Hierarchical Latent Space Models for Multiplex Social Network Analysis

Alexander Martin Loewi
aloewi@cmu.edu

April 16, 2018

Abstract

Background Multigraphs are an extremely common form of network data, in which several edge types are defined over a single set of nodes. Despite its ubiquity, there are relatively few network models designed to handle this data, and its particular challenges.

Aim We have two major objectives with this novel method. One is to be able to tune explicitly between the two extremes of flexibility, and comparability, when finding an optimal layout for each of the layers of a multigraph. The other is to be able to remove unnecessary layers, and thus remove unnecessary complexity from this complex data format.

Data The data used is of two kinds. First, because a novel method is developed, simulations are carried out on a variety of synthetic graphs, in order to understand the properties of the model in as controlled a manner as possible. Second, the model is fit to a novel five-layer dataset of the staff at a charter school in West Baltimore.

Method The method developed here is based on the widely used Latent Space Model, but both reparameterizes and regularizes it, to allow for several novel properties. The model is fit with a combination of a heuristic starting point, and proximal gradient descent, using the group lasso.

Results Rather than finding a consistent trade-off between comparability and accuracy, we found that many networks can be made far more comparable without sacrificing any accuracy.

Conclusion The basic functionality of method has been demonstrated in a variety of synthetic, and organic contexts, and has already suggested that many multigraphs could be made easier to visualize without any compromise in accuracy of representation.

Intellectual merit This method allows network data that was previously prohibitively complex to be examined in an intuitive manner, taking a full pattern into account all at once, instead of examining the pieces independently.

Broader impacts The ability of this method to act as a cognitive aid in complex visualizations has already inspired applications in medical diagnostics, and deep learning interpretation.

Keywords: latent space model; social network analysis; multigraph

1 Introduction

1.1 The Difficulty and Ubiquity of 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). If the population is static over time, longitudinal networks are also easily representable as multigraphs. Despite the widespread existence of data of this form its inherent difficulty has led to only a small number of models for it, all of which have important shortcomings for the practicing social network analyst.

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 layers of data (i.e. the graphs formed by the different edge types). To be expressive, a graph layout needs to be flexible in how it places nodes – however in order to be comparable, two layouts need to place nodes in relatively similar places. A useful multigraph model would be able to tune explicitly between these two extremes of flexibility, and comparability.

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.

1.3 Previous Work

Our model combines several methods that have proven to be exceptionally valuable – Latent Space Models, [**?**], the LASSO estimator, [Friedman and Hastie(2008)], and in particular the group LASSO [**?**]. The substantial amount of work that has

been done on them all 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, [Friedman and Hastie(2008)], to generalized linear models [Friedman et al.(2010)Friedman, Hastie, and Tibshirani], with the element-wise, group, and combined, sparse group LASSO, [Vincent and Hansen(), Qin and Scheinberg(), Wu and Lange(2008)], and more.

On the social network methodology side, two recent papers have been published that take the general approach of extending LSMs to multigraph data. Our approach has important advantages over both of them. One paper takes the extreme of modeling the multigraph as a single object, which only makes visualization more difficult [Gollini and Murphy(2016)]. The other treats each layer as independent, and allows correlations between them. While a highly intuitive idea, this does not solve the comparability problem, or that of dimensionality reduction [Salter-Townshend and McCormick(2017)]. Furthermore, there is evidence that the extreme flexibility of this model leads to a substantial risk of overfitting. While the fact that ours is less flexible is not an unmitigated blessing, we believe it has benefits in interpretability, and it can also rely on its regularization to avoid the problem encountered by [Salter-Townshend and McCormick(2017)].

