Shrinking Characteristics of Precision Matrix Estimators: An Illustration via Regression

Matt Galloway

2019-03-09

Contents

Chapter 1

Introduction

In many statistical applications, estimating the covariance matrix, Σ, for a set of random variables is a critical task. The covariance matrix is useful because it characterizes the *relationship* between variables. For instance, suppose we have three variables $X, Y,$ and Z and their covariance matrix is of the form

$$\Sigma_{xyz} = \begin{pmatrix} 1 & 0 & 0.5 \\ 0 & 1 & 0 \\ 0.5 & 0 & 1 \end{pmatrix}$$

We can gather valuable information from this matrix. First of all, we know that each of the variables has an equal variance of 1. Second, we know that variables X and Y are likely independent because the covariance between the two is equal to 0. This implies that any information in X is useless in trying to gather information about Y. Lastly, we know that variables X and Z are moderately, positively correlated because their covariance is 0.5.

Unfortunately, estimating Σ well is often computationally expensive and, in a few settings, extremely challenging. For this reason, emphasis in the literature and elsewhere has been placed on estimating the inverse of Σ which we like to denote as $\Omega \equiv \Sigma^{-1}$.

1.0.1 references

The reference we need here is **?**!

1. From **?**: "To fit many predictive models, only a characteristic of the population precision matrix needs to be estimated. For example, in binary linear discriminant analysis, the population precision matrix is needed for prediction only through the product of the precision matrix and the difference between the two conditional distribution mean vectors. Many authors have proposed methods that directly estimate this characteristic **?**; **?**; **?**."
2. From **?**: "**?** proposed a method for estimating the characteristic $\Omega_* \mu_*$ directly, but like the direct linear discriminant methods, this apporach does not lead to an estimate of Ω_*."
3. Discuss linear regression application.... where $\beta = \Omega_* \Sigma_{*XY}$. Like **?**, this approach estimates the regression coefficients "using shrinkage estimators of the marginal precision matrix for the predictors".

- Guide
- Model and notation

1.1 Outline

- Why do we care about covariance matrices? Covariance matrices are an object that summarizes the relationship between variables.
- What is a precision matrix? Precision matrix is the inverse of a covariance matrix.
- Precision matrices are powerful, essential in representing relationships in variables
- In fact, required to be estimated to fit many statistical models.
- That is, we must provide an object that accurately summarizes the relationship and interaction between variables in our model.
- Many proposed shrinkage estimators when p is particularly large. Shrinkage.

- However, to fit these models, Molstad and Rothman focused on the fact that onnly a "characteristic" of a precision matrix is required to fit these models. "We propose a framework to shrink a user-specified characteristic of a precision matrix estimator."
- For instance, LDA, linear regression, ...
- Their approach/methodology was published in BLAH. Shrinking characteristic of precision matrix estimators.
- BREAK
- We wanted to explore the regression application that was mentioned but not pursued/investigated
- This document will outline the research and work leading up to that
- The document is outlined as follows...
- We will begin chapter 2 with a brief introduction to precision matrix estimation. This will include mentions of estimation methods/approaches and we will spend a particular amount of time elaborating on one approach which is estimation via the ADMM algorithm. This dicussion will be useful when we circle back to SCPME, the regression application, and the augmented ADMM algorithm developed in BLAH.
- Lastly, the document will end with two brief tutorials for the R packages ADMMsigma and SCPME. These packages, developed by myself, will be mentioned throughout the document and were instrumental in this pursued line of research and facilitated many simulations and research direction explored throughout. Both packages have since been published on CRAN

1.1.1 Notation and Definitions

For strictly positive integers n and p, we will denote $\mathbb{R}^{n \times p}$ as the class of real matrices with dimenson $n \times p$. The class of real, symmetric matrices with dimension $p \times p$ will be denoted as $\mathbb{S}^p$ and $\mathbb{S}^p_+$ if we further require the object to be positive definite. The sample size in a given data set will most often correspond with n and p will correspond with the dimension of the prediction vector. If the dimension of the response vector exceeds one, we will denote it as r.

Most matrices will take the form of either Σ, which should be taken to represent the population covariance matrix, or Ω, which should be taken to represent the population *precision* matrix. Note that the precision matrix is simply the inverse of the covariance matrix ($\Omega \equiv \Sigma^{-1}$). We will sometimes add a subscript star if the object is oracle - or known- a priori (Ω_*) and the population object's estimator that minimizes/maximizes a pre-specified objective function will have a hat attached to it ($\hat{\Omega}$).

