Graphic Lasso: FNR/FPR issue in Simulation

Jiaxin Hu

December 20, 2020

- Compare the FNR/FPR under different tuning parameter selection criteria.
- Compare the FNR/FPR under different fitting functions.

1 Under different tuning parameter selection criteria.

Here we consider two criteria to select a proper tuning parameter in graphical lasso.

- **BIC:** With given tuning parameter sequence $\{\rho_i\}_{i=1}^n$, we select the parameter ρ_k such that

  $$k = \cancel{\rho_k} = \arg\min_{i\in[n]} 2n\left(\operatorname{tr}(S\hat{\Theta}_i) - \log\det(\hat{\Theta}_i)\right) + \log(n)m_i,$$

  where n is the sample size, $\hat{\Theta}_i$ is the estimated inverse covariance matrix with parameter ρ_i, and m_i is the number of non-zero off-diagonal elements in $\hat{\Theta}_i$.

- **maxPN:** With given tuning parameter sequence $\{\rho_i\}_{i=1}^n$, we select the parameter ρ_k such that

  $$\rho_k = \arg\min_{i\in[n]} \max\left\{FN(\hat{\Theta}_i,\Theta), FP(\hat{\Theta}_i,\Theta)\right\},$$

  where Θ is the true inverse covariance matrix, $FN(\cdot,\Theta)$ is the FNR between the estimate $\hat{\Theta}_i$ with parameter ρ_i and true matrix, and $FP(\cdot,\Theta)$ is the FPR of the estimate and true matrix.

  Note that maxPN criterion is not able to be used in practice because we do not have the true matrix Θ in the real-world application. The tuning parameter selected by maxPn can be considered a "reasonable" parameter, whose FNR and FPR would not be too large. In this sense, I believe the comparison between BIC and maxPN helps us to check whether BIC is a good criterion.

See Figures 1 and 2 for the simulation results.

With "my" R code, BIC has a similar performance with maxPN criterion. With `glasso` package, BIC outperforms than maxPN in terms of FNR while maxPN outperforms BIC in terms of FPR, under all settings except the dense setting.

2 Under different fitting functions.

At this point, we have three versions of the graphical lasso model: my R code (translated from the Matlab code but with a different regular lasso algorithm), R package `glasso`, and the Matlab code.

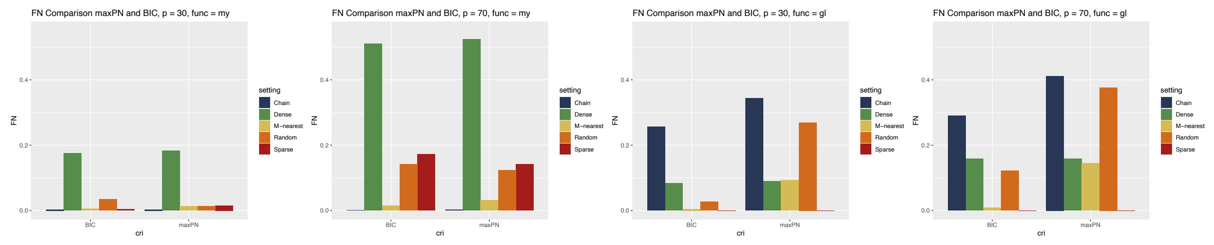

Figure 1: FNR for five different settings under different tuning parameter selection criteria. In all subfigures, the left cluster shows the results from BIC, and the right cluster from maxPN. The left two subfigures use "my" R code while the right two subfigures use the function from the package `glasso`. The first and third subfigures show the results with $p = 30$ while the second and fourth subfigures show the results with $p = 70$. All the values are the average results from 9 duplications.

