Hierarchical Latent Space Models for Multiplex Social Network Analysis

Alex Martin Loewi
aloewi@cmu.edu

Francisco Ralston Fonseca
fralston@andrew.cmu.edu

Octavio Mesner
omesner@cmu.edu

Abstract

An extension of Latent Space Models is developed, for parsimoniously fitting multigraph data. The major advantage of the model is that it allows for interpretable visualization of multigraphs, a necessary but difficult task in social network analysis and many other fields including medical diagnostics, and deep learning. The model is fit using a group LASSO penalty that both increases interpretability, and also solves a second important problem by allowing for graph-wise dimensionality reduction of this complicated form of data.

1 Introduction

1.1 The Curious Case of SNA

Social network analysis (SNA) is a complex, and inherently interdisciplinary field. It takes the already complex fields of psychology and sociology, and focuses on the most interdependent processes in these disciplines. This then creates systems so complicated that sophisticated statistical models are necessary to make any sense of them. While this complexity makes SNA an exciting intersection, effectively spanning these two disparate approaches is exceptionally difficult. As a result, while it attracts a great deal of attention, many important problems in SNA are under-studied, simply because of the small number of people who are trained to deal with both the substance, and methods, of the field.

One of these problems is that of modeling multigraphs. Multigraphs, also called multiview networks, can be thought of as either a single graph with multiple edge types, or as multiple networks over the same set of nodes. Data of this kind is not uncommon, at either large or small scales. Twitter has Following relationships, as well as Retweets, and Likes, each of which can be thought of as simply one layer within the full relationship between two nodes. In small scales, surveys for networks very rarely ask about a single type of relationship, and many of the most famous network data sets have multiple relationships (such as Trust, Friendship, and Respect, in Samson's monk data). Despite the existence of data of this form for decades, the strange nature and inherent difficulty of this data has led to an extremely small number of ways to model it, all of which have serious shortcomings. The models from statistics tend to ignore the problems of the practitioner, and the practitioners rarely have the training to develop their own methods.

1.2 The Practical Difficulties of Multigraphs

In particular, practitioners have two immediate problems when dealing with multigraphs, both of which are consequences of the inherent complexity of the data.

1.2.1 Visualization

The first problem is that multigraphs are too complex to easily visualize, and visualization has always been a cornerstone of network analysis. In particular, the difficulty comes in comparing the many

31st Conference on Neural Information Processing Systems (NIPS 2017), Long Beach, CA, USA.

layers of data (i.e. the graphs formed by the different edge types). Because two layers have the same set of nodes, comparing them requires the nodes to be in similar positions for the two layouts. However, graph layout algorithms are designed to pay attention only to the connectivity in a graph, and find the node positions that best reflect that connectivity – if the node positions are fixed, all of the meaningful structure in a layer may be obscured. To define the problems in terms of its extremes, there is a trade-off in terms of comparability (fixed nodes) and expressiveness (free nodes). A useful multigraph model would be able to tune explicitly between these two extremes.

1.2.2 Dimensionality

The second problem is that multigraphs may be unnecessarily complex. Empirically, there are often high degrees of correlation between the layers in a multigraph, and the smaller the number of graphs, the easier analysis becomes. These two observations combine to suggest another useful property of a new model – the ability to remove redundant layers. While existing models often take the data and make it even more complicated, that runs counter to the needs of an applied social network analyst.

1.3 Previous Work

Our model combines several methods that have proven to be exceptionally valuable – Latent Space Models, [1], the LASSO estimator, [2], and in particular the group LASSO. The substantial amount of work that has been done on them provides many different possible approaches to designing, and fitting, our own model. Efficient optimization procedures, often employing coordinate or block-coordinate optimization, exist for fitting the LASSO to linear models, [2], to generalized linear models [3], with the element-wise, group, and combined, sparse group LASSO, [4, 5, 6], and more. This year already, two different papers have been published that attempt to use Latent Space ideas to model multigraphs – however, our approach has substantial advantages over both of them, in part by focussing on problems these models are not actually attempting to solve. One approach takes the extreme of modeling the multigraph as a single object, which only makes visualization more difficult [7]. The other treats each layer as independent, and allows correlations between them, which does not solve the comparability problem, or that of dimensionality reduction [8]. Furthermore, that model is described as "much too flexible" but the authors themselves, which may be because it suffers from an identifiability issue. Allowing correlations between layers means that an edge may exist either because its positions are close, or because the positions in a correlated layer are close. While clearly expressive, this ambiguity is not a desirable quality in social science.

