Project

Numerical Results: Given a network structure, we generate a covariance matrix for the first class as follows (Peng et al. 2009). We create a $p \times p$ matrix with ones on the diagonal, zeroes on elements not corresponding to network edges, and values from a uniform distribution with support on $\{[-.4, -.1] \cup [.1, .4]\}$ on elements corresponding to edges. To ensure positive definiteness, we divide each off-diagonal element by 1.5 times the sum of the absolute values of off-diagonal elements in its row. Finally, we average the matrix with its transpose, achieving a symmetric, positive-definite matrix A. We then create the (i,j) element of Σ_1 as

$$d_{ij}(A^{-1})_{ij}/\sqrt{(A^{-1})_{ii}(A^{-1})_{jj}}$$

where $d_{ij} = 0.6$ if $i \neq j$ and $d_{ij} = 1$ if $i \neq j$. We create Σ_1 equal to Σ_1,then reset one of its ten subnetwork blocks to the identity. We create Σ_3 equal to Σ_2, and reset an additional subnetwork block to the identity. Finally, for each class we generate independent, identically distributed samples from a $\mathcal{N}(0, \Sigma_k)$ distribution.

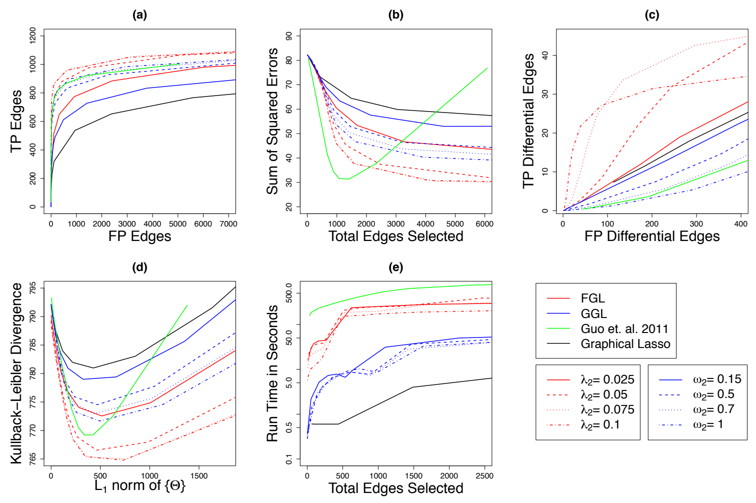

Figure 1: Performance of FGL, GGL, Guo et al. (2011)s method, and the graphical lasso on simulated data with 150 observations in each of 3 classes, and 500 features. Black lines display models derived using the graphical lasso, green lines display the proposal of Guo et al. (2011), red lines display FGL, and blue lines display GGL. (a): The number of edges correctly identified to be nonzero (TP Edges) is plotted against the number of edges incorrectly identified to be nonzero (FP edges). (b): The sum of squared errors in edge values is plotted against the total number of edges estimated to be nonzero. (c): The number of edges correctly found to have values differing between classes (TP Differential Edges) is plotted against the number of edges incorrectly found to have values differing between classes (FP Differential Edges). (d): The dKL of the estimated models from the true models is plotted against the L_1 norm of the off-diagonal entries of the estimated precision matrices. (e): Running time (in seconds) is plotted against the number of non-zero edges estimated. Note the use of a log scale on the y-axis.

FGL uses λ_1 and λ_2 but for GGL they reparametrize the results in terms of $\omega_1 = \lambda_1 + \frac{1}{\sqrt{2}}\lambda_2$ and $\omega_2 = \frac{1}{\sqrt{2}}\lambda_2/(\lambda_1 + \frac{1}{\sqrt{2}}\lambda_2)$. The main simulation study generates three networks with p = 500 features belonging to ten equally sized unconnected subnetworks, each with a power law degree distribution, i.e. a scale-free network. Of the ten subnetworks, eight have the same structure and edge values in all three classes, one is identical between the first two classes and missing in the third (i.e. the corresponding features are singletons in the third network), and one is present in only the first class.