There will be significant matrix algebra notation throughout the document. The trace operator which sums the diagonal elements of a matrix will take the form $tr\left(\cdot\right)$ and the *exponential* trace operator will be denoted similarly as $etr\left(\cdot\right)$. The determinant of a matrix $\mathbf{A}$ will be denoted as $|\mathbf{A}|$ but may also take the form as simply $det\left(\mathbf{A}\right)$. The kronecker product of two matrices $\mathbf{A}$ and $\mathbf{B}$ will be denoted as $\mathbf{A} \otimes \mathbf{B}$. Lastly, the Frobenius norm which sums the square of all entries in a matrix will be denoted as $\left\|\mathbf{A}\right\|_F$ and we will define $\left\|\mathbf{A}\right\|_1 := \sum_{i,j}\left|\mathbf{A}_{ij}\right|$ where the i-jth element in matrix $\mathbf{A}$ is denoted as $\left(\mathbf{A}\right)_{ij}$ or simply $\mathbf{A}_{ij}$.

Chapter 2

Precision Matrix Estimation

2.0.1 Literature Review?

- Articles that review covariance and precision matrix estimation in high dimensional settings are **?** and **?**.
- The person who initially proposed the lasso-penalized gaussian likelihood precision matrix estimation is **?**.
- Other authors who have proposed methods with non-lasso penalizations are (Frobenius norms) **?**, **?**, (non-convex penalties) **?**, and **?**.
- Standard alternating direction method of multipliers approaches are found in **?**.
- ADMM with modification based on majorize-minimize principle is found in **?**.
- Quadratic equation with ridge is solve in closed form using a similar approach to **?**; **?**.
- L1-penalized gaussian likelihood precision matrix estimator (**?**; **?**; **?**).

2.1 Background

Consider the case where we observe n independent, identically distributed copies of the random variable (X_i) where $X_i \in \mathbb{R}^p$ is normally distributed with some mean, μ, and some variance, Ω^{-1}. That is, $X_i \sim N_p\left(\mu, \Omega^{-1}\right)$.

Because we assume independence, we know that the probability of observing these specific observations $X_1, ..., X_n$ is equal to

$$\begin{aligned} f(X_1, ..., X_n; \mu, \Omega^{-1}) &= \prod_{i=1}^{n} (2\pi)^{-p/2} \left|\Omega\right|^{1/2} \exp\left[-\frac{1}{2}\left(X_i - \mu\right)' \Omega \left(X_i - \mu\right)\right] \\ &= (2\pi)^{-np/2} \left|\Omega\right|^{n/2} \text{etr}\left[-\frac{1}{2}\sum_{i=1}^{n}\left(X_i - \mu\right)\left(X_i - \mu\right)' \Omega\right] \end{aligned}$$

where $\text{etr}\left(\cdot\right)$ denotes the exponential trace operator. It follows that the log-likelihood for μ and Ω is equal to the following:

$$l(\mu, \Omega | X) = const. + \frac{n}{2}\log\left|\Omega\right| - tr\left[\frac{1}{2}\sum_{i=1}^{n}\left(X_i - \mu\right)\left(X_i - \mu\right)' \Omega\right]$$

If we are interested in estimating μ, it is relatively straight forward to show that the maximum likelihood estimator (MLE) for μ is $\hat{\mu}^{MLE} = \sum_{i=1}^{n} X_i/n$ which we typically denote as $\bar{X}$. However, in addition to μ, many applications require the estimation of Σ as well. We can also find a maximum likelihood estimator:

$$\begin{aligned} \hat{\Omega}^{mle} &= \arg\max_{\Omega \in S_{+}^{p}}\left\{const. + \frac{n}{2}\log\left|\Omega\right| - tr\left[\frac{1}{2}\sum_{i=1}^{n}\left(X_i - \mu\right)\left(X_i - \mu\right)' \Omega\right]\right\} \\ &= \left[\frac{1}{n}\sum_{i=1}^{n}\left(X_i - \bar{X}\right)\left(X_i - \bar{X}\right)'\right]^{-1} \end{aligned}$$

By setting the gradient equal to zero and plugging in the MLE for μ, we find that the MLE for Ω is the inverse our usual sample estimator for the covariance matrix, denoted here as S^{-1}. It turns out that we could have just as easily computed the maximum likelihood estimator for the precision matrix $\Omega \equiv \Sigma^{-1}$ and taken its inverse:

$$\hat{\Omega}^{mle} = \arg\min_{\Omega \in S_{+}^{p}} \left\{ tr\left(S\Omega\right) - \log\left|\Omega\right| \right\} \tag{2.1}$$

so that $\hat{\Omega}^{MLE} = S^{-1}$. Beyond the formatting convenience, computing estimates for Ω as opposed to Σ often poses less computational challenges – and accordingly, the literature has placed more emphasis on efficiently solving for Ω instead of Σ[1].

