Uniwersytet Wrocławski
Wydział Matematyki i Informatyki
Instytut Matematyczny
specjalność: Zastosowania rachunku prawdopodobieństwa i statystyki
nurt statystyczny

Michał Makowski

Precision matrix estimation in Gaussian graphical models

Praca magisterska
napisana pod kierunkiem
dr hab. Małgorzaty Bogdan

Wrocław 2018

Abstract

Relationships between variables in large data sets could be represented as graphs, where nodes represent variables, and edges connect variables which are conditionally dependent, given all other variables. In such graphs number of edges going out from a given node can be used as the measure of the importance of a given variable. Very often quantitative variables can be transformed so that their distribution resembles the normal distribution, and their joint distribution can be modelled using the multivariate normal distribution. In related Gaussian graphical models edges correspond to nonzero elements of a precision matrix, which is an inverse of a covariance matrix. In a case when the number of observations in a database is comparable to or smaller than the number of variables, classical maximum likelihood estimates of the precision matrix do not exist or have the very large variance. We cover the problem of a sparse precision matrix estimation. We analyze and compare two different regularization methods, gLasso and gSLOPE, which stabilize the performance of MLE. The first is the well-known method for this kind of problems, specifically, gLASSO uses ℓ_1 norm penalty to shrink the estimates to zero, the second is a novel method, which uses *Sorted* ℓ_1 *Penalized Regression* (SLOPE) to estimate matrix coefficients. SLOPE was introduced by Bogdan et al. in 2013 as a new estimator for a vector of coefficients in a linear model, gSLOPE was introduced by Piotr Sobczyk. We developed ADMM algorithm for these two approaches. ADMM for gSLOPE is our main result, as gSLOPE was not implemented efficiently earlier. We compare the performance of the obtained solutions using synthetic experiments. The simulations showed that gSLOPE systematically outperforms gLASSO with respect to ROC curves, which illustrates the compromise between the specificity and sensitivity in discovering the graph structure.

Contents

0.1 Introduction

This thesis was written as a final paper for my studies at the Faculty of Mathematics, University of Wrocław. With Małgorzata Bogdan, PhD, we have faced the problem of a precision matrix estimation and its application in a probabilistic graphical models theory.

Graphical models help to model dependencies between random variables. If a multivariate random variable can be presented in the form of the exponential family, then the graph model a property of conditional dependence; edges connect variables which are conditionally dependent, given all other variables.

Precision matrix is closely connected to graphical models which are a powerful tool to model relationships among a large number of random variables in a complex system and are used in a wide array of scientific applications. The precision matrix also plays a significant role in many high-dimensional inference problems. For example, knowledge of the precision matrix is crucial for classification and discriminant analyses, it is also useful for a broad range of applications such as portfolio optimization, speech recognition, and genomics. Many examples are given by [KF09].

The problem of precision matrix estimation in high dimensional Gaussian graphical models is the main point covered in this thesis. We present the implementation and results of graphical Lasso and graphical SLOPE. Lasso is the well-known variable selection and regularization method, SLOPE (*Sorted* l_1 *Penalized Regression*) is variable selection method introduced by Bogdan et. al in 2015, which in some settings outperforms Lasso [Bog+15].

Graphical SLOPE algorithm has never been implemented efficiently earlier. Its derivation is the main result of this thesis. To implement graphical SLOPE we used ADMM (*Alternative Direction Method of Multipliers*) algorithm, which was developed in 1970's. It is gaining popularity recently, many examples of usage are given in [Boy+10]. The implementation gave us an opportunity for a comprehensive investigation of gSLOPE properties.

At the beginning of the thesis we introduce a basic background of a convex optimization theory, then we proceed to ADMM algorithm explanation. In the next chapter we introduce graphical models. We provide basics of graph theory and then we proceed to the concept of probabilistic Gaussian graphical models. Later we describe estimation methods, graphical Lasso and graphical SLOPE. After that, we derive ADMM algorithm for both methods. Finally, we present the main results of our work. As we implemented graphical estimators using the R programming language, we performed a number of synthetic experiments and analyze results. At the end, there is a short conclusion and reference about possible ways of development presented methods.

We assume that the reader is familiar with probability and statistics at a graduate level, but do not have knowledge about graphical models and convex optimization. Thus we provide a short introduction into both fields, a more experienced reader can skip the first part.

All scripts used in this thesis are stored on my GitHub account: `https://github.com/mmaku`. An R package is planned to be published in the future.

Current work on gSLOPE is focused on

0.2 Notation

Although many concepts presented below are defined later, we decided to include most important of them at the beginning to make it easier for the reader. The first six rules are not strict, but usually we follow this notation which helps to keep the paper clean

- lower-case letter x denotes a scalar, vector or realization of random variable,
- straight lower-case letter x denotes a realization of random vector,
- upper-case letter X denotes a random variable or matrix,
- bold upper-case letter $\mathbf{X}$ a sample of random vectors,
- double stroke upper-case $\mathbb{X}$ letter usually denotes a random vector,
- $\mathbf{int}$ denotes the interior of a set,
- $\mathbf{bd}$ denotes the boundary of a set
- $\mathbf{cl}$ denotes the closure of a set,
- $\mathbf{aff}(\ldots), \mathbf{conv}(\ldots)$ denotes the affine and convex hull,
- S^n, S^n_+, S^n_{++} denotes the set of the symmetric, symmetric positive semidefinite and symmetric positive definite matrices, respectively,
- $\mathbf{dom}(\ldots)$ denotes a domain of a function,
- $L(\ldots)$ denotes the Lagrangian dual function,
- $\mathcal{L}_\rho(\ldots)$ denotes the augmented Lagrangian with penalty parameter ρ,
- $\mathbb{I}[\ldots]$ denotes the indicator function,
- $\mathbf{sign}(\ldots)$ denotes the signum function,
- $\mathbf{prox}_{f,\rho}(\ldots)$ denotes the proximity operator of a function f with a penalty ρ,
- $\mathrm{tr}(\ldots)$ denotes the trace of a matrix,
- $\mathrm{diag}\ldots$ denotes the diagonal of a matrix,
- $\mathrm{rank}(\ldots)$ the rank of a matrix,
- Symbol $\equiv$ denotes equivalence relation,
- $[A]^k$ denotes the set of all k-element subsets of A,
- $\mathbb{P}(\ldots)$ denotes the probability measure (might be used as a pdf),
- $\perp\!\!\!\perp$ denotes the condition independence relation,
- $\mathbb{L}(\ldots)$ denotes the (log-)likelihood function,
- $\mathbb{E}[\ldots]$ denotes the expected value,
- J_λ denotes the series of regularizer for graphical SLOPE,
- $\mathbf{S}$ denotes the sample covariance matrix,
- $\mathbf{1}$ denotes the identity matrix,
- $\mathbf{\Theta}$ denotes the precision matrix,
- $\mathrm{qt}_r(\ldots)$ denotes the quantile function of the Student's t-distribution with r degrees of freedom.

Chapter 1

Convex optimization

Convex optimization is a special class of optimization problems, where we try to find a solution of a so-called convex problem. In this chapter, we introduce convex optimization concepts. A reader with some theoretical background regarding convex optimization could jump to the last section of this chapter.