Our first set of simulations illustrates the effect of varying tuning parameters on the performances of FGL and GGL. We generated 100 three-class data sets with p = 500 features and n = 150 observations per class, as described in Section 7.1.1. Class 1s network had 490 edges, class 2s network is missing 49 of those edges, and class 3s network is missing an additional 49 edges. Figure 2 shows the results, averaged over the 100 data sets. In each plot, the lines for FGL and for GGL indicate the results obtained with a single value of the similarity tuning parameters λ_2, and ω_2. The graphical lasso and the proposal of Guo et al. (2011) are included in the comparisons. For all the graphs, as λ_1 or ω_1 increases sparsity increases, or equivalently the number of edges selected decreases. So essentially

Figure 2(a) displays the number of true edges selected against the number of false edges selected. As the sparsity tuning parameters λ_1 and ω_1 decrease, the number of edges selected increases. At many values of the similarity tuning parameter λ_2, FGL dominates the other methods. At some choices of the similarity tuning parameter ω_2, GGL performs as well as Guo et al. (2011). FGL, GGL, and Guo et al. (2011)s proposal dominate the graphical lasso.

Figure 2(b) displays the sum of squared errors (SSE) between estimated edge values and true edge values:

$$SSE = \sum_{k=1}^{K} \sum_{i \neq j} (\hat{\theta}_{ij}^{k} - (\Sigma_k^{-1})_{ij})^2$$

Unlike the proposal of Guo et al. (2011), FGL, GGL, and the graphical lasso tend to overshrink edge values towards zero due to the use of convex penalty functions. Thus, while FGL and GGL attain SSE values that are as low as those of Guo et al. (2011), they do so when estimating much larger networks. When simultaneous edge selection and estimation are desired, it may be useful to run FGL or GGL once and then to re-run them with smaller penalties on the selected edges, as in Meinshausen (2007).

Figure 2(c) evaluates each methods success in detecting differential edges, or edges that differ between classes. For FGL, it is computed as the number of pairs $k < k', i < j$ such that $\hat{\theta}_{ij}^{(k)} \neq \hat{\theta}_{ij}^{(k')}$. Since GGL, the proposal of Guo et al. (2011), and the graphical lasso cannot yield edges that are exactly identical across classes, for those approaches the number of differential edges is computed as the number of pairs $k < k', i < j$ such that $|\hat{\theta}_{ij}^{(k)} - \hat{\theta}_{ij}^{(k')}| > 10^{-2}$. The number of true positive differential edges is plotted against the number of false positive differential edges. Note that by controlling the total number of non-zero edges, the sparsity tuning parameters λ_1 and ω_1 have a large effect on the number of edges that are estimated to differ between the two networks. FGL yields fewer false positives than the competing methods, since it shrinks between-class differences in edge values to zero. Since neither GGL nor Guo et al. (2011)s method are designed to shrink edge values towards each other, by this measure neither method outperforms even the graphical lasso.

Figure 2(d) displays the sum of the dKLs of the estimated distributions from the true distributions, as a function of the ℓ_1 norm of the off-diagonal elements of the estimated precision matrices, i.e. $\sum_k \sum_{i\neq j} |\hat{\theta}_{ij}^{(k)}|$. The dKL from the multivariate normal model with inverse covariance estimates $\Theta^{(1)}, ..., \Theta^{(k)}$to the multivariate normal model with the true precision matrices $\Sigma^{(1)}, ..., \Sigma^{(k)}$ is

$$\frac{1}{2}\sum_{k=1}^{K}(-\log\det(\Theta^{(k)}\Sigma^{(k)}) + \mathrm{trace}(\Theta^{(k)}\Sigma^{(k)}))$$

At most values ofλ_2, FGL attains a lower dKL than the other methods, followed by Guo et al. (2011)s method, then by GGL. The graphical lasso has the worst performance, since it estimates each network separately.

The graphical lasso is fastest, but FGL and GGL are much faster than the proposal of Guo et al. (2011), due to the results from Section 4. Timing comparisons were performed on an Intel Xeon x5680 3.3 GHz processor. It is worth mentioning that the FGL algorithm is much faster in problems with only two classes, since in that case there is a closed-form solution to the generalized fused lasso problem (Section 3.2).