As in regression settings, we can construct a *penalized* log-likelihood estimator by adding a penalty term, $P\left(\Omega\right)$, to the log-likelihood so that

$$\hat{\Omega} = \arg\min_{\Omega \in S_{+}^{p}} \left\{ tr\left(S\Omega\right) - \log\left|\Omega\right| + P\left(\Omega\right) \right\} \tag{2.2}$$

$P\left(\Omega\right)$ is often of the form $P\left(\Omega\right) = \lambda\|\Omega\|_F^2/2$ or $P\left(\Omega\right) = \|\Omega\|_1$ where $\lambda > 0$, $\left\|\cdot\right\|_F^2$ is the Frobenius norm and we define $\left\|A\right\|_1 = \sum_{i,j}\left|A_{ij}\right|$. These penalties are the ridge and lasso, respectively.

2.2 ADMM algorithm

Many efficient methods have been proposed to solve for $\hat{\Omega}$ when $\alpha = 1$. The most popular method is the graphical lasso algorithm (glasso) introduced by **?**. However, no methods (to the best of my knowledge) have estimated Ω when $\alpha \in (0, 1)$. We will use the alternating direction method of multipliers (ADMM) algorithm to do so.

As the authors state in **?**, the "ADMM is an algorithm that is intended to blend the decomposability of dual ascent with the superior convergence properties of the method of multipliers." For our purposes, we will only focus on the ADMM algorithm but it is encouraged to read the original text from Boyd and others for a complete introduction to the other two methods.

In general, suppose we want to solve an optimization problem of the following form:

[1] Notice that here we are *minimizing* the negative log-likelihood as opposed to maximizing the log-likelihood. Both procedures will result in the same estimate.

$$\begin{aligned} &\text{minimize } f(x) + g(z) \\ &\text{subject to } Ax + Bz = c \end{aligned}$$

where $x \in \mathbb{R}^n, z \in \mathbb{R}^m, A \in \mathbb{R}^{p \times n}, B \in \mathbb{R}^{p \times m}$, $c \in \mathbb{R}^p$, and f and g are assumed to be convex functions (following **?**, the estimation procedure will be introduced in vector form though we could just as easily take x and z to be matrices). In addition to penalized precision matrix estimation, optimization problems like this arise naturally in several statistics and machine learning applications – particularly regularization methods. For instance, we could take f to be the squared error loss, g to be the l_2-norm, c to be equal to zero and A and B to be identity matrices to solve the ridge regression optimization problem. In all cases, our goal is to find $x^* \in \mathbb{R}^n$ and $z^* \in \mathbb{R}^m$ that achieves the infimum p^*:

$$p^* = inf\left\{f(x) + g(z) | Ax + Bz = c\right\}$$

To do so, the ADMM algorithm uses the *augmented lagrangian*

$$L_\rho(x, z, y) = f(x) + g(z) + y'(Ax + Bz - c) + \frac{\rho}{2}\left\|Ax + Bz - c\right\|_2^2$$

where $y \in \mathbb{R}^p$ is the lagrange multiplier and $\rho > 0$ is a scalar. Clearly any minimizer, p^*, under the augmented lagrangian is equivalent to that of the lagrangian since any feasible point (x, z) satisfies the constraint $\rho\left\|Ax + Bz - c\right\|_2^2/2 = 0$.