In the first section we introduce concepts of convex optimization, its basics definitions and theorems, then we present a concept of duality. Later we describe Augmented Direction Method of Multipliers, the appropriate method for our need. As mathematical optimization is the extremely broad field, we tried to compress it as much as it is possible. The most important concepts needed to understand ADMM algorithm are presented, but for the better understanding of theory, we suggest the reader study one of the cited books.

The material in this chapter is based on [BV09; Par14; Boy+10]. Some ideas come from the class *Math 301: Advanced Topics in Convex Optimization* of prof. Candes at Stanford University [Can15].

1.1 Background

1.1.1 Sets convexity

Affine and convex sets

We start from a very basic definition.

Definition 1.1.1 (Convex set). Let $C \subseteq \mathbb{R}^n$, the set C is *convex* if for any two points $x_1, x_2 \in C$ and scalar $\theta \in [0,1]$, we have $\theta x_1 + (1-\theta)x_2 \in C$.

Roughly speaking, a set is convex if every point in the set can be seen along an unobstructed straight path between it and any other point in the set, where unobstructed means lying in the set.

This idea can be generalized to more than two points.

Definition 1.1.2 (Convex combination). Let $x_1, \ldots, x_k \in \mathbb{R}$, and $\theta_1, \ldots, \theta_k \in \mathbb{R}$ that satisfy $\forall_i \ \theta_i \geqslant 0$ and $\theta_1 + \cdots + \theta_k = 1$. A point of the form $\theta_1 x_1 + \cdots + \theta_k x_k$, is called a *convex combination*.

It can be shown that a set is convex, if and only if it contains every convex combination of its points, see [BV09], section 2.1.4. A convex combination of points can be seen of as its weighted average.

Definition 1.1.3 (Convex hull). Let $C \subseteq \mathbb{R}^n$. A set of all convex combinations of points from the set C is called a *convex hull* of C, and is denoted by $\mathbf{conv}(C)$, formally

$$\mathbf{conv}(C) = \{\theta_1 x_1 + \ldots + \theta_k x_l \colon \forall i \quad x_i \geqslant 0 x_1, \ldots, x_k \in C; \ \ \theta_1 + \ldots + \theta_k = 1\}.$$

A convex hull of C is the smallest convex set that contains C.

If in above definitions we take $\theta \in \mathbb{R}$ instead of $\theta \in [0,1]$ we get the definitions of an *Affine set*, *Affine combination* and *Affine hull*, respectively. Of course every affine set is also convex.

Cones

Definition 1.1.4 (Cone, convex cone). Let $C \subseteq \mathbb{R}^n$. The set C is called a *cone*, or *nonnegative homogeneous*, if for every $x \in C$ and $\theta \geqslant 0$, we have $\theta x \in C$. The set C is a *convex cone* if it is convex and a cone, which means for any $x_1, x_2 \in C$ and $\theta_1, \theta_2 \geqslant 0$, we have

$$\theta_1 x_1 + \theta_2 x_2 \in C.$$

Definition 1.1.5 (Conic combination). Let $x_1, \ldots, x_k \in \mathbb{R}^n$, and $\theta_1, \ldots, \theta_k \in \mathbb{R}$ that satisfy $\forall_i \ \theta_i \geqslant 0$. A point of the form $\theta_1 x_1 + \cdots + \theta_k x_k$, is called a *conic combination* (or a *nonnegative linear combination*) of $x_1, \ldots, x_k$.

If points $x_1, \ldots, x_k$ are in a convex cone C, then every conic combination of $x_1, \ldots, x_k$ belong to C. Conversely, a set C is a convex cone if and only if it contains all conic combination of its elements. Proof of this statements can be found in [Roc], chapter 1.

Definition 1.1.6 (Conic hull). A *conic hull* of a set C is the set of all conic combinations of points in C, that is

$$\{\theta_1 x_1 + \ldots + \theta_k x_l \colon \forall i \quad x_i \in C; \ \ \theta_i \geqslant 0\}.$$

The conic hull of C is the smallest conic set that contains C.

Hyperplanes and halfspaces

Definition 1.1.7 (Hyperplane). Let $a \in \mathbb{R}^n$, $a \neq 0$ and $b \in \mathbb{R}$. A *hyperplane* H is a set of the form

$$H = \{x \colon \ a^T x = b\}.$$

Analytically it is the solution set of a nontrivial linear equation among the components of x, geometrically, it can be interpreted as the set of points with a constant inner product to a given vector a, or as a hyperplane with normal vector a; the constant $b \in \mathbb{R}$ determines the offset of the hyperplane from the origin.

A hyperplane divides $\mathbb{R}^n$ into two *halfspaces*. Let $a \neq 0$, a (closed) halfspace is a set of the form

$$\{x \colon \ a^T x \leqslant b\}.$$

Halfspaces are convex, but not affine.

Norm balls and norm cones

Suppose $\|\cdot\|$ is any norm on $\mathbb{R}^n$. From the general properties of norms it can be shown that a *norm ball* of radius r and center x_c, given by $\{x \colon \ \|x - x_c\| \leqslant r\}$, is convex.

Definition 1.1.8 (Norm cone). The *norm cone* associated with the norm $\|\cdot\|$ is the set

$$C = \{(x, t) \colon \ \|x\| \leqslant t\} \subseteq \mathbb{R}^{n+1}.$$

Of course, it is a convex cone.

Positive semidefinite cone

The curled inequality symbol $\succeq$ (and its strict form $\succ$) is used to denote generalized inequality: between vectors, it represents componentwise inequality; between symmetric matrices, it represents matrix inequality, that is for a matrix $A \in \mathbb{R}^{n \times n}$, $A \succeq 0$ means that A is the positive semidefinite.

By S_+^n we denote a set of symmetric positive semidefinite $n \times n$ matrices, that is

$$S_+^n = \{x \in S^{n \times n} \colon \ x \succeq 0\}.$$

Lemma 1.1.1. *The set S_+^n is a convex cone.*

Proof. Directly from the definition of positive semidefiniteness: for any $x \in \mathbb{R}^n$, $A \succeq 0, B \succeq 0$ and $\alpha_1, \alpha_2 \geqslant 0$ we have

$$x^T (\alpha_1 A + \alpha_2 B) x = x^T \alpha_1 A x + x^T \alpha_2 B x \geqslant 0.$$

■

The set S_+^n is called a *positive semidefinite cone*.
Analogously, by S_{++}^n we denote a set of symmetric positive definite $n \times n$ matrices, that is

$$S_{++}^n = \{x \in S^{n \times n} : x \succ 0\}.$$

The notation S_{++} and S_+ is used when dimensionality is not specified and could be omitted.

Proper cones

Definition 1.1.9 (Proper cone). A cone $K \subseteq \mathbb{R}^n$ is called a *proper cone* if it satisfies the following:

- K is convex,
- K is closed,
- K has nonempty interior,
- K is *pointed*, which means that it contains no line (i.e. $x \in K, -x \in K \Rightarrow x = 0$).