2 The Model

2.1 Problem Statement

Motivated by these problems, we propose a model with the following three objectives:

1. Use current methods from statistical SNA to model multigraphs
2. Model the layers of the multigraph in a way that can tune for comparability
3. Allow the model to remove redundant layers

A model with all of these qualities, which we term the Hierarchical Latent Space Model, can be described in the following way:

2.2 The Likelihood Function

Using the canonical Latent Space Model [**?**] as a starting point, we model a set of conditionally independent binary dyads.

The existence of each of these dyads is a function of an intercept, a set of covariates (omitted here for clarity), and the distance between the latent "position" variables z of the two nodes in the dyad. The goal of the model will be to estimate these positions. An optimal set of variables would place positions close together for nodes with an observed edge, and vice versa.

$$\eta_{ijk} = \alpha_k - \|z_{ik} - z_{jk}\|_2^2$$

The indices i and j are over the nodes; the index k refers to the different layers in the multigraph. Because the edges are binary, we use the inverse logit σ to transform η, but it should be observed that for real-valued edges, other link functions could be easily substituted.

$$\sigma(\eta_{ijk}) = \frac{1}{1+\exp(-\eta_{ijk})} \in [0,1]$$

All together, the unpenalized likelihood of the HLSM thus takes the form of a binomial:

$$L = \prod_k \prod_i \prod_{j<i} \sigma(\eta_{ijk})^{y_{ijk}} (1-\sigma(\eta_{ijk}))^{1-y_{ijk}}$$

and the log likelihood is

$$\begin{aligned}
\ell &= \sum_k \sum_i \sum_{j<i} y_{ijk} \ln \sigma(\eta_{ijk}) + (1-y_{ijk}) \ln(1-\sigma(\eta_{ijk})) \\
&= \sum_k \sum_i \sum_{j<i} -y_{ijk} \ln(1+\exp(-\eta_{ijk})) + (1-y_{ijk}) \ln\left(\frac{\exp(-\eta_{ijk})}{1+\exp(-\eta_{ijk})}\right) \\
&= \sum_k \sum_i \sum_{j<i} -y_{ijk} \ln(1+\exp(-\eta_{ijk})) + (1-y_{ijk})[-\eta_{ijk} - \ln(1+\exp(-\eta_{ijk}))] \\
&= \sum_k \sum_i \sum_{j<i} -y_{ijk} \ln(1+\exp(-\eta_{ijk})) - \eta_{ijk} - \ln(1+\exp(-\eta_{ijk})) + y_{ijk}\eta_{ijk} + y_{ijk} \ln(1+\exp(-\eta_{ijk}))] \\
&= \sum_k \sum_i \sum_{j<i} (y_{ijk}-1)\eta_{ijk} - \ln(1+\exp(-\eta_{ijk}))
\end{aligned}$$

2.3 Regularization

While the notation z for the latent variables is consistent with the literature, and somewhat more intuitive, our contribution comes from a reparameterization of the model, where

$$z_{ik} = b_i + \epsilon_{ik}$$

This parameterization takes an explicit mid-point between the two most recently published papers on this problem. One fits the layers independently, and allows correlations between them [8]; another treats all of the layers simultaneously, as part of a single complex object [7]. Our model allows not only for similar behaviors to both of these extremes, but additionally allows the user to tune between them manually.

Our model starts with a "base" position b_i for each node, which is the hierarchical layer in the model. It then adds a layer-specific perturbation ϵ_{ik}. The behavior of the model then depends entirely on the regularization placed on the ϵ's. When there is none, each layer is fit independently. When there is an arbitrarily large amount, the perturbations are all driven to zero, and all k layers are represented identically by the base parameters b. With intermediate values however, the user can find the point between these two extremes that allows for the graphs to be distinct, but also renders them sufficiently similar to be comparable, by not allowing the perturbations to stray too far from their shared base position.

