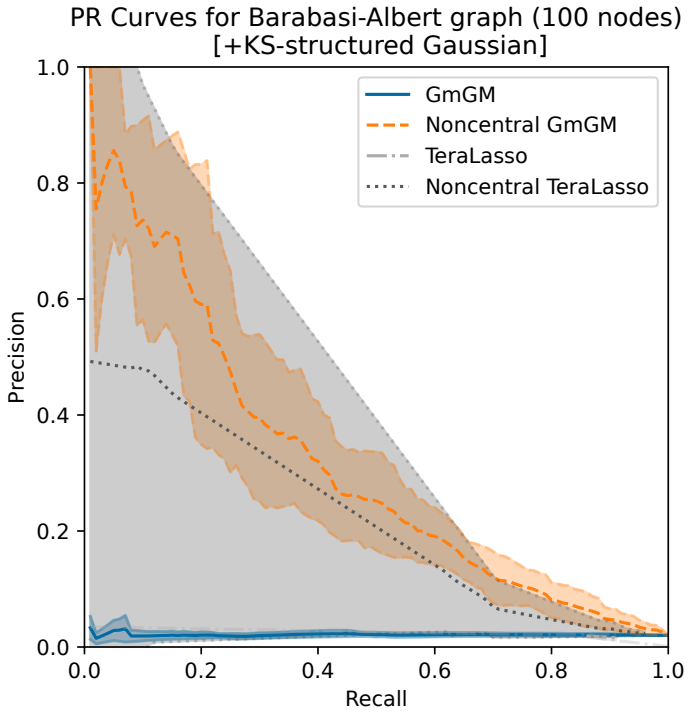


Figure H.3: (Figure 3.3) Precision and recall for on a synthetic dataset in which data was generated from a noncentral Kronecker-sum normal distribution. Noncentral TeraLasso's high susceptibility to changes in the regularization parameter led to our grid search being too coarse-grained during some trials.

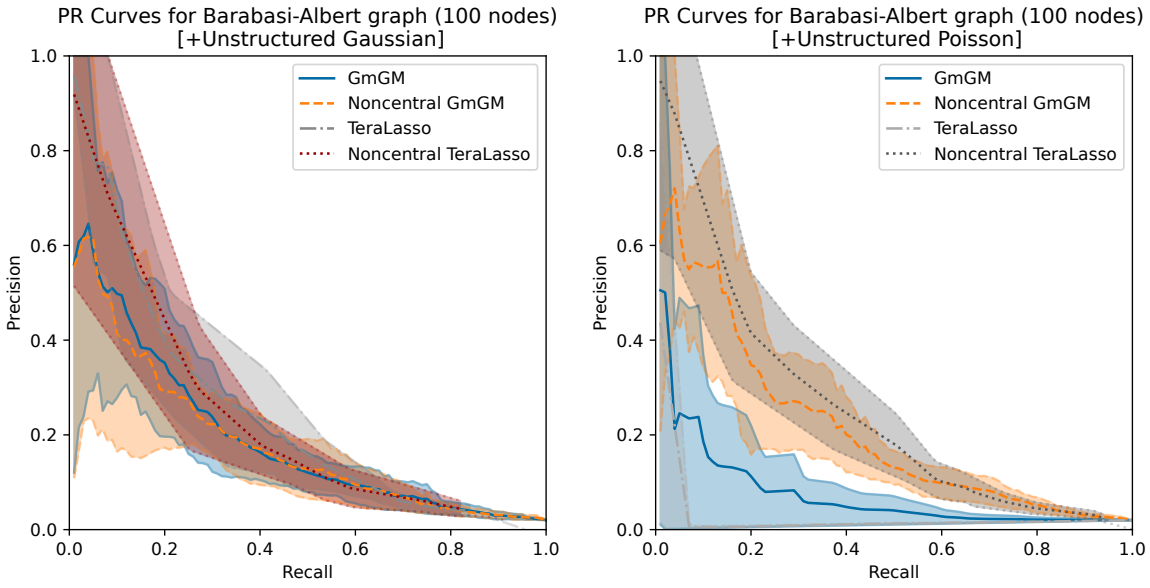


Figure H.4: (Figure 3.4) Precision and recall for on a synthetic dataset in which data was generated from a zero-mean Kronecker-sum normal distribution, with independent Gaussian (left) and Poisson (right) noise added to every element. Error bars are the best/worst performance over 10 trials; the center line is average performance.