A proper cone can be used to define a *generalized inequality*, which is partial ordering on $\mathbb{R}^n$. We define the partial ordering on $\mathbb{R}^n$ associated with cone K by

$$x \preceq_K y \Rightarrow y - x \in K.$$

Strict partial ordering is defined by

$$x \prec_K y \Rightarrow y - x \in \mathbf{int}\, K.$$

If $K = \mathbb{R} \setminus \{0\}$ then $\succeq_k$ is just the usual ordering $\leqslant$ and $\succ_k$ is the strict partial ordering ("$<$").

Operations that preserve sets convexity

Below we just list some operations that preserve convexity, proofs of this facts can be found in [BV09], section 2.3.

- Intersection.
- An image of a convex set under an affine function set is convex.

Proper cones and generalized inequalities

Definition 1.1.10 (Proper cone). A cone $K \subset \mathbb{R}^n$ is called *proper cone* if it satisfies the following:

- K is convex,
- K is closed
- K has nonempty interior,
- K is *pointed*, which means that it contains no line (i.e. $x \in K, -x \in K \Rightarrow x = 0$)

A proper cone can be used to define a *generalized inequality* , which i a partial ordering on $\mathbb{R}^n$. We define partial ordering on $\mathbb{R}^n$ associated with cone K by

$$x \preceq y \Rightarrow y - x \in K.$$

Strict partial ordering is defined by

$$x \prec y \Rightarrow y - x \in \operatorname{int} K.$$

When K is a set of nonnegative real numbers, then $\succeq_K$ is just usual ordering $\leqslant$ and $\succ_K$ is strict partial ordering ("$<$"), so generalized inequalities include as a special case ordinary (nonstrict and strict) inequality in $\mathbb{R}$.

Generalized inequalities have a number of useful properties that can be found in a subsection 2.4.1 of [BV09].

Separating and supporting hyperplanes

In this subsection we briefly describe an idea that is important in convex optimization: the use of hyperplanes or affine functions to separate convex sets that do not intersect. The basic result is the separating hyperplane theorem.

Theorem 1.1.2 (Hyperplane separation theorem). *Suppose* C *and* D *are nonempty disjoint convex sets. Then there exist* a $a \neq 0$ *and* b *such that* $\forall_{x \in C}$ $a^T x \leqslant b$ *and* $\forall_{x \in D}$ $a^T x \geqslant b$. *In other words, the affine function* $a^T x - b$ *is nonpositive on* C *and nonnegative on* D. *The hyperplane* $\{x\colon a^T x = b\}$ *is called a separating hyperplane for the sets* C *and* D, *or is said to separate the sets* C *and* D

Proof of theorem 1.1.2 can be found in [BV09], subsection 2.5.1.

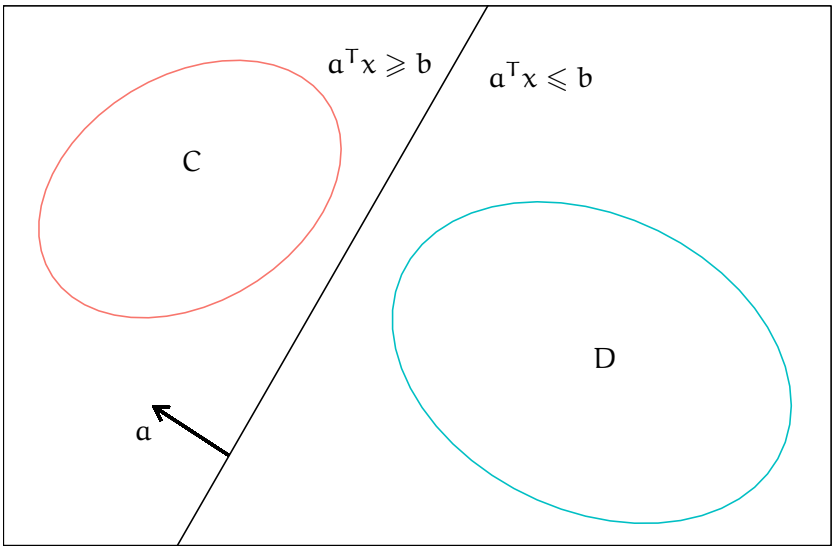


Figure 1.1 – The hyperplane $\{x\colon a^T x = b\}$ separates the disjoint convex sets C and D. The affine function $a^T x - b$ is nonpositive on the set C and nonnegative on the set D.

Definition 1.1.11 (Supporting hyperplane). Suppose $C \subseteq \mathbb{R}^n$, and x_0 is a point in its boundary $\mathbf{bd}\, C$, that is,

$$x_0 \in \mathbf{bd}\, C = \mathbf{cl}\, C \setminus \mathbf{int}\, C.$$

If $a \neq 0$ satisfies

$$\forall x \in C \quad a^T x \leqslant a^T x_0,$$

then the hyperplane $x\colon a^T x = a^T x_0$ is called a *supporting hyperplane* to C at the point x_0.

Similarly to hyperplane separation there, there exist supporting hyperplane theorem which states that for any nonempty convex set C, and any $x_0 \in \mathbf{bd}\, C$, there exists a supporting hyperplane to C at x_0.

Dual cones

Definition 1.1.12 (Dual cone). Let K be a cone. The set K^* defined as

$$K^* = \{y\colon \forall x \in K \quad x^T y \geqslant 0\}$$

is called a *dual cone of* K.

As the name suggests, K^* is a cone, and is always convex, even when the original cone K is not, [BV09], section 2.6. Geometrically, $y \in K^*$ if and only if $-y$ is the normal of a hyperplane that supports K at the origin.

Dual cone properties induce that if K is a proper cone, then so is its dual K^*, and moreover, that $K^{**} = K$, see section cited above.

If a cone K is the proper cone, it induces a generalized inequality $\succeq_K$. Then its dual cone K^* is also the proper cone, and therefore induces a generalized inequality. We refer to the generalized inequality $\succeq_{K^*}$ as the dual of the generalized inequality $\succeq_K$.

1.1.2 Functions convexity

Convex function

Recall the definition of a convex function.

Definition 1.1.13 (Convex function). A function $f\colon \mathbb{R}^n \to \mathbb{R}$ is *convex* if $\mathbf{dom}\,(f)$ is a convex set and if for all $x, y \in \mathbf{dom}\,(f)$, and $0 \leqslant \theta \leqslant 1$, we have

$$f(\theta x + (1-\theta)y) \leqslant \theta f(x) + (1-\theta)f(y) \tag{1.1}$$

Which means that the line segment between $(x, f(x))$ and $(y, f(y))$, which is the *chord* from x to y, lies above the graph of f. A function f is *strictly convex* if strict inequality holds in eq. (1.1) whenever $x \neq y$ and $0 < \theta < 1$. We say f is (strictly) concave if $-f$ is (strictly) convex.

There exist first and second-order conditions for function convexity, they can be found in every book about mathematical optimization.

Epigraph

Definition 1.1.14 (Function graph). The graph of a function $f\colon \mathbb{R}^n \to \mathbb{R}$ is defined as

$$\{(x, f(x))\colon x \in \mathbf{dom}\,(f)\},$$

which is a subset of R^{n+1}.

Definition 1.1.15. The epigraph of a function $f\colon \mathbb{R}^n \to \mathbb{R}$ is defined as

$$\mathbf{epi}\,(f) = \{(x, t)\colon x \in \mathbf{dom}\,(f)\,;\ \ f(x) \leqslant t\},$$

which is a subset of R^{n+1}.