Furthermore, with the use of a group lasso penalty applied to all of the perturbations within a single layer k, the model itself will perform graph-wise dimensionality reduction. This is far more valuable to the practitioner than an element-wise sparse solution, which would still require them to consider all of the many layers in their multigraph.

3 Fitting

3.1 The Optimization Problem

Together these two elements form our optimization problem, which is to minimize the sum of the negative log likelihood and a penalty on the deviations ϵ_{ik}. We tested two approaches with respect to the penalty on the deviations ϵ_{ik}. In the first model, we used the standard element-wise lasso, by penalizing the L1 norm of each deviation ϵ_{ik}. This first model is represented by Equation 1.

$$\min_{\epsilon,b} \sum_k \sum_i \sum_{j<i} [(1 - y_{ijk})\eta_{ijk} + \ln(1 + \exp(\eta_{ijk}))] + \lambda \sum_{k,i} \|\epsilon_{ik}\|_1 \tag{1}$$

In the second approach, we used a group lasso penalty on each layer k thus encouraging sparsity over layers, instead of on each individual deviation ϵ_{ik}. This approach is illustrated in Equation 2.

$$\min_{\epsilon,b} \sum_k \sum_i \sum_{j<i} [(1 - y_{ijk})\eta_{ijk} + \ln(1 + \exp(\eta_{ijk}))] + \lambda \sum_{k} \|\epsilon_{k}\|_2 \tag{2}$$

Since both these objective functions are not differentiable, we have several options, but the clear first choice is to use proximal gradient descent.

3.2 Derivation of the Gradients

Then, the gradients of the η_{ijk} are:

$$\frac{\partial \eta_{ijk}}{\partial b_i} = -2(b_i + \epsilon_{ik} - b_j - \epsilon_{jk})$$

$$\frac{\partial \eta_{ijk}}{\partial b_j} = 2(b_i + \epsilon_{ik} - b_j - \epsilon_{jk})$$

$$\frac{\partial \eta_{ijk}}{\partial \epsilon_{ik}} = -2(b_i + \epsilon_{ik} - b_j - \epsilon_{jk})$$

$$\frac{\partial \eta_{ijk}}{\partial \epsilon_{jk}} = 2(b_i + \epsilon_{ik} - b_j - \epsilon_{jk})$$

$$\begin{aligned}
\frac{\partial \ell}{\partial b_m} &= \sum_k \sum_i \sum_{j<i} (y_{ijk} - 1) \frac{\partial \eta_{ijk}}{\partial b_m} - \frac{\partial \ln(1 + \exp(-\eta_{ijk}))}{\partial b_m} \\
&= \sum_k \left[\sum_{j<m} (y_{mjk} - 1) \frac{\partial \eta_{mjk}}{\partial b_m} - \frac{\partial \ln(1 + \exp(-\eta_{mjk}))}{\partial b_m} + \sum_{i>m} (y_{imk} - 1) \frac{\partial \eta_{imk}}{\partial b_m} - \frac{\partial \ln(1 + \exp(-\eta_{imk}))}{\partial b_m} \right] \\
&= \sum_k \Bigg[-2 \sum_{j<m} \left[(y_{mjk} - 1)(b_m + \epsilon_{mk} - b_j + \epsilon_{jk}) + \frac{\exp(-\eta_{mjk})}{1 + \exp(-\eta_{mjk})} (b_m + \epsilon_{mk} - b_j - \epsilon_{jk}) \right] \\
&\qquad + 2 \sum_{i>m} \left[(y_{imk} - 1)(b_i + \epsilon_{ik} - b_m - \epsilon_{mk}) + \frac{\exp(-\eta_{ijk})}{1 + \exp(-\eta_{mjk})} (b_i + \epsilon_{ik} - b_m - \epsilon_{mk}) \right] \Bigg] \\
&= 2 \sum_k \Bigg[- \sum_{j<m} \left(y_{mjk} - 1 + \frac{\exp(-\eta_{mjk})}{1 + \exp(-\eta_{mjk})} \right) (z_{mk} - z_{jk}) \\
&\qquad + \sum_{i>m} \left(y_{imk} - 1 + \frac{\exp(-\eta_{ijk})}{1 + \exp(-\eta_{mjk})} \right) (z_{ik} - z_{mk}) \Bigg]
\end{aligned}$$