Table 1. *Performances as a function of n and p. Means over 100 replicates are shown for dKL, and for sensitivity (Sens.) and false discovery rate (FDR) of detection of edges (DE) and differential edge detection (DED).*

	p	n	dKL	DE Sens.	DE FDR	DED Sens.	DED FDR
FGL	500	50	545.1	0.502	0.966	0.262	0.996
		200	517.5	0.570	0.053	0.228	0.485
		500	516.6	0.590	0.001	0.192	0.036
	1000	50	1119.3	0.600	0.970	0.245	0.998
		200	1035.0	0.666	0.063	0.223	0.557
		500	1033.3	0.681	0.000	0.194	0.025
GGL	500	50	549.8	0.490	0.973	0.337	0.996
		200	520.8	0.505	0.060	0.244	0.903
		500	519.7	0.524	0.010	0.194	0.921
	1000	50	1127.9	0.587	0.976	0.316	0.998
		200	1041.7	0.615	0.061	0.239	0.908
		500	1039.4	0.629	0.007	0.197	0.920

In addition to the 500-feature network pair, we generate a pair of networks with p = 1000 features, each of which is block diagonal with 500×500 blocks corresponding to two copies of the 500-feature networks just described.

Let TP = true positives ($\Omega_{ij} = 1$ and $\hat{\Omega}_{ij} = 1$) and FN = false negatives ($\Omega_{ij} = 1$ and $\hat{\Omega}_{ij} = 0$) and P = total positives. ($\Omega_{ij} = 1$)

$$\begin{aligned}
\text{Sensitivity} &= TPR \\
&= \frac{TP}{P} \\
&= \frac{TP}{TP + FN} \\
&= \frac{P(\Omega_{ij} = 1, \hat{\Omega}_{ij} = 1)}{P(\Omega_{ij} = 1, \hat{\Omega}_{ij} = 1) + P(\Omega_{ij} = 1, \hat{\Omega}_{ij} = 0)} \\
&= \frac{P(\Omega_{ij} = 1, \hat{\Omega}_{ij} = 1)}{P(\Omega_{ij} = 1, \hat{\Omega}_{ij} = 1) + P(\Omega_{ij} = 1, \hat{\Omega}_{ij} = 0)} \\
&= \frac{P(\Omega_{ij} = 1, \hat{\Omega}_{ij} = 1)}{P(\Omega_{ij} = 1)} \\
&= P(\hat{\Omega}_{ij} = 1 | \Omega_{ij} = 1).
\end{aligned}$$

Sentivity is the proportion of $\Omega_{ij} = 1$ that give us $\hat{\Omega}_{ij} = 1$. High Sensitivity is good.

Let FP = false positives ($\Omega_{ij} = 0$ but $\hat{\Omega}_{ij} = 1$).

$$\begin{aligned}
\text{False Discovery Rate} &= FDR \\
&= \frac{FP}{TP + FP} \\
&= \frac{P(\Omega_{ij} = 0, \hat{\Omega}_{ij} = 1)}{P(\Omega_{ij} = 1, \hat{\Omega}_{ij} = 1) + P(\Omega_{ij} = 0, \hat{\Omega}_{ij} = 1)} \\
&= \frac{P(\Omega_{ij} = 0, \hat{\Omega}_{ij} = 1)}{P(\hat{\Omega}_{ij} = 1)} \\
&= P(\Omega_{ij} = 0 | \hat{\Omega}_{ij} = 1)
\end{aligned}$$

Thus is is the probability of have no edge given that we predicted an edge. We want this to be as low as possible.

For differential edge detection it is the same thing.

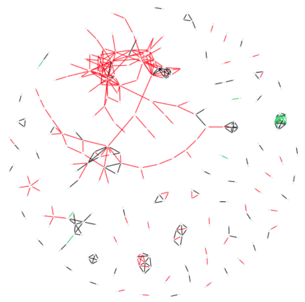

Figure 2: Conditional dependency networks inferred from 17,772 genes in normal and cancerous lung cells. 278 genes have nonzero edges in at least one of the two networks. Black lines denote edges common to both classes. Red and green lines denote tumor-specific and normal-specific edges, respectively.