'Epi' means 'above' so epigraph means 'above the graph'. The definition is illustrated in the fig. 1.2. The link between convex sets and convex functions is via the epigraph: a function is convex if and only if its epigraph is a convex set.

Operations that preserve sets convexity

Below we just list some operations that preserve convexity, proofs of this facts can be found in [BV09], section 3.2.

- Nonnegative weighted sum of convex functions is a convex function (i.e. the set of convex function is a convex cone).
- Pointwise maximum and supremum over a set of convex functions.

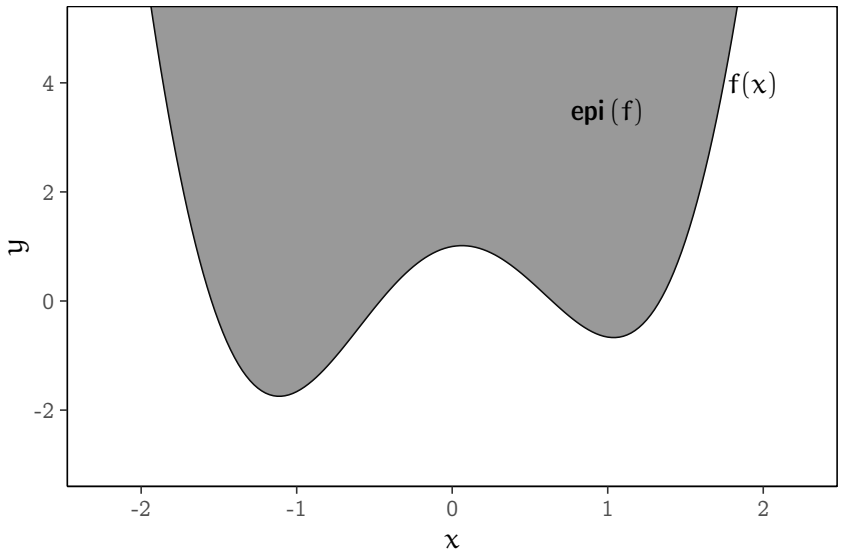


Figure 1.2 – Epigraph of a function f, shown shaded. The black lower boundary, is the graph of the function f.

The conjugate function

Now we introduce an operation that plays a significant role in the theory of convex analysis and optimization.

Definition 1.1.16. Let $f\colon\ \mathbb{R}^n \to \mathbb{R}$ and $t \in \mathbf{dom}\,(f)$. The function $f^*\colon\ \mathbb{R}^n \to \mathbb{R}$, defined as

$$f^*(t) = \sup_{x \in \mathbf{dom}\,(f)} t^T x - f(x) \tag{1.2}$$

is called a *conjugate* of the function f. The domain of the conjugate function consists of $t \in R^n$ for which the supremum is finite, that is for which the difference $t^T x - f(x)$ is bounded above on $\mathbf{dom}\,(f)$.

A conjugate f^* is a convex function since it is the pointwise supremum of a family of convex (indeed, affine) functions of the variable t. This is true whether or not f is convex.

Convex conjugate has many useful proprieties, many of them are listed and proved in [BV09].

Convexity with respect to generalized inequalities

We now consider generalizations of the notions of monotonicity and convexity, using generalized inequalities instead of the usual ordering on $\mathbb{R}$.

Definition 1.1.17 (Generalized monotonicity). Suppose $K \succeq \mathbb{R}^n$ is a proper cone with associated generalized inequality $\succeq_K$. A function $f\colon\ \mathbb{R}^n \to \mathbb{R}$ is called K-nondecreasing if

$$x \succeq_K y \Rightarrow f(x) \leqslant f(y),$$

and K-increasing if

$$x \succeq_K y \quad \text{and} \quad \Rightarrow f(x) < f(y).$$

We define K-nonincreasing and K-decreasing functions in a similar way.

Example 1.1.1 (Matrix monotone functions). A function $f\colon\ S^n \to \mathbb{R}$ is called matrix monotone (increasing, decreasing) if it is monotone with respect to the positive semidefinite cone. Examples of matrix monotone functions

- A function $\mathrm{tr}\,(WX)$, where $W \in S^n$, is matrix nondecreasing if $W \preceq 0$, and matrix increasing if $W \prec 0$ (conversly, it is matrix nonincreasing if $W \succeq 0$, and matrix decreasing if $W \succ 0$).

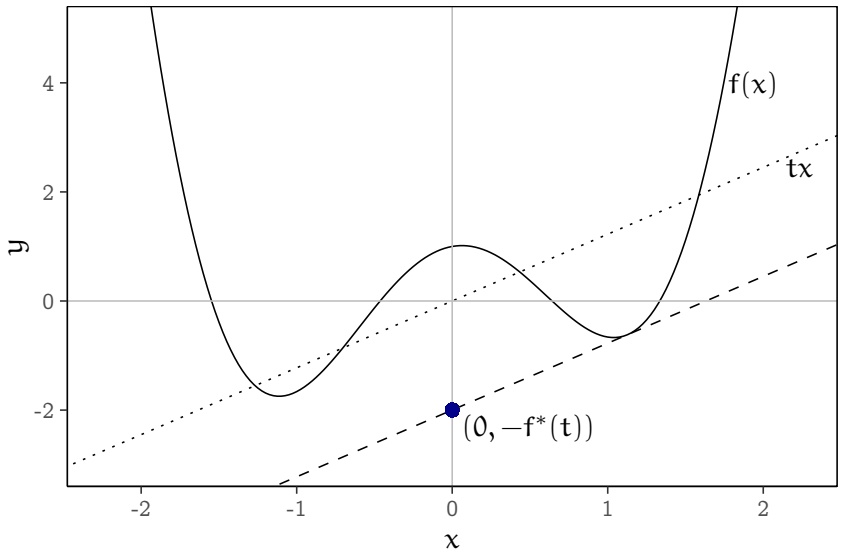


Figure 1.3 – A function $f\colon \mathbb{R} \to \mathbb{R}$, and its conjugate for $t = 1.225$. The conjugate function $f^*(t)$ is the maximum gap between the linear function tx and $f(x)$ on the domain. If f is differentiable, this occurs at a point x where $f'(x) = t$.

- A function $\operatorname{tr}\left(X^{-1}\right)$ is matrix decreasing on S^n_{++}.
- A function $\det X$ is matrix increasing on S^n_{++}, and matrix nondecreasing on S^n_{+}.

1.2 Convex optimization problems

1.2.1 Optimization problems

Basic terminology

We use the notation

$$\begin{array}{rll} \text{minimize} & f_0(x) & \\ \text{subject to} & f_i(x) \leqslant 0, & i = 1, 2, \ldots, m \\ & h_i(x) = 0, & i = 1, 2, \ldots, p \end{array} \tag{1.3}$$

to describe the problem of finding an x that minimizes $f_0(x)$ among all x that satisfy the conditions $f_i(x) \leqslant 0$ for $i = 1, 2, \ldots, m$ and $h_i(x) = 0$ for $i = 1, 2, \ldots, p$.