$$\frac{\partial \ell}{\partial \epsilon_{mk}} = 2\Bigg[-\sum_{j<m} \left(y_{mjk} - 1 + \frac{\exp(-\eta_{mjk})}{1+\exp(-\eta_{mjk})} \right) (z_{mk} - z_{jk}) + \sum_{i>m} \left(y_{imk} - 1 + \frac{\exp(-\eta_{ijk})}{1+\exp(-\eta_{mjk})} \right) (z_{ik} - z_{mk}) \Bigg]$$

3.3 The Proximal Operator

3.3.1 Group lasso

The proximal operator for the penalty term using the L2 norm[1] is:

$$\text{prox}_{\|.\|,\lambda t}(\epsilon_k) = \begin{cases} \frac{\|\epsilon_k\| - \lambda t}{\|\epsilon_k\|} \epsilon_k, & \|\epsilon_k\| \geq \lambda t \\ 0, & \|\epsilon_k\| < \lambda t \end{cases}$$

$$\text{prox}_{\|.\|,\lambda t}(b_i) = b_i$$

Therefore, let β be the vector with all parameters ϵ and b. The Proximal Gradient Descent method in this case is:

$$\beta^+ = \text{prox}_{\|.\|,\lambda t}(\beta - t\nabla \ell(\beta))$$

3.3.2 Standard lasso

The proximal operator for the penalty term using the L1 norm is:

$$\text{prox}_{\|.\|_1,\lambda t}(\epsilon_{i,k}) = \text{sign}(\epsilon_{i,k}) \max(\epsilon_{i,k} - \lambda t, 0)$$

$$\text{prox}_{\|.\|_1,\lambda t}(b_i) = b_i$$

Therefore, let β be the vector with all parameters ϵ and b. The Proximal Gradient Descent method in this case is:

$$\beta^+ = \text{prox}_{\|.\|_1,t}(\beta - t\nabla \ell(\beta))$$

3.4 Non-Convexity and Initialization

The original Latent Space Model is known to be non-convex in the latent positions z [1], and our model is no different. This issue is typically approached by using MCMC samplers [1, 8] which are capable of exploring non-convex spaces, however these are known to have issues as well. Using them on this problem proved not only to take far more time, but also to have substantially worse results. We restrict our discussion to the non-Bayesian approaches taken.

As we use methods designed for convex problems, we start by using a careful initialization of our procedure. This attempts to begin the algorithm close to a global minimum of the objective, so that when we do descend the likelihood, we descend a good one, knowing that local minima exist. This initialization was composed of several steps.

1. Create a proposal 'base' graph from $\hat{y}_{ij} = \sum_k y_{ijk} > 0$
2. Fit a separate single-graph latent space model to each layer
3. Transform the layer estimates to minimize their distance from the base estimate

[1] Using the results from Homework 3

The third step in this procedure was initially done with a Procrustes transform [1], as is used in Bayesian estimation of Latent Space Models. This transform is used because proposal graphs in a sampling procedure can produce a graph with many different unimportant variations, as the likelihood is invariant to translation, rotation, scaling, and flipping. The Procrustes transform is thus used to remove these transformations, while keeping the variations that are of importance to the likelihood. However this transform minimizes the overall distance between *all* pairs of points in two shapes, and in this problem, we require something slightly different. The goal here is to minimize only distances between positions that correspond to the same node – this is the property that minimizes the ϵ's, and makes the embeddings comparable.

To solve this problem we developed and propose the Anticrustean transformation. A closed form solution, should one exist, has not been found, but a simple implementation combines line searches over the several classes of transforms to which the likelihood is invariant. Together, these steps produce a good initial estimate of all of the models parameters, which are used as starting points in the proximal gradient procedure.