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 between comparability and expressiveness
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

$$\ell = \sum_k \sum_i \sum_{j<i} (y_{ijk} - 1)\eta_{ijk} - \ln(1 + \exp(-\eta_{ijk}))$$

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 re-parameterization of the model, where

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

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 without regard to the others, as each ϵ can be arbitrarily large, and thus does not need to rely on the base position in its representation. When there is an arbitrarily large amount of regularization, the perturbations are all driven to zero, and all k layers are represented identically by the base parameters b. With intermediate values, 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 that groups together all of the perturbations within a single layer k, the model will perform graph-wise dimensionality reduction, by setting a redundant layer equal to the base layer. This is far more valuable to the practitioner than an element-wise sparse solution, which would still require them to consider all of the 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 an element-wise lasso, by penalizing the norm of each deviation ϵ_{ik}. This can be seen in by Equation 1. It is important to note that the latent parameters are all two-dimensional, to allow for visualization. Consequently even with this element-wise approach, the group-lasso approach and the L2 penalty is necessary, following [**?**]. However the grouping is only over the two dimensions of the individual parameter.

$$\min_{b,\epsilon} -\ell(b,\epsilon) + \lambda \sum_i \sum_k \|\epsilon_{ik}\|_2 \tag{1}$$

In the second approach, we used a group lasso penalty that is over all of the perturbations ϵ_k within each layer k, instead of on each individual deviation ϵ_{ik}. This encourages sparsity over the layers, meaning an entire graph might be zeroed out (its perturbations driven to zero, leading it to be expressed as just the base layer). This approach is illustrated in Equation 2.

$$\min_{b,\epsilon} -\ell(b,\epsilon) + \lambda \sum_k \|\epsilon_k\|_2 \tag{2}$$

Because neither of these objective functions is differentiable, we use proximal gradient descent in fitting the model.

3.2 Derivation of the Gradients

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

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

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

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

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

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

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

3.3 The Proximal Operator

The proximal operator for the penalty term using the L2 norm 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:

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

3.4 Non-Convexity and Initialization

The original Latent Space Model is stated in the original paper as non-convex in the latent positions z [**?**], and our model is assumed to be the same. This issue is typically resolved by using MCMC samplers [**?**, Salter-Townshend and McCormick(2017)]

which are capable of exploring non-convex spaces, however using them on this problem proved not only to take far more time, but also to have substantially worse results. We focus our discussion on the convex approaches used.

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 reduce our chances of being caught in a poor local minimum. 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 (including the base layer)
3. Transform the layer estimates to minimize their distance from the base estimate

The third step in this procedure was initially done with a Procrustes transform [**?**], 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 propose the Anticrustean transformation. A simple implementation combines line searches over the several classes of transforms to which the likelihood is invariant, finding in each line search the position that minimizes the sum of the squared distances between two instantiations of the same node. 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 Proof of Concept Results

To test our model, we first simulate a toy data set consisting of a multigraph with $N = 20$ vertices and $K = 2$ layers, followed by the real and novel data set that motivated this technique. 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 element-wise lasso and the layer-wise 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.

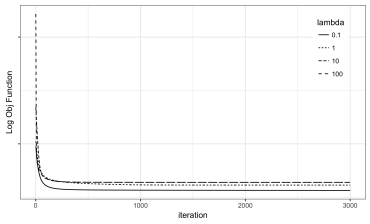

Figure 1: Value of the objective function for different values of λ in the standard lasso approach. We ran 3000 iterations of the proximal gradient descent algorithm.

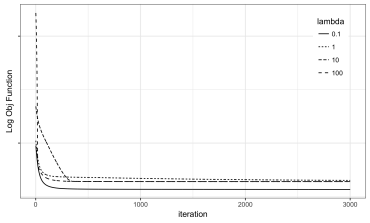

Figure 2: Value of the objective function for different values of λ in the group lasso approach. We ran 3000 iterations of the proximal gradient descent algorithm.

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.

In Figure 5, we plot the results of our algorithm on a novel data set of teachers at a school in West Baltimore. As can be seen, the graphs are highly visually comparable, while also maintaining a substantial amount of expressiveness to demonstrate their individual structure. We believe this convincingly demonstrates the potential utility of our technique as a tool in social network analysis, as well as many other fields.