We call $x \in \mathbb{R}^n$ the *optimization variable* and the function $f_0\colon \mathbb{R}^n \to \mathbb{R}$ the *objective function*. The inequalities $f_i(x) \leqslant 0$ are called *inequality constraints*, and the corresponding functions $f_i\colon \mathbb{R}^n \to \mathbb{R}$ are called the *inequality constraint functions*. Likewise, the equations $h_i(x) = 0$ are called the *equality constraints*, and the functions $h_i\colon \mathbb{R}^n \to \mathbb{R}$ are the *equality constraint functions*. If there are no constraints (i.e. m and p are equal to zero) we say the problem (1.3) is *unconstrained*.

Definition 1.2.1 (Domain of the optimization problem). The set of points for which the objective and all constraint functions are defined,

$$\mathcal{D} = \bigcap_{i=0}^{m} \mathbf{dom}\,(f_i) \cap \bigcap_{i=1}^{p} \mathbf{dom}\,(h_i)$$

is called a *domain* of the optimization problem (1.3).

As points from domain might do not meet constraints we need to distinguish points that do.

Definition 1.2.2 (Feasibility). A point $x \in \mathcal{D}$ is called a *feasible* if it satisfies the constraints. The problem (1.3) is said to be feasible if there exists at least one feasible point, and *infeasible* otherwise. The set of all feasible points is called a *feasible set* or the *constraint set*.

Among the feasible points we might find the solutions for problem (1.3).

Definition 1.2.3 (Optimal value). The *optimal value* p^* of the problem (1.3) is defined as

$$p^* = \inf \{f_0(x) \colon f_i(x) \leqslant 0, i = 1, 2, \ldots, m; \;\; h_i(x) = 0, i = 1, 2, \ldots, p\}.$$

We allow p^* to take on the extended values $\pm\infty$. If the problem is infeasible, we have $p^* = \infty$ (following the standard convention that the infimum of the empty set is ∞). If there are feasible points x_k with $f_0(x_k) \to -\infty$ as $k \to \infty$, then $p^* = -\infty$, and we say the problem (1.3) is unbounded below.

Optimal and locally optimal points

Definition 1.2.4 (Optimal point, optimal set). We say x^* is an *optimal point*, or solves the problem (1.3), if x^* is feasible and $f_0(x^*) = p^*$. The set of all optimal points is the *optimal set*, denoted

$$X_{\text{opt}} = \{x \colon f_i(x) \leqslant 0, i = 1, \ldots, m; \;\; h_i(x) = 0, i = 1, 2, \ldots, p; \;\; f_0(x) = p^*\}.$$

If there exists an optimal point for the problem (1.3), we say the optimal value is *attained*, and the problem is *solvable*. If X_{opt} is empty, we say the optimal value is *not attained* (this always occurs when the problem is unbounded from below).

A feasible point x with $f_0(x) \leqslant p^* + \epsilon$ (where $\epsilon > 0$) is called an ϵ-suboptimal, and the set of all ϵ-suboptimal points is called an ϵ-suboptimal set for the problem (1.3).

Definition 1.2.5 (Locally optimal point). We say a feasible point x is *locally optimal* if there is an $r > 0$ such that

$$f_0(x) = \inf\{f_0(z) \colon f_i(z) \leqslant 0, i = 1, 2, \ldots, m; \;\; h_i(z) = 0, i = 1, 2, \ldots, p; \;\; \|z - x\|_2 \leqslant r\}.$$

In other words, x solves the optimization problem

$$\begin{aligned} \text{minimize} \quad & f_0(z) \\ \text{subject to} \quad & f_i(z) \leqslant 0, \quad i = 1, 2, \ldots, m \\ & h_i(z) = 0, \quad i = 1, 2, \ldots, p \\ & \|z - x\|_2 \leqslant r \end{aligned}$$

with variable z.

Roughly speaking, this means x minimizes f_0 over nearby points in the feasible set.

If x is feasible and $f_i(x) = 0$, we say the i-th inequality constraint $f_i(x) \leqslant 0$ is *active* at x. If $f_i(x) < 0$, we say the constraint $f_i(x) \leqslant 0$ is *inactive*. We say that a constraint is *redundant* if deleting it does not change the feasible set.

Remark 1.2.1. The equality constraints are active at all feasible points.

Standard form problems

There are many ways of manipulation optimization problems to obtain standard formulation (1.3), some techniques were presented in [Boy+10], chapter 4.

1.2.2 Convex optimization

Standard form

A *convex optimization* problem is one of the form

$$\begin{aligned} \text{minimize} \quad & f_0(x) \\ \text{subject to} \quad & f_i(x) \leqslant 0, \quad i = 1, 2, \ldots, m \\ & a_i^T(x) = b_i, \quad i = 1, 2, \ldots, p, \end{aligned} \tag{1.4}$$

where $f_1, \ldots, f_m$ are convex functions. Comparing (1.4) with the standard form problem (1.3), the convex problem has three additional requirements:

- the objective function must be convex,
- the inequality constraint functions must be convex,
- the equality constraint functions $h_i = a_i^T x - b_i$ must be affine.

Note an important property: The feasible set of a convex optimization problem is convex, since it is the intersection of the domain of the problem

$$D = \bigcap_{i=0}^{m} \mathbf{dom}\,(f_i)\,,$$

which is a convex set, with m (convex) sets $\{x\colon\ f_i(x) \leqslant 0\}$ and p hyperplanes $\{x\colon\ a_i^T x - b_i\}$. Thus in a convex optimization problem, we minimize a convex objective function over a convex set.

Remark 1.2.2. Without loss of generality we can assume that $a_i \neq 0$: if $a_i = 0$ and $b_i = 0$ for some i, then the i-th equality constraint can be deleted; if $a_i = 0$ and $b_i \neq 0$, the i-th equality constraint is inconsistent, and the problem is infeasible.

Concave maximization problems We will also refer to

$$\begin{aligned} \text{maximixe} \quad & f_0(x) \\ \text{subject to} \quad & f_i(x) \leqslant 0, \quad i = 1, 2, \ldots, m \\ & a_i^T(x) = b_i, \quad i = 1, 2, \ldots, p, \end{aligned}$$

as a convex optimization problem if the objective function f_0 is concave, and the inequality constraint functions $f_p, \ldots, f_m$ are convex. This concave maximization problem is effortlessly solved by minimizing the convex objective function $-f_0$. All of the results, conclusions are easily transposed to the maximization case.

Local and global optima

A fundamental property of convex optimization problem is a theorem presented below.