Appendix I

Addendum to Chapter 4 ('The strong product model for network inference without independence assumptions')

We would like to clarify the relevant axes in our definitions of the stridewise/blockwise traces, which we gave as:

$$\mathrm{tr}^{d_1}\left[\mathbf{A}\right]_{ij} = \mathrm{tr}\left[\mathbf{A}\left(\mathbf{I}\otimes\mathbf{J}^{ij}\right)\right]$$
$$\mathrm{tr}_{d_2}\left[\mathbf{A}\right]_{ij} = \mathrm{tr}\left[\mathbf{A}\left(\mathbf{J}^{ij}\otimes\mathbf{I}\right)\right]$$

These could be more clearly written as follows, where d_{rows} is the amount of rows and d_{cols} is the amount of columns of the original data matrix:

$$\mathrm{tr}^{d_{rows}}\left[\mathbf{A}\right]_{ij} = \mathrm{tr}\left[\mathbf{A}\left(\mathbf{I}\otimes\mathbf{J}^{ij}\right)\right]$$
$$\mathrm{tr}_{d_{cols}}\left[\mathbf{A}\right]_{ij} = \mathrm{tr}\left[\mathbf{A}\left(\mathbf{J}^{ij}\otimes\mathbf{I}\right)\right]$$

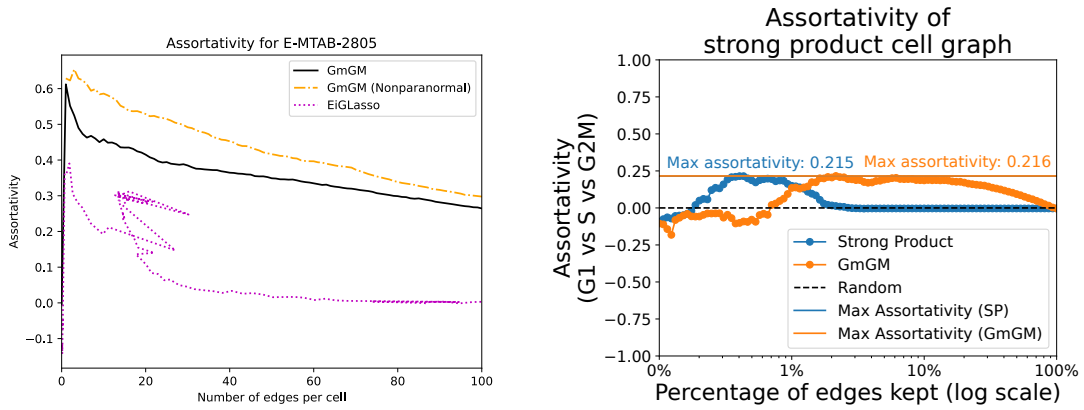
Perhaps counterintuitively, $\mathrm{tr}^{d_{rows}}\left[\mathbf{A}\right]$ is a $d_{cols}\times d_{cols}$ matrix and $\mathrm{tr}_{d_{cols}}\left[\mathbf{A}\right]_{ij} = \mathrm{tr}\left[\mathbf{A}\left(\mathbf{J}^{ij}\otimes\mathbf{I}\right)\right]$ is a $d_{\text{rows}}\times d_{\text{rows}}$ matrix - this is because the superscript/subscript represent the size of the blocks to do the trace over; if one divides a $d_{rows}d_{cols}\times d_{rows}d_{cols}$ matrix into blocks of size $d_{rows}\times d_{rows}$, and consider the matrix of traces of such blocks, one obtains a $d_{cols}\times d_{cols}$ matrix.

In Section 4.4.2, we wrote: 'In this paper, we follow Deng and Tsui (2013) by imposing a Frobenius norm penalty $\|\log\boldsymbol{\Gamma}\|_F^2$ on the logarithms of our parameters.' We would like to clarify that here, $\boldsymbol{\Gamma}$ is meant to represent an arbitrary parameter - i.e. any of $\{\boldsymbol{\Psi}_1, \boldsymbol{\Psi}_{2,w}, \boldsymbol{\Psi}_{2,b}\}$.

We would also like to explain in more detail how the synthetic data of Section 4.5.1 was generated. In the paper, we had described how the ground-truth parameters were generated[1], but we did not adequately describe how the data was generated from such parameters. We generated the data $\mathbf{D}$ from within the distribution our model assumed, i.e. $\mathbf{D}\sim\mathcal{N}\left(\mathbf{0}, y\boldsymbol{\Psi}_1\otimes x\boldsymbol{\Psi}_{2,b} + y\boldsymbol{\Psi}_1\oplus\frac{1}{x}\boldsymbol{\Psi}_{2,w}\right)$. By validating our model on within-distribution synthetic data, we are able to isolate the effect of certain parameters, such as x and problem size in Figure 4.3; the remaining sections validate our problem on real-world out-of-distribution datasets.