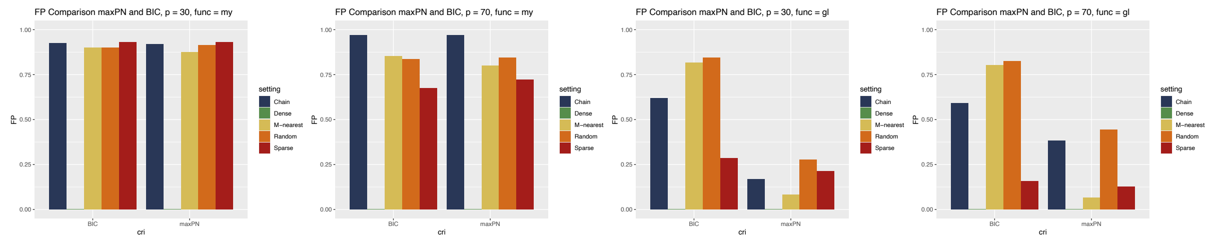

Figure 2: FPR for five different settings under different tuning parameter selection criteria. The experiment settings are the same as FNR experiment. All the values are the average results from 9 duplications.

2.1 Extension from criterion comparison

Note that I use my R code, denoted by "my", and the R package, denoted by "glasso", to check the goodness of BIC criterion and the difference between these two versions at the same time. Then, Figures 1 and 2 also imply some differences between "my" version and "glasso".

No matter which criterion we use, there exist big differences between the results from the two version, and the differences vary under different settings. Particularly, "glasso" has better FNR under the sparse and dense settings while "my" code has better FNR under chain, mnearest, random settins. However, in terms of FPR, "glasso" has much better performance than "my" code under all settings.

Figure 3 plots the true inverse matrix and estimated inverse matrices from "my" code and "glasso" with two criteria and $p = 30$. We choose the sparse setting where $(\Sigma^{-1})_{ii} = 1$ and $(\Sigma^{-1})_{i,i-1} = (\Sigma^{-1})_{i-1,i} = 0.5$. According to the plot, the "glasso" seems give a more significant pattern than "my" code.

2.2 Comparison with fixed tuning parameter

Since the results in Figures 1, 2, and 3 rely on the selection criteria, I consider the comparison for the three versions under a fixed tuning parameter.

I choose the sparse setting with $p = 30$ and $\rho = 0.3$, which usually give a good results under sparse setting. Figure 4 plots the true inverse matrix and the estimated inverse matrix from "my" code, "glasso", and "Matlab" code. We can see that both "glasso" and "Matlab" seem like give a more significant pattern than "my" code.

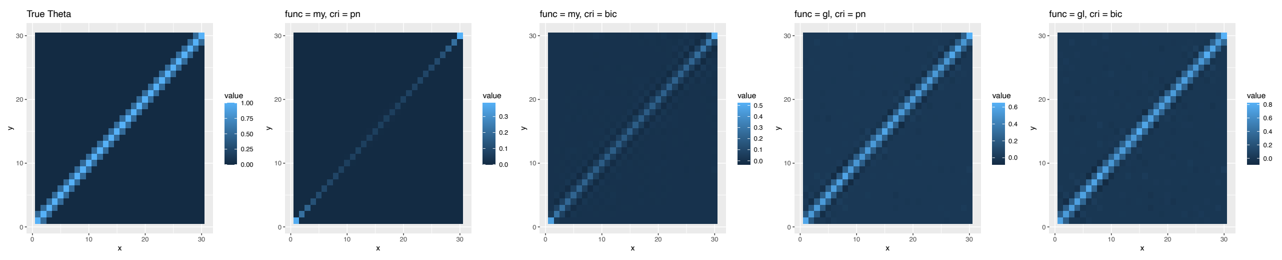

Figure 3: True inverse matrix and the estimated inverse matrices from: “my” code with BIC, “my” code with maxPN, “glasso” with BIC, and “glasso” with maxPN.