Lemma 1.2.1. *For problems of the form* (1.4) *any locally optimal point is also (globally) optimal*

Proof. Suppose that x is locally optimal for a convex optimization problem, that is x is feasible and

$$f_0(x) = \inf\{f_0(z)\colon \text{ is feasible};\ \ \|z - x\|_2 \leqslant r\}, \tag{1.5}$$

for some $r > 0$. Now suppose that x is not globally optimal, that is there is a feasible y such that $f_0(y) < f_0(x)$. Evidently $\|y - x\|_2 > r$, since otherwise $f_0(x) \leqslant f_0(y)$. Consider the point z given by

$$z = (1 - \theta)x + \theta y, \quad \theta = \frac{r}{2\|y - x\|_2}.$$

Thus we have $\|z - x\|_2 = r/2 < r$, and by convexity of the feasible set, z is feasible. By convexity of f_0 we have

$$f_0(z) \leqslant (1 - \theta)f_0(x) + \theta f_0(y) < f_0(x),$$

which contradicts (1.5). Hence there exists no feasible y with f0(y) < f0(x), that is x is globally optimal. ■

An optimality criterion for differentiable f_0

Suppose that the objective f_0 in a convex optimization problem is differentiable, so that for all $x, y \in \mathbf{dom}\,(f_0)$,

$$f_0(y) \geqslant f_0(x) + \nabla f_0(x)^T (y - x)$$

Let X denote the feasible set of underlying problem, then x is optimal if and only if $x \in X$ and

$$\nabla f_0(x)^T (y - x) \geqslant 0 \tag{1.6}$$

for all $y \in X$. This optimality criterion can be understood geometrically: if $\nabla f_0(x) \neq 0$, it means that $-\nabla f_0(x)$ defines a supporting hyperplane to the feasible set at x.

Proof of this condition can can be found in subsection 4.2.3, [BV09].

1.2.3 Convex problems

Linear optimization problems

When the objective and constraint functions are all affine, the problem is called a *linear program* (LP). A general linear program has the form

$$\begin{array}{ll} \text{minimize} & c^T x + d \\ \text{subject to} & Gx \preceq j \\ & Ax = b, \end{array}$$

where $G \in \mathbb{R}^{m \times n}$, $A \in \mathbb{R}^{p \times n}$ and other variables are vectors in $\mathbb{R}^n$. Linear programs are, of course, convex optimization problems.

The geometric interpretation of an LP: the feasible set of the LP is a polyhedron P; the problem is to minimize the affine function $c^T x + d$ over P.

Quadratic optimization problems

The convex optimization problem (1.4) is called a *quadratic program* (QP) if the objective function is (convex) quadratic, and the constraint functions are affine. A quadratic program can be expressed in the form

$$\begin{array}{ll} \text{minimize} & (1/2)x^T P x + q^T x + r \\ \text{subject to} & Gx \preceq h \\ & Ax = b, \end{array}$$

where $P \in S^n_+$, $G \in \mathbb{R}^{m \times n}$, $A \in \mathbb{R}^{p \times n}$ and other variables are vectors in $\mathbb{R}^n$. In a quadratic program, we minimize a convex quadratic function over a polyhedron. Quadratic programs include linear programs as a special case, by taking $P = 0$.

Generalized inequality constraints and conic problems

One very useful generalization of the standard form convex optimization problem (1.4) is obtained by allowing the inequality constraint functions to be vector valued, and using generalized inequalities in the constraints:

$$\begin{array}{ll} \text{minimize} & f_0(x) \\ \text{subject to} & f_i(x) \preceq_{K_i} 0, \quad i = 1, 2, \ldots, m \\ & Ax = b \end{array} \tag{1.7}$$

where $f_0 \colon \mathbb{R}^n \to \mathbb{R}$, the sets $K_i \subseteq \mathbb{R}^{k_i}$ are proper cones, and the functions $f_i \colon \mathbb{R}^n \to \mathbb{R}^{k_i}$ are K_i-convex. This problem is called a (standard form) *convex optimization problem with generalized inequality constraints*. Problem (1.3) is a special case with K_i equal to nonnegative real numbers for every i.

Many of the results for ordinary convex optimization problems hold for problems with generalized inequalities. Some examples

- The feasible set, any sublevel set, and the optimal set are convex.
- Any point that is locally optimal for the problem (1.7) is globally optimal.
- The optimality condition for differentiable f_0, given by (1.6), holds without any change.

By the analogy to linear programming, we refer to the conic form problem

$$\begin{array}{ll} \text{minimize} & c^T x \\ \text{subject to} & x \preceq_K 0 \\ & Ax = b \end{array}$$

as a *conic form problem* in standard form.

The convex optimization problems with generalized inequality constraints are the main method we will use in Chapter 3 to obtain a solution for our problem. The subclass which is used by us is optimization over the set of matrices.

Semidefinite programming When K is S_+^n, the cone of positive semidefinite $n \times n$ matrices, the associated conic form problem is called a *semidefinite program* (SDP), and has the form

$$\begin{array}{ll} \text{minimize} & c^T x \\ \text{subject to} & x_1 F_1 + \cdots + x_m f_m \preceq 0 \\ & Ax = b \end{array} \tag{1.8}$$

where $G, F_1, \ldots, F_m \in S^n$, and $A \in \mathbb{R}^{p \times m}$. If the matrices $G, F_1, \ldots, F_m$ are all diagonal, then the linear matrix inequality in the problem (1.8) is equivalent to a set of n linear inequalities, and the SDP reduces to a linear program.

1.3 Duality

1.3.1 The Lagrange dual function

The Lagrangian

Recall a standard optimization problem (1.3)

$$\begin{array}{ll} \text{minimize} & f_0(x) \\ \text{subject to} & f_i(x) \leqslant 0, \quad i = 1, 2, \ldots, m \\ & h_i(x) = 0, \quad i = 1, 2, \ldots, p, \end{array} \tag{1.9}$$

with variable $x \in \mathbb{R}^n$. Assume its domain $\mathcal{D}$ is nonempty, and denote the optimal value of (1.9) by p^*. Problem is not necessary convex.

The idea behind Lagrangian duality is to take constraints in (1.9) into account by augmenting the objective function f_0 with a weighted sum of the constraint functions.

Definition 1.3.1 (Lagrangian). Define *Lagrangian* as the function $L : \mathbb{R}^n \times \mathbb{R}^m \times \mathbb{R}^p \to \mathbb{R}$ associated with the problem (1.9) as

$$L(x, \mu, \nu) = f_0 + \sum_{i=1}^{m} \mu_i f_i(x) + \sum_{i=1}^{p} \nu_i h_i(x),$$

with $\mathbf{dom}\,(L) = \mathcal{D} \times \mathbb{R}^m \times \mathbb{R}^p$.

We refer to μ_i and ν_j as the *Lagrange multiplier* associated with i-th inequality constraint $f_i(x) \leqslant 0$ and j-th equality constraint $h_j(x) = 0$, respectively. The vectors μ and ν are called *dual variables* or *Lagrange multiplier vectors* associated with the problem (1.9).

The lagranigian can be rewritten in the terms of inner product as

$$L(x, \mu, \nu) = f_0 + \langle \mu, f(x) \rangle + \langle \nu, h(x) \rangle,$$

where the functions $f(x)$ and $h(x)$ are vectorized forms of the underlying functions.

The Lagrange dual function

Definition 1.3.2 (Lagrange dual function). Define *Lagrangian dual function* (or just *dual function*) as the function $g : \mathbb{R}^m \times \mathbb{R}^p \to \mathbb{R}$ as the minimum value of the Lagrangian over x

$$g(\mu, \nu) = \inf_{x \in \mathcal{D}} L(x, \mu, \nu) = \inf_{x \in \mathcal{D}} \left(f_0 + \sum_{i=1}^{m} \mu_i f_i(x) + \sum_{i=1}^{p} \nu_i h_i(x) \right),$$

with $\mu \in \mathbb{R}^m$ and $\nu \in \mathbb{R}^p$.