Regarding the real-world results, we mentioned that we used as input to the models $\mathbf{D} - \frac{1}{d_1 d_2}\mathbf{1}^T\mathbf{D}\mathbf{1}$ instead of $\mathbf{D}$, 'to lessen the impact of the zero-mean assumption'. As we showed in Chapter 3, not

[1]'For our synthetic data, we generated our ground truth graphs to have the form $y\mathbf{A}\otimes x\mathbf{B} + y\mathbf{A}\oplus\frac{1}{x}\mathbf{C}$, where $\mathbf{A},\mathbf{B},\mathbf{C}$ are Barabasi-Albert random graphs with 500 nodes each.' - although it would have been clearer if we had written $y\boldsymbol{\Psi}_1\otimes x\boldsymbol{\Psi}_{2,b} + y\boldsymbol{\Psi}_1\oplus\frac{1}{x}\boldsymbol{\Psi}_{2,w}$ instead.



(Left) Chapter 2's E-MTAB-2805 test using GmGM and EiGLasso. (Right) Chapter 4's E-MTAB-2805 test using the strong product model and GmGM.

doing anything to account for a nonzero mean can seriously degrade model performance. However, the results of Chapter 3 are specific to the Kronecker sum model - our 'strong product model' would require a new approach (although it is likely that the approach would be similar in structure to that of Chapter 3). $\mathbf{D} - \frac{1}{d_1 d_2}\mathbf{1}^T\mathbf{D}\mathbf{1}$ amounts to subtracting the average of all dataset elements from each element in the dataset, and thus is a good first attempt to center our data (and we saw that in Chapter 3, at least for Kronecker sum models, this approach worked almost as well as the more sophisticated approach developed in that chapter).

We would also like to address the differences in the E-MTAB-2805 test in Chapters 4 and 2, in which we get different results on the same dataset; we've repeated the plots here for convenience.

The reason the results are so different is due to a change in how we chose to keep the final edges after running GmGM - in Chapter 4, we used global thresholding. In all subsequent papers, we used per-row thresholding, as we found it gave better results when clustering - although in the case of E-MTAB-2805, it appears to have worsened assortativity, a fact we did not realize until it was pointed out to us. It is surprising, although we stand by the practice of thresholding per-row in general as good clustering is the typical goal of having a cell network; thresholding overall may yield better assortativity, but this is primarily because the most dominant cell type gets most of the edges and the remaining cell types tend to become singletons.

Appendix J

Addendum to Chapter 5 ('The Cartesian Gaussian additive noise model for directed network inference in omics data')

We did not make it clear how we implemented LGAM when running our experiments. Our LGAM implementation came 'for free' from our ⊞-LGAM implementation. To see why, observe their defining equations:

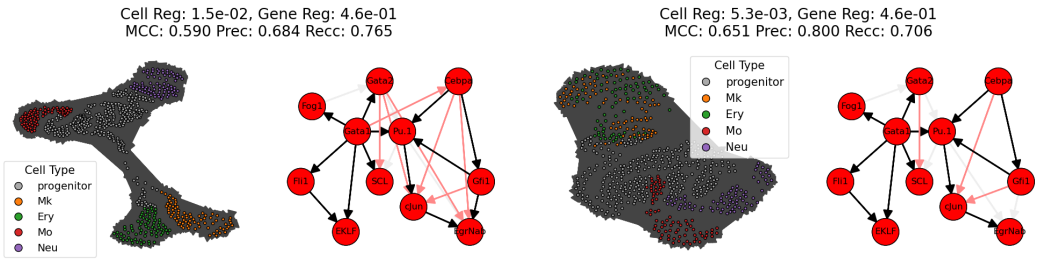
$$\mathbf{X} = \mathbf{C}_1\mathbf{X} + \mathbf{X}\mathbf{C}_2^T \qquad (\boxplus\text{-LGAM})$$
$$\mathbf{X} = \mathbf{C}_1\mathbf{X} \qquad (\text{LGAM})$$