The algorithm consists of the following repeated iterations:

$$\begin{aligned} x^{k+1} &= \arg\min_{x \in \mathbb{R}^n} L_\rho(x, z^k, y^k) \\ z^{k+1} &= \arg\min_{z \in \mathbb{R}^m} L_\rho(x^{k+1}, z, y^k) \\ y^{k+1} &= y^k + \rho(Ax^{k+1} + Bz^{k+1} - c) \end{aligned}$$

In the context of precision matrix estimation, we can let f be equal to the non-penalized likelihood, g be equal to $P\left(\Omega\right)$, and use the constraint Ω equal to some Z so that the lagrangian is

$$L_{\rho}(\Omega, Z, \Lambda) = f\left(\Omega\right) + g\left(Z\right) + tr\left[\Lambda\left(\Omega - Z\right)\right]$$

and the augmented lagrangian is

$$L_{\rho}(\Omega, Z, \Lambda) = f\left(\Omega\right) + g\left(Z\right) + tr\left[\Lambda\left(\Omega - Z\right)\right] + \frac{\rho}{2}\left\|\Omega - Z\right\|_{F}^{2}$$

In the `ADMMsigma` package, we instead take

$$P\left(\Omega\right) = \lambda\left[\frac{1-\alpha}{2}\left\|\Omega\right|_{F}^{2} + \alpha\left\|\Omega\right\|_{1}\right]$$

so that solving the full penalized log-likelihood for Ω results in solving

$$\hat{\Omega} = \arg\min_{\Omega \in S_{+}^{p}}\left\{tr\left(S\Omega\right) - \log\left|\Omega\right| + \lambda\left[\frac{1-\alpha}{2}\left\|\Omega\right|_{F}^{2} + \alpha\left\|\Omega\right\|_{1}\right]\right\} \tag{2.3}$$

where $0 \leq \alpha \leq 1$. This penalty, know as the *elastic-net* penalty, was explored by Hui Zou and Trevor Hastie (**?**) and is identical to the penalty used in the popular penalized regression package `glmnet`. Clearly, when $\alpha = 0$ the elastic-net reduces to a ridge-type penalty and when $\alpha = 1$ it reduces to a lasso-type penalty. Having this flexibility and generalization allows us to perform cross validation across proposed α values in addition to proposed λ values.

The ADMM algorithm is now the following:

$$\Omega^{k+1} = \arg\min_{\Omega \in \mathbb{S}_{+}^{p}} \left\{ tr\left(S\Omega\right) - \log\left|\Omega\right| + tr\left[\Lambda^{k}\left(\Omega - Z^{k}\right)\right] + \frac{\rho}{2}\left\|\Omega - Z^{k}\right\|_{F}^{2} \right\} \tag{2.4}$$

$$Z^{k+1} = \arg\min_{Z \in \mathbb{S}^{p}} \left\{ \lambda\left[\frac{1-\alpha}{2}\left\|Z\right\|_{F}^{2} + \alpha\left\|Z\right\|_{1}\right] + tr\left[\Lambda^{k}\left(\Omega^{k+1} - Z\right)\right] + \frac{\rho}{2}\left\|\Omega^{k+1} - Z\right\|_{F}^{2} \right\} \tag{2.5}$$

$$\Lambda^{k+1} = \Lambda^{k} + \rho\left(\Omega^{k+1} - Z^{k+1}\right) \tag{2.6}$$

2.2.1 Algorithm

Set $k = 0$ and initialize Z^{0}, Λ^{0}, and ρ. Repeat steps 1-3 until convergence:

1. Decompose $S + \Lambda^{k} - \rho Z^{k} = VQV'$ via spectral decomposition[2]. Then

$$\Omega^{k+1} = \frac{1}{2\rho}V\left[-Q + \left(Q^{2} + 4\rho I_{p}\right)^{1/2}\right]V' \tag{2.7}$$

2. Elementwise soft-thresholding for all $i = 1, ..., p$ and $j = 1, ..., p$[3].

$$\begin{aligned} Z_{ij}^{k+1} &= \frac{1}{\lambda(1-\alpha) + \rho}\text{sign}\left(\rho\Omega_{ij}^{k+1} + \Lambda_{ij}^{k}\right)\left(\left|\rho\Omega_{ij}^{k+1} + \Lambda_{ij}^{k}\right| - \lambda\alpha\right)_{+} \\ &= \frac{1}{\lambda(1-\alpha) + \rho}\text{soft}\left(\rho\Omega_{ij}^{k+1} + \Lambda_{ij}^{k}, \lambda\alpha\right) \end{aligned} \tag{2.8}$$

3. Update Λ^{k+1}.

$$\Lambda^{k+1} = \Lambda^{k} + \rho\left(\Omega^{k+1} - Z^{k+1}\right)$$

where $\text{soft}(a, b) = \text{sign}(a)(|a| - b)_{+}$.

[2]Proof of (**??**) in section **??**.
[3]Proof of (**??**) in section **??**.