When the Lagrangian is unbounded below in x, the dual function takes on the value $-\infty$. Since the dual function is the pointwise infimum of a family of affine functions of (μ, ν), it is concave, even when the problem (1.9) is not convex.

Lower bounds

The dual function yields lower bound on the optimal value p^* of the problem (1.9).

Lemma 1.3.1. *For any $\mu \succeq 0$ and any ν we have*

$$g(\mu, \nu) \leqslant p^*. \tag{1.10}$$

Proof. Suppose $\tilde{x}$ is a feasible point for the problem (1.9), we have

$$\sum_{i=1}^{m} \mu_i f_i(\tilde{x}) + \sum_{i=1}^{p} \nu_i h_i(\tilde{x}) \leqslant 0,$$

since each term in the first sum is nonpositive, and each term in the second sum, zero. Therefore

$$L(\tilde{x}, \mu, \nu) = f_0(\tilde{x}) + \sum_{i=1}^{m} \mu_i f_i(\tilde{x}) + \sum_{i=1}^{p} \nu_i h_i(\tilde{x}) \leqslant f_0(\tilde{x}).$$

Hence

$$g(\mu, \nu) = \inf_{x \in \mathcal{D}} L(x, \mu, \nu) \leqslant L(\tilde{x}, \mu\nu) \leqslant f_0(\tilde{x}).$$

Since $g(\mu, \nu) \leqslant f_0(\tilde{x})$ holds for every feasible point $\tilde{x}$, the inequality (1.10) follows. ■

When $g(\mu, \nu) = -\infty$ The inequality (1.10) holds, but is pointless. The dual function gives a nontrivial lower bound on p^* only when $\mu \succeq 0$ and $(\mu, \nu) \in \mathbf{dom}\,(g)$, that is $g(\mu, \nu) > -\infty$. In next sections we refer to a pair (μ, ν) with $\mu \succeq 0$ and $(\mu, \nu) \in \mathbf{dom}\,(g)$ as *dual feasible*, for reasons that will become clear later. Two following plots show the idea.

The Lagrange dual function and conjugate functions

The conjugate function defined by the (1.2) and Lagrange dual function are closely connected. To see this relation, consider the obvious problem

$$\begin{aligned} \text{minimize} \quad & f_0(x) \\ \text{subject to} \quad & x = 0. \end{aligned}$$

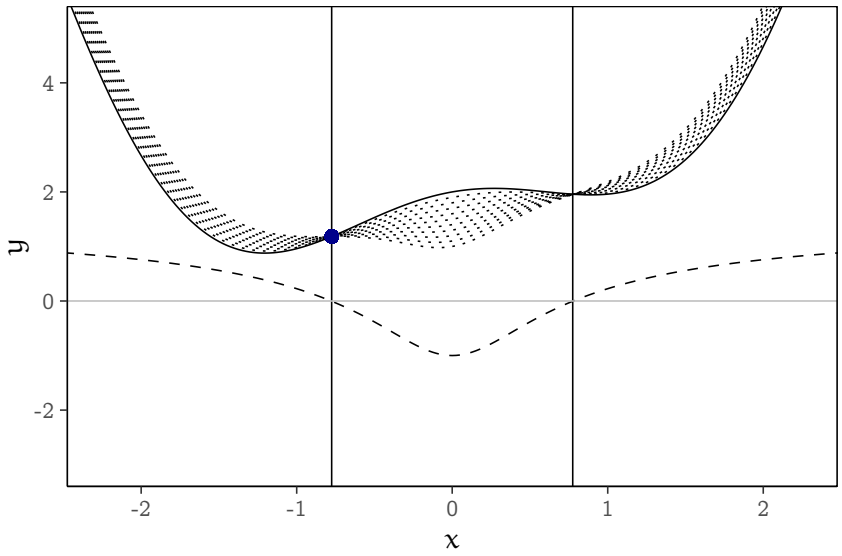


Figure 1.4 – The solid curve shows the objective function $f_0(x) = x/2+2-x^2\cos x$, and the dashed curve shows the constraint function $f_1(x) = -1 + \arctan^2 2x$. The feasible set is the interval $[-0.78, 0.78]$, which is indicated by the two solid vertical lines. The optimal solution is given by $x^* = -0.78$, $p^* = 1.18$ (shown as a dot). The dotted curves show $L(x,\mu)$ for $\mu = 0.1, 0.2, \ldots, 1.0$. Each of these has a minimum value smaller than p^*, since on the feasible set (and for $\mu \geqslant 0$) we have $L(x,\mu) \leqslant f_0(x)$.

The Lagrangian of this problem is given by $L(x,\nu) = f(x) + \nu^T x$, the dual function by

$$g(\nu) = \inf_x \left(f(x) + \nu^T x\right) = -\sup_x \left((-\nu)^T x - f(x)\right) = -f^*(-\nu).$$

More useful is generalization of this connection, consider an optimization problem with linear inequality and equality constraints

$$\begin{array}{ll} \text{minimize} & f_0(x) \\ \text{subject to} & Ax \preceq b \\ & Cx = d. \end{array} \tag{1.11}$$

The conjugate of f_0 we can be used to obtain the dual function for the problem (1.11) as

$$\begin{aligned} g(\mu,\nu) &= \inf_x \left(f_0(x) + \mu^T(Ax-b) + \nu^T(Cx-d)\right) \\ &= -b^T\mu - d^T\nu + \inf_x \left(f_0(x) + (A^T\mu + C^T\nu)^T x\right) \\ &= -b^T\mu - d^T\nu - f_0^*(-A^T\mu - C^T\nu). \end{aligned} \tag{1.12}$$

The domain of g comes from the domain of f_0^*

$$\mathbf{dom}\,(g) = \{(\mu,\nu):\ -A^T\mu - C^T\nu \in \mathbf{dom}\,(f_0^*)\}.$$

1.3.2 The Lagrange dual problem

Each pair of parameters (μ,ν) gives us a lower bound of the solution of problem (1.9) by the Lagrange dual function. This lower bound depends on those parameters. A natural question is: What is the best lower bound that can be obtained from the Lagrange dual function?

This leads us to another optimization problem

$$\begin{array}{ll} \text{maximize} & g(\mu,\nu) \\ \text{subject to} & \mu \succeq 0. \end{array} \tag{1.13}$$

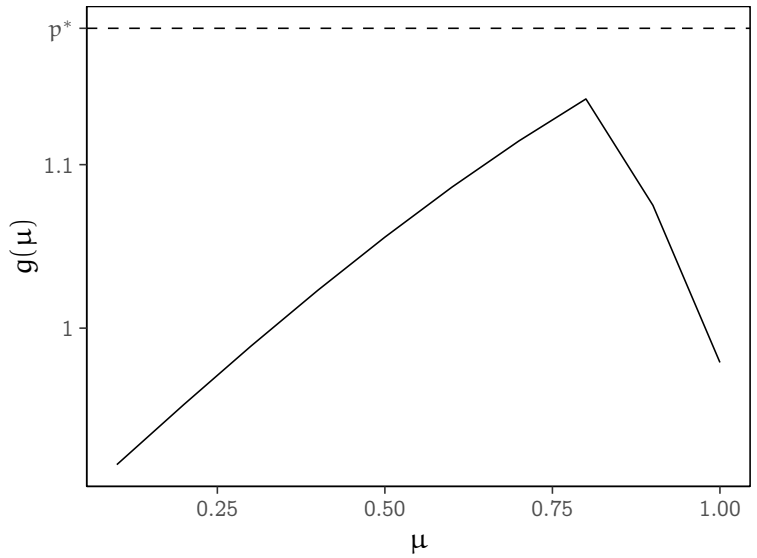


Figure 1.5 – The dual function $g(\mu)$ for the problem in Figure 1.4. Neither f_0 nor f_1 is convex, but the dual function is concave. It is visible that p^* is bounded from below by $g(\mu)$.