Thus, ⊞-LGAM reduces to LGAM if we force $\mathbf{C}_2 = \mathbf{0}$. As we fit our model in terms of $\mathbf{L}_\ell = \sqrt{d_{\setminus\ell}}\left(\frac{1}{K}\mathbf{I} - \mathbf{C}_\ell\right)$, this is equivalent to forcing $\mathbf{L}_2 = \frac{\sqrt{d_{\setminus 2}}}{K}\mathbf{I}$. Analogously for tensor-variate datasets, for all $\ell > 1$ we force $\mathbf{L}_\ell = \frac{\sqrt{d_{\setminus\ell}}}{K}$ and only let $\mathbf{L}_1$ be free. This is trivial to implement - our method for ⊞-LGAM was to flip-flop over each axis; to implement standard LGAM, we do not need to flip-flop, and rather only need to run the optimization problem for the first axis. As the values we enforce for the other $\mathbf{L}_\ell$ are valid parameters in the ⊞-LGAM context, this does not have any negative consequences (such as ruining convexity).

We also did not spend enough time justifying and discussing the limitations of Assumption 4 (that the DAGs have known order). This is a very strong assumption, and the need for it is the key limitation of our paper. We do discuss how to estimate the ordering in the equal-variance scenario (Theorem 9) and give a practical example for scRNA-seq (Section 5.5.4), but these are just estimates - in general, one will very rarely *know* the causal ordering in 'omics data. This holds for both genes and cells - in fact, the concept of a causal ordering for cells itself is dubious, although one could reasonably interpret such results through the lens of pseudo-time. The lack of knowledge of the ordering hampers our model's performance, but not to an extreme degree (Table 5.2).

We make this assumption to ensure feasibility of learning the objective. Modifying our method to work without it is, in theory, trivial - one needs only relax the assumption of each $\mathbf{L}_\ell$ being lower-triangular to being acyclic. However, the optimization problem in this case would become NP-Hard. There are some single-axis attempts to address this, notably the NOTEARS methodology, that in future work would be useful to combine with our approach. We show in Table 5.2 that our approach still performs well with the wrong ordering, but there is a noticeable drop in performance - and thus ample theoretical and practical justification for future work on a unified NOTEARS+⊞-LGAM approach.

We would also like to clarify some details about Figure 5.2, which we have repeated here for convenience. There are two subfigures, each representing a ⊞-LGAM result that arose during the grid search we performed. Each subfigure contains both a cell graph and a gene graph; the layout of the gene graph was manually determined to be able to see all edges clearly, whereas the cell graph was



(Left) A gene graph arising from our ⊞-LGAM's grid search. Chosen for display as its cell graph qualitatively captures cell development the best. (Right) The best gene graph arising from our ⊞-LGAM model's grid search. (All) Black edges are true positives, near-white edges are false negatives, pale red edges are false positives.

plotted using the Fruchterman-Reingold force-directed algorithm (Fruchterman and Reingold, 1991) - the igraph package's default plotting method. Importantly, the Fruchterman-Reingold algorithm is blind to the cell labels; they did not play a role in the graph layout.

The subfigures display the results of our method on a simulated scRNA-seq dataset by Krumsiek et al. (2011). This dataset was chosen for several reasons:

1. As the Krumsiek dataset is synthetic, we have access to the ground truth gene network - a rarity in scRNA-seq data. This enables the use of quantitative metrics such as MCC.

2. The cells are cleanly divided into five cell types (progenetors, megakaryocytes, erythrocytes, monocytes, and neutrophils) that have natural pseudotemporal structure.

3. The dataset is very lopsided (640 cells and only 11 genes). We spent a decent amount of time in the paper discussing the dangers of lopsidedness - this dataset is closer to the worst case scenario on that front, and serves as a demonstration that, in practice, our method can still perform well in the presence of extreme lopsidedness.

4. The ground truth graph is not a DAG - it has cycles! This is a realistic occurrence in transcriptomics, which violates our assumptions. Like in the previous point, it gives us a chance to show that our method can still perform structure recovery even when the DAG assumption is violated.

The rightmost subfigure shows the results of the best-performing result based on the MCC of the gene graph; this occurred with our regularization parameter for the cells being 5.3×10^{-3} and for the genes being 4.6×10^{-1}. The cell graph here is dense enough that it is hard to make out much structure in its plot beyond the fact that, visually, cells of the same type seem to cluster together.