5 More Extensive Simulation Results

While the toy model is compelling, it is clear that a more thorough examination of the properties of limits of this method will require a richer variety of test cases.

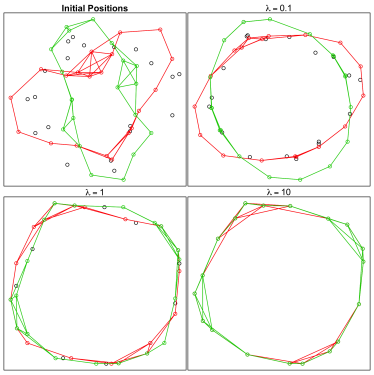

Figure 3: Scatter plots showing the initial positions (top left plot) used in the optimization problem and the final ones for different values of λ in the standard lasso approach. Black points represent the base positions b_i and the red and green points (and lines) represent the two different layers 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.

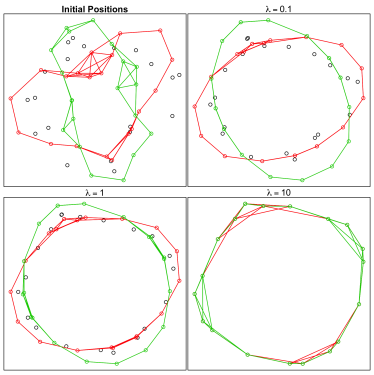

Figure 4: Scatter plots showing the initial positions (top left plot) used in the optimization problem and the final ones for different values of λ in the group lasso approach. Black points represent the base positions b_i and the red and green points (and lines) represent the two different layers of 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.

The first possibility are of course Erdos-Renyi graphs, but they have serious drawbacks. The value of this model comes only from its being able to represent several layers that are somewhat, or even very, different – there is no value in a model that only models identical layers. To test carefully the degree to which

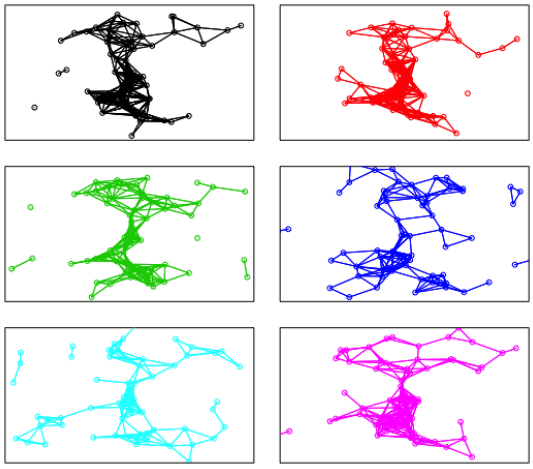

Figure 5: A Real-World Network of Teachers in West Baltimore

layers can differ, and still be well represented by the HLSM, it is necessary to be able to introduce precise amounts of disturbances into a graph – this allows a model to be fit to g and g', where g' is a known and controllable distance from g. Unfortunately with completely random Erdos-Reyni graphs, shuffling a single edge could result in its being moved to any other location in the graph – possibly fundamentally changing the pattern of connectivity, such as by disconnecting two components. This "small" difference thus would have large repercussions for the graph, and so perhaps the model. This was considered too unpredictable.

5.1 Blockmodels as a Model Graph Type

The solution to this problem was to use blockmodels as the substrate for testing, and when shuffling edges, only shuffling them within blocks. In other words, one shuffle consists of removing an edge between two nodes in group 1, and placing it instead between two other nodes – both also in group 1. In this way, Differences can be introduced to a graph while explicitly maintaining the overall structure, and thus ensuring comparability.

There were two more points of consideration in using blockmodels. The first was whether they were sufficiently realistic as models for real-world networks. However while simple, the blockmodel has been a staple of network analysis for several decades, and block structures are extremely common empirical attributes of real-world networks. Their simplicity and power was thus considered a perfect combination for this first round of explorations.