4 Results

To test our model, we simulate a toy data set consisting of a multigraph with $N = 20$ vertices and $K = 2$ layers. We initialize our optimization problem using the initialization procedure described in the previous section. We then run 3000 iterations of the proximal gradient algorithm. Figures 1 and 2 show the improvement in the objective function value over the 3000 iterations for, respectively, the standard lasso and the group lasso approaches using different values for λ. We can observe that the function values appear to converge to a local minimum. In both approaches we used a fixed step size $t = 10^{-3}$ in the proximal gradient algorithms.

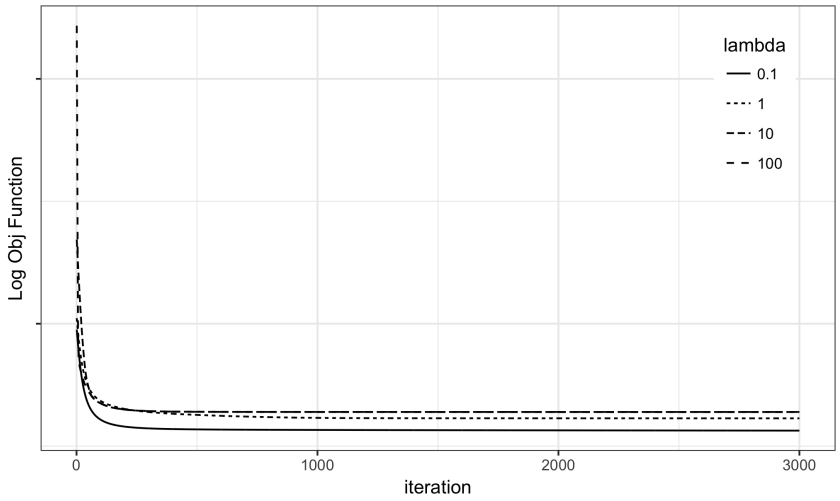

Figure 1: Value of the objective function for different values of λ in the standard lasso approach. We ran 3000 iterations of the proximal gradient descent algorithm.

Figures 3 and 4 compare the optimal positions found by each of the models with the initial one. Black points represent the base positions b_i and the red and green points (and lines) represent the two different layers of the graph. We can observe that as we increase the value of λ deviations are more penalized and the optimal solution tends to be one where all layers collapse to a single graph. Additionally, we can also compare the resulting optimal positions between the two approached (standard lasso and group lasso). For $\lambda = 1$ the two layers are significantly more distinct than the ones in the standard lasso case.

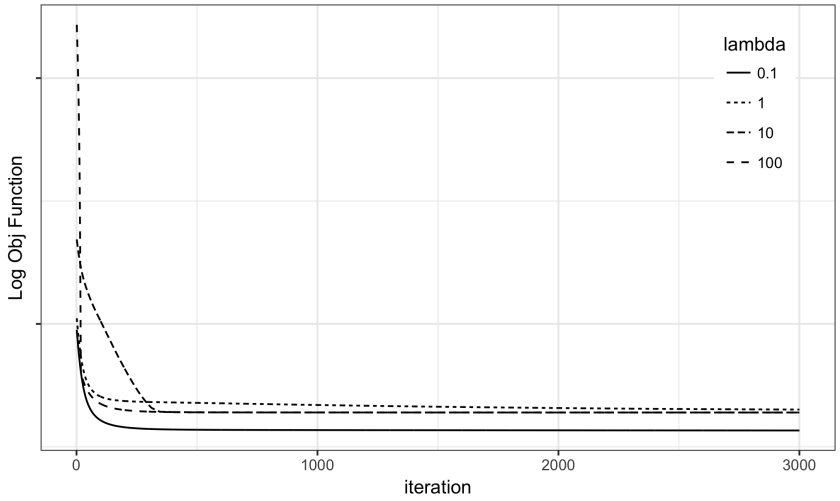

Figure 2: Value of the objective function for different values of λ in the group lasso approach. We ran 3000 iterations of the proximal gradient descent algorithm.