Analysis of lung cancer microarray data: We applied FGL to a dataset containing 22,283 microarray-derived gene expression measurements from large airway epithelial cells sampled from 97 patients with lung cancer and 90 controls (Spira et al. 2007). We omitted genes with standard deviations in the bottom 20% since a greater share of their variance is likely attributable to non-biological noise. The remaining genes were normalized to have mean zero and standard deviation one within each class. To avoid disparate levels of sparsity between the classes and to prevent the larger class from dominating the estimated networks, we weighted each class equally instead of by sample size in (2.4). **This was odd. I'm not sure why they did this. Also the choice of λ_1 and λ_2 seems rather arbitrary.** Since our goal was data visualization and hypothesis generation, we chose a high value for the sparsity tuning parameter,$\lambda_1 = 0.95$, to yield very sparse network estimates. We ran FGL with a range of λ_2 values in order to identify the edges that differed most strongly, and settled on $\lambda_2 = 0.005$ as providing the most interpretable results. Application of Theorem 1 (**Theorem 1 discusses the conditions required for a partition of the graph vertices to be totally disconnected.**) revealed that only 278 genes were connected to any other gene using the chosen tuning parameters. Identification of block diagonal structure using Theorem 1 and application of the FGL algorithm took less than two minutes. (Note that this data set is so large that it would be computationally prohibitive to apply the proposal of Guo et al. (2011)!) FGL estimated 134 edges shared between the two networks, 202 edges present only in the cancer network, and 18 edges present only in the normal tissue network. Results are shown in Figure 2.

First thing was that they only tried one type of network. This seems questionable: To avoid disparate levels of sparsity between the classes and to prevent the larger class from dominating the estimated networks, we weighted each class equally instead of by sample size in (2.4). Since our goal was data visualization and hypothesis generation, we chose a high value for the sparsity tuning parameter, $\lambda_1 = 0.95$(high sparsity?), to yield very sparse network estimates.We ran FGL with a range of λ_2 values in order to identify the edges that differed most strongly, and settled on $\lambda_2 = 0.005$ (very little borrowing of structure?) as providing the most interpretable results. - I'm not sure why this is the case. Also these choice of λ do not seem to be in agreement with what they found in the simulation set-up.

	Group	
	$\lambda_1 = 0.07, \lambda_2 = 0.05$	$\lambda_1 = 0.05, \lambda_2 = 0.55$
F	0.05642315	0.06010037
FP	0.3391197	0.02918924
FN	0.3035487	0.8517427

Table 1: Nearest Neighbor Network for Group Lasso Penalty

	Fused	
	$\lambda_1 = 0.05, \lambda_2 = 0.05$	$\lambda_1 = 0.02, \lambda_2 = 0.1$
F	0.06335103	0.06954624
FP	0.4269327	0.002245857
FN	0.3139659	0.9723261

Table 2: Nearest Neighbor Network for Fused Lasso Penalty

Additional Notes: Our FGL and GGL proposals have a number of advantages over these existing approaches. Methods for estimating time-varying networks cannot be easily extended to the setting where the classes lack a natural ordering. Guo et al. (2011)s proposal is a closer precursor to our method, and can in fact be stated as an instance of the problem (2.4) with a hierarchical group lasso penalty

$$\lambda \sum_{i \neq j} \sqrt{\sum_k |\theta_{ij}^{(k)}|}$$

that encourages a shared pattern of sparsity across the K classes. But the approach of Guo et al. (2011) has a number of disadvantages relative to FGL and GGL. (1) The penalty (5.20) is not convex, so convergence to the wrong local maximum is possible. (2) Because (5.20) is not convex, it is not possible to achieve the speed improvements described in Section 4. Consequently, the Guo et al. (2011) proposal is quite slow relative to our approach, as seen in Figures 2(e), 4(e), and 5(e), and essentially cannot be applied to very high-dimensional data sets. (3) Unlike FGL and GGL, it uses just one tuning parameter, and is unable to control separately the sparsity level and the extent of network similarity. (4) In cases where we expect edge values as well as network structure to be similar between classes, FGL is much better suited than GGL and Guo et al. (2011)s proposal, both of which encourage shared patterns of sparsity but ignore the signs and values of the nonzero edges.