This problem is called a *Lagrange dual problem* (or just *dual problem*) associated with the optimization problem (1.9). In this context the original problem (1.9) is sometimes called the *primal problem*. The term *dual feasible*, to describe a pair (μ^*, ν^*) with $\mu \succeq 0$ and $g(\mu^*, \nu^*) > -\infty$, now makes sense. It means, as the name implies, that pair (μ^*, ν^*) is feasible for the dual problem (1.13).

We refer to (μ^*, ν^*) as *dual optimal* or *optimal Lagrange multipliers* if they are optimal for the problem (1.13).

Since the objective function of (1.13) is concave and the constraint is convex, this is a convex optimization problem (we consider minimization $-g$). That is true regardless convexity of the primal problem (1.9).

Explicit dual constraints

Usually, the dimension of a domain of dual function $\mathbf{dom}\,(g)$ is smaller than $m + p$. In many cases we can identify the affine hull of $\mathbf{dom}\,(g)$, and describe it as a set of linear equality constraints. Roughly speaking, this means we can identify the equality constraints that are 'hidden' or 'implicit' in the objective g of the dual problem (1.13). In this case we can form an equivalent problem, in which these equality constraints are given explicitly as constraints.

Weak duality

By d^* we denote the optimal value of Lagrange dual problem, that is the best lower bound on p^* that can be obtained. We can write

$$d^* \leqslant p^*. \tag{1.14}$$

This simple, but important inequality is called a *weak duality*. It holds even if the original problem is not convex and is true when d^* and p^* are infinite. If $p^* = -\infty$, that is the primal problem is unbounded from below, we must have $d^* = -\infty$. Conversely, if $d^* = \infty$, the primal problem is infeasible, that is $p^* = \infty$.

The difference $p^* - d^*$ is called an *optimal duality gap*. It gives the gap between the optimal value of the primal problem and the best (i.e. the greatest) lower bound on it that can be obtained from the Lagrange dual function. It is always nonnegative.

Strong duality

If the d^* attains p^*, that is the optimal duality gap is zero, then we say that *strong duality* holds. This means that the best bound that can be obtained from the Lagrange dual function is tight. Strong duality does not, in general, hold. Usually, when (1.9) is a convex problem, it does (but not always). There are many results that establish conditions on the problem under which strong duality holds. These conditions are called *constraint qualifications*.

1.3.3 Optimality conditions

Suboptimality and stopping criteria

If we can find a dual feasible (μ, ν), we establish a lower bound on the optimal value of the primal problem: $p^* \geqslant g(\mu, \nu)$. Thus a dual feasible point (μ, ν) provides a proof that $p^* \geqslant g((\mu, \nu))$. Strong duality means there exist arbitrarily good certificates for this fact.

Dual feasible points allow us to bound how suboptimal a given feasible point is, without knowing the exact value of p^*. Indeed, if x is primal feasible and (μ, ν) is dual feasible, then

$$f_0 - p^* \leqslant f_0 - g(\mu, \nu).$$

In particular, this establishes that x is ϵ-suboptimal, with $\epsilon = f_0(x) - g(\mu, \nu)$, it also establishes that the pair $\mu, \nu)$ is ϵ-suboptimal for the dual problem.

We refer to a gap between primal and dual objectives,

$$f_0 - g(\mu, \nu),$$

as the *duality gap* associated with the primal feasible point x and dual feasible point (μ, ν). A primal-dual feasible pair $x, (\mu, \nu)$ localizes the optimal value of the primal (and dual) problems to an interval

$$p^* \in [g(\mu, \nu), f_0(x)],$$
$$d^* \in [g(\mu, \nu), f_0(x)],$$

the width of which is the duality gap.

If the duality gap of the primal dual feasible pair $x, (\mu, \nu)$ is zero, that is, $f_0(x) = g(\mu, \nu)$, then the point x is primal optimal and the point (μ, ν) is dual optimal. We can think of the point (μ, ν) as a certificate that proves the point x is optimal and, similarly, think of the point x as a certificate that proves the point (μ, ν) is dual optimal.

These observations can be used in optimization algorithms to provide nonheuristic stopping criteria. The stopping criteria for ADMM are presented in the next section.

Solving the primal problem via the dual If strong duality holds and a dual optimal solution (μ^*, ν^*) exists, then any primal optimal point is also a minimizer of $L(x, \mu^*, \nu^*)$. This sometimes allows us to compute a primal optimal solution from a dual optimal solution.

1.3.4 Conclusion

Duality is an extremely important concept in the theory of mathematical optimization. It can be extended by theorems of alternatives, similarly, every concept from this section could be presented in the term of generalized inequalities. This theory is essential in this thesis, we just wanted to present the reader the basics of convex optimization before introducing the concept of ADMM.

1.4 ADMM

1.4.1 Precursors of ADMM

In this subsection, we briefly introduce and review two optimization algorithms that are forerunners to the Alternating Direction Method of Multipliers. Those algorithms are not used later, but they

provide some intuition and background.

Dual ascent

Let's consider the equality-constrained convex optimization problem

$$\begin{aligned} &\text{minimize} && f(x) \\ &\text{subject to} && Ax = b, \end{aligned} \tag{1.15}$$

with the variable $x \in \mathbb{R}$, where $A \in \mathbb{R}^{m \times n}$ and $f\colon \mathbb{R}^n \to \mathbb{R}$ is convex. The Lagrangian for the problem (1.15) is given by

$$L(x, \nu) = f(x) + \nu^T (Ax - b),$$

with the dual function given by

$$g(\nu) = \inf_x l(x, \nu) = -f^*(A^T \nu) - b^T \nu,$$

where ν is the dual variable or Lagrange multiplier and f^* is the convex conjugate of the function f, both defined earlier. We know that the dual problem is given by

$$\text{minimize} \quad g(\nu)$$

for $y \in \mathbb{R}^m$. Assuming that strong duality holds, the optimal values of the primal and dual problems are the same and the primal optimal point x^* can be recovered from the dual optimal point ν^* as

$$x^* = \arg\min_x L(x, \nu^*),$$

provided there is only one minimizer of this Lagrangian, for example, the function f is strictly convex. Generally, we will use $\arg\min_x f(x)$ to denote any minimizer of the function f.

The dual ascent method is solved using the gradient ascent. Assuming differentiability of the dual function g, we can evaluate the gradient $\nabla g(\nu)$. Having the minimum