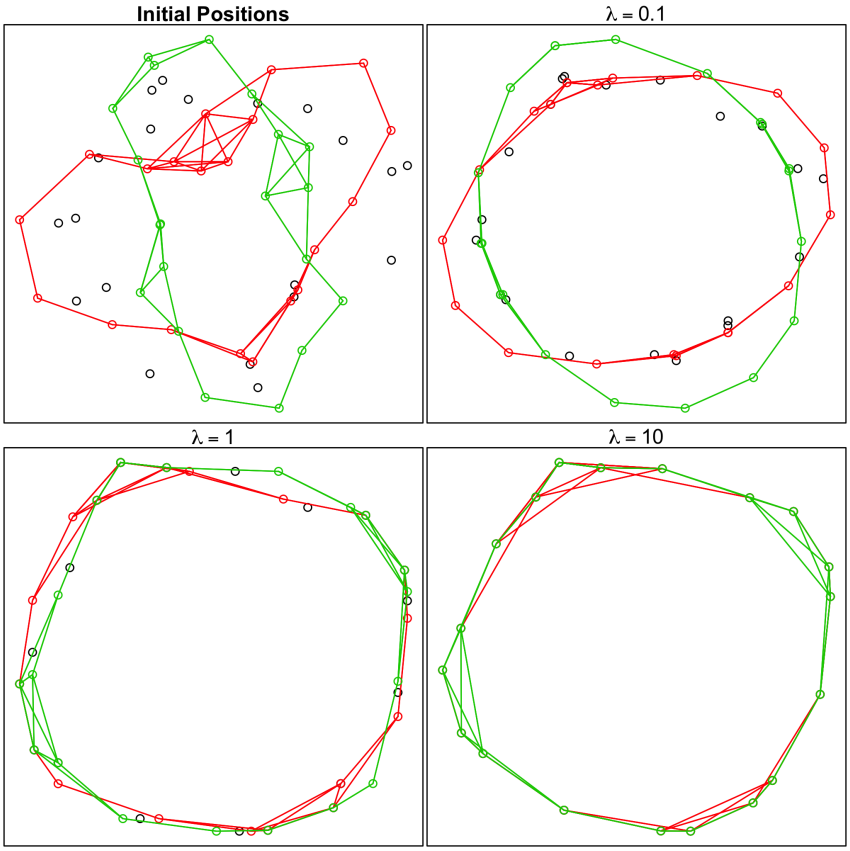

Figure 3: Scatter plots showing the initial positions (top left plot) used in the optimization problem and the final ones for different values of λ in the standard lasso approach. Black points represent the base positions b_i and the red and green points (and lines) represent the two different layers of the graph. As the value of λ increases deviations are more penalized and the optimal solution tends to be one where all layers collapse to a single graph.