The second point was that once blocks are determined, it is not clear that they should be treated identically. Similar to the point raised above with respect to Erdos-Renyi graphs, some edges have a greater effect on the overall structure of a graph than others, and this point is identical for blocks in graphs. However

as a first pass, every edge was treated as contributing equally to the accuracy of the final representation.

The test cases were thus composed of graphs drawn from manually specified blockmodels, each of 36 nodes, with a density of 0.1 on the off-diagonal elements, and of 0.6 on-diagonal. Variations were made by randomly shuffling a small number of edges, by having different numbers of blocks, and of having the blocks be of differing sizes. Thus one layer may have been evenly split into two groups of similar sizes, while the other layer in the same multigraph would have three groups, one large, and two small. These were then fit over a range of lambda values, and the true-positive and true-negative rates of the model in representing the edges of the multigraph were recorded for each layer.

5.2 A Surprising Lack of Effect on Accuracy

It was initially presumed that the more regularization was applied, the more the modeled graph would differ from the observed graph. To test this, both the true positive and true negative rates were assessed, for each layer of the multigraph, and at each value of lambda for which the model was fit. The TPR is plotted as a solid line, the TNR as a dashed-and-dotted line, and each is colored with the same color as its corresponding layer. One full example can be seen in Figure 6

However as can be seen in the representative example, there is no real observed loss in accuracy until the very end, at the highest values of regularization. There has still been a substantial amount of alignment between the layers though, as can be seen when comparing the first pane (the initialization point) and last. This was a consistent finding across a number of graph layer combinations, and suggests that there is an unnecessary flexibility in many LSM representations. This was not unknown, but there had also not been a need to deal with it. Here, we demonstrate the approximate amount of regularization that can be applied before the modeled graph differs more from the observed graph than it would when fit as its own layer.

6 Conclusion

A combination of convex methods and a good heuristic starting point have been demonstrated to fit this novel class of models in a way finds a local minimum, and that produces a model fit with desirable visual properties.

In addition, it was found that multigraphs can often be represented in very parsimonious ways, if the layers are drawn from a classic family of graphs. For these, regularization is valuable typically to a point immediately prior to the elimination of a layer, after which accuracy drops modestly, but noticeably. Prior to this, there are often very slight, zero, or even positive changes in accuracy as regularization increases. This suggests that there is substantial flexibility in the way a single LSM could be displayed, and that rather than finding a compromise between multiple conflicting layers, a large amount of the behavior of this model is simply in eliminating unnecessary variation between the representations.