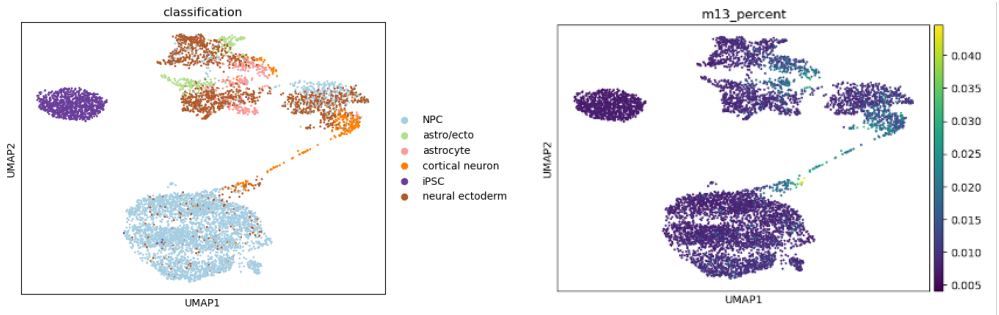
The leftmost subfigure shows the result of our third-best-performing result based on the MCC of the gene graph; it was chosen for display because, visually, the cell graph appeared to be capturing the developmental process of the cells (progenitors bifurcating into two branches, one of which further bifurcates into monocytes and neutrophils and another which bifurcates into erythrocytes and megakaryocytes, mirroring the PCA plot in Figure 5.1). We highlighted this result as the optimizing characteristic (MCC on the gene graph) did not refer to the cells at all, and yet by optimizing for gene performance we still were able to acquire relevant emergent features in the cell graph.

Finally, the examiners pointed out work on the Graphical ElasticNet (Kovacs, Ruckstuhl, Obrist, and Buhlmann, 2021) that serves as good relevant background to our Equation 5.4. Previously, we had stated that Equation 5.4 was analogous to ElasticNet (Zou and Hastie, 2005); the analogy is stronger with the Graphical ElasticNet, and we are grateful to the examiners for pointing it out.

Appendix K

Addendum to Chapter 6 ('Making multi-axis Gaussian graphical models scalable for millions of samples and features')

We would like to clarify some details of Figure 6.4, a UMAP plot which is repeated below for convenience. The UMAP is created in the standard way, i.e. based off of the k-nearest-neighbors network (where $k = 15$, the default in the scanpy package) of the gene expression of all genes in the dataset.



(left) The cell type classification produced by GmGM using the marker genes from Appendix E.13. NPC stands for 'neural progenitor cell', and 'astro/ecto' denotes cells which could not be distinguished between astrocyte and neural ectoderm with the markers given. (right) The average gene expression of the genes in GmGM module m13; it tends to express itself only in and around cells that are identified as cortical neurons. (both) UMAP plots have been used for visualization; the overexpression of m13 in the suspected cortical neuron cells has also been validated statistically.

The figure consists of a left and right subfigure; the left figure shows the cell type labellings applied to the cell clusters learned by GmGM. The right plot shows the average percentage expression of all genes in the module m13, highlighting the correspondence between that gene module's expression and the cells labeled as cortical neurons in the leftmost plot.

During the viva, the examiners raised the worry that, by computing the average of all genes, our results may be dominated by the most expressed gene in the module. This is a very reasonable worry, and would make our plot not representative of the module m13, only its most expressed member. To help assuage such worries, we have added Figure K.1, in which we show that the upper quartile of the module still corresponds with cortical neurons.

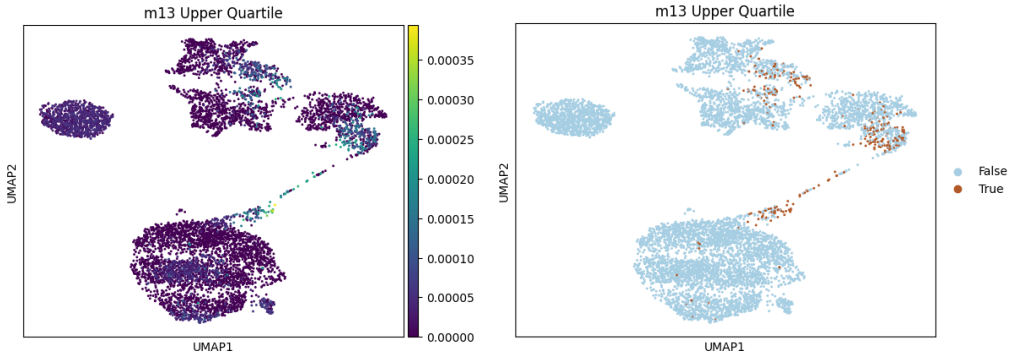


Figure K.1: (Left) The percentage of gene expression taken by the upper-quartile-ranked gene in m13 (i.e. the gene in m13 whose expression is greater than 75% of other m13 genes but less than the remaining 25%). (Right) The same as the left plot, but with a cutoff of 0.01% to make the differences clearer.