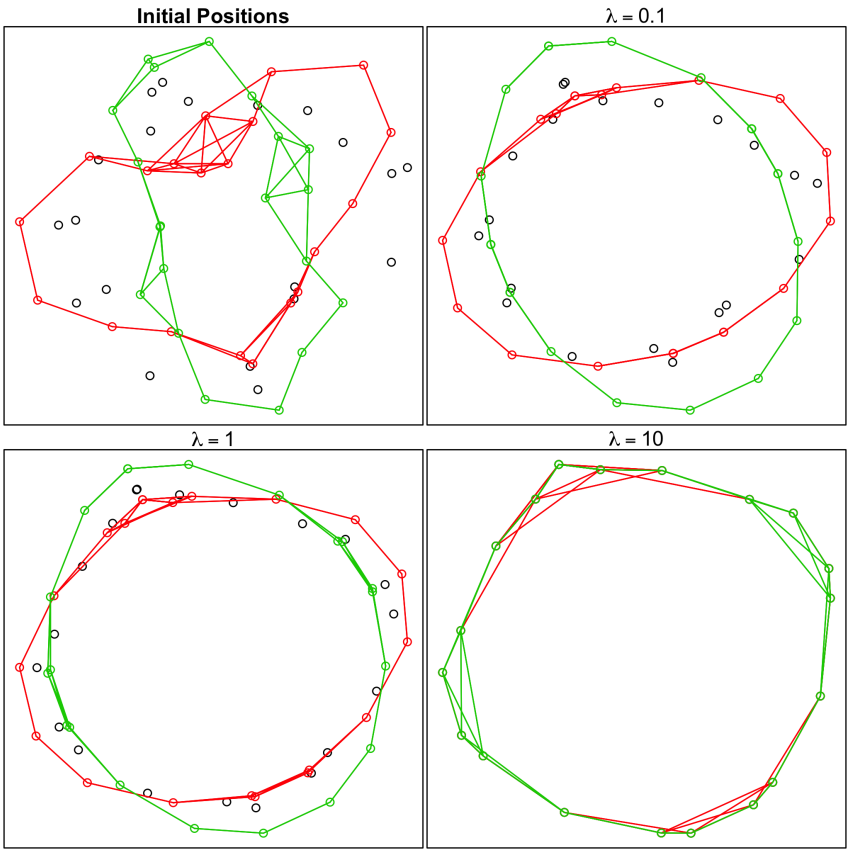

Figure 4: Scatter plots showing the initial positions (top left plot) used in the optimization problem and the final ones for different values of λ in the group lasso approach. Black points represent the base positions b_i and the red and green points (and lines) represent the two different layers of the graph. As the value of λ increases deviations are more penalized and the optimal solution tends to be one where all layers collapse to a single graph. We can observe that for $\lambda = 1$ the two layers are significantly more distinct than the ones in the standard lasso case.

5 Conclusion

Despite not being a convex problem, a combination of convex methods and a good heuristic starting point have been shown to efficiently fit this novel class of models. In addition, they have demonstrated all of the properties for which they were intended, and several others additionally. As demonstrated, HLSMs are capable of producing comparable graph layouts for the layers of a multigraph, and are also capable of performing layer-wise dimensionality reduction in order to meaningfully simplify this complex form of data.

They are also valuable for refining the layouts determined by current Latent Space Model implementations, as can been seen in the differences between the initializations, and the final estimates. While it is possible that many likelihoods are feasible from a numerical point of view, this further convex optimization is valuable in finding solutions that are meaningful to human interpreters. Furthermore, our fitting procedure was found to converge to a good solution far faster than the HMC implementation with which we compared it.

Having demonstrated the basic feasibility, effectiveness, and utility, of HLSMs, the next steps will be to push its limits on different kinds of graphs, and different numbers of layers. Several different guests at our poster session mentioned additional problems, including in medical domains, and the interpretability of deep neural nets, to which our method might be able to contribute. We look forward to exploring these additional problem spaces.

References

[1] Peter D Hoff, Adrian E Raftery, and Mark S Handcock. Latent space approaches to social network analysis. *Journal of the American Statistical Association*, 97(460):1090–1098, 2002. doi: 10.1198/016214502388618906. URL https://doi.org/10.1198/016214502388618906.

[2] Jerome Friedman and Trevor Hastie. Regularization Paths for Generalized Linear Models via Coordinate Descent. pages 1–22, 2008.

[3] Jerome Friedman, Trevor Hastie, and Robert Tibshirani. A note on the group lasso and a sparse group lasso arXiv : 1001 . 0736v1 [math . ST] 5 Jan 2010. pages 1–8, 2010.

[4] Martin Vincent and Niels Richard Hansen. Sparse group lasso and high dimensional multinomial classification. pages 1–31.

[5] Zhiwei Tony Qin and Katya Scheinberg. Efficient Block-coordinate Descent Algorithms for the Group Lasso. pages 1–23.

[6] Tong Tong Wu and Kenneth Lange. Coordinate descent algorithms for lasso penalized regression. 2(1):224–244, 2008. doi: 10.1214/07-AOAS147.

[7] Isabella Gollini and Thomas Brendan Murphy. Joint modeling of multiple network views. *Journal of Computational and Graphical Statistics*, 25(1):246–265, 2016. doi: 10.1080/10618600.2014.978006. URL https://doi.org/10.1080/10618600.2014.978006.

[8] Michael Salter-Townshend and Tyler H. McCormick. Latent space models for multiview network data. *Ann. Appl. Stat.*, 11(3):1217–1244, 09 2017. doi: 10.1214/16-AOAS955. URL https://doi.org/10.1214/16-AOAS955.

Appendix: Link to R scripts

The proximal descent methods described in this report were implemented using the R language. All scripts are available in the following git hub repository: https://github.com/amloewi/hlsm. The main R script is located at https://github.com/amloewi/hlsm/blob/master/R/final_project.R