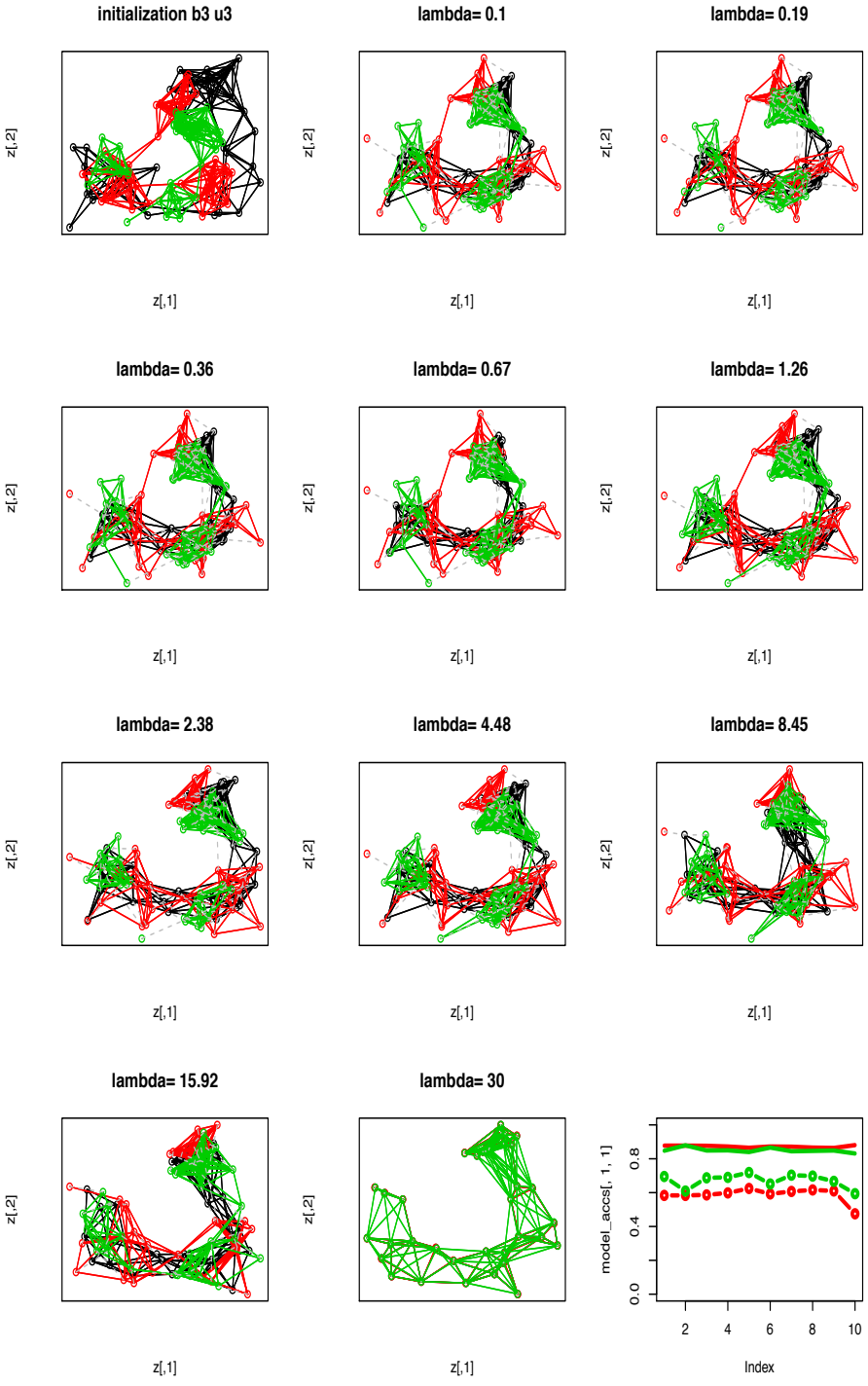

Figure 6: One Multigraph Fit to Varying Values of Lambda, and Accuracies

Future work will attempt to understand on a deeper level when a multigraph can be reconciled and when it can't, possibly finding heuristics to predict whether or not this procedure will be useful even before it is run. In addition, the accuracy will be weighted by block size and density, to emphasize edges based on their importance to the graph structure – it can be seen that weighting each edge equally is likely undesirable. A compelling candidate is to weight by surprise – an error of omission or commission could be weighted by how surprising it is, which is a known quantity, given the densities of the blocks in the model. In this way, errors in sparse blocks are more serious than errors in dense ones, making the model better at representing the crucial bridging structures that determine so much of a global networks behavior.

Acknowledgments

Michael Salter-Townshend and Tyler McCormick were kind enough to give very helpful feedback on my idea, as well as to describe some of the shortcomings of their model that were important to address in a new one. While the author built two parallel implementations of fitting procedures, the one that worked best, and the implementation that is being used here, was written by Francisco Fonseca. The idea for the model, including the parameterization, regularization, and initialization, plus the testing done, are the author's alone.

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`

References

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

[Friedman et al.(2010)Friedman, Hastie, and Tibshirani] 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.

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

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

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

[Gollini and Murphy(2016)] 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`.

[Salter-Townshend and McCormick(2017)] 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`.