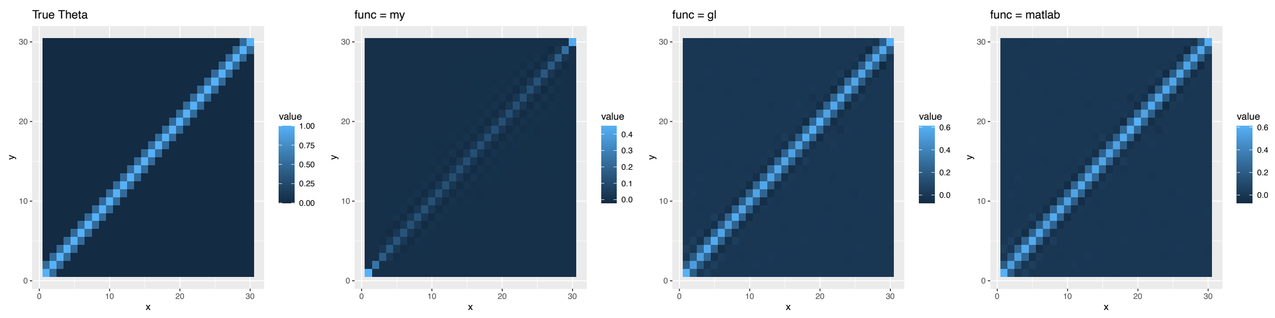

Figure 4: True inverse matrix and the estimated inverse matrices from: “my” code, “glasso” with BIC, and “Matlab” code.

shouldn’t we expect similar performance btw my code vs. matlab?

I repeat this experiment for five times. Figure 5 shows the FNR and FPR results for these three versions. The figure implies that “my” code has the worst performance in FNR while “matlab” has the worst performance in FPR. It seems like “glasso” is the best one among these three.

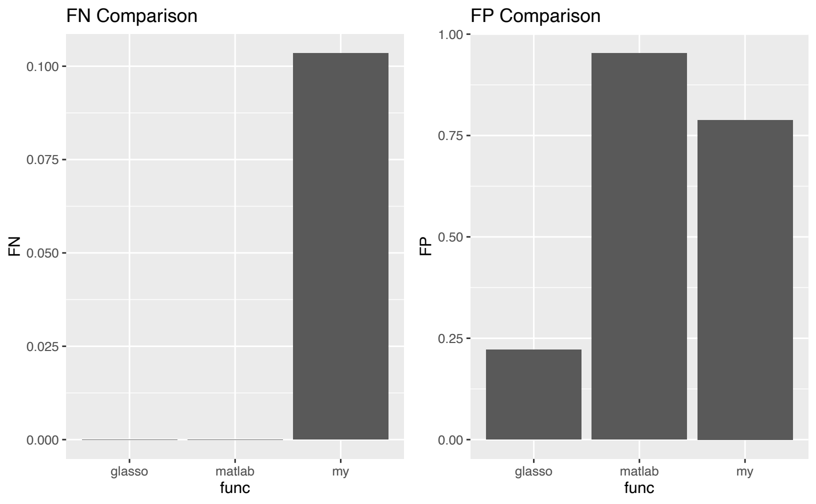

Figure 5: FNR (left) and FNR (right) results for “glassp”,“matlab” and “my” code under sparse setting with $p = 30$ and $\rho = 0.3$.

3 Others

- In terms of the running time of these three versions, the "glasso" package is much much fasters than the other two versions, and "my" code is faster than "matlab" version.

- In the comparison of selection criteria, the choice of $\{\rho_i\}_{i=1}^n$ is very tricky. Let $t = \max |S|_{ij}$. I use the sequence $(t/100, t/75, t/20, t/25, t/10, t/2)$. However, in some settings, this sequence is not good enough. For example, in a mnearest case, $\rho = 0.15$ and $\rho = 0.18$ lead to totally different results in FNR, from 1 to 0, and also in FPR, from 0 to 0.9.

  Therefore, it is very hard to choose a good tuning parameter and even evaluate whether the selection criterion is good enough. Further, it becomes very hard to evaluate the fitting functions with fixed ρ because it is hard to choose a proper ρ for comparison.

  That's why most paper presents ROC (with multiple rho's) instead of a single rho.

  Based on your simulation, can we conclude R glass is more accurate than matlab?

  Back to our model (see the screenshot I sent you earlier).
  Line 133 of simulation_preliminary.m shows bad estimation based on Glasso (matlab version).
  What is the reason? Is it because computational implementation (matlab vs. R) or something else?

  Try plotting similar figures as your Figure 3/4 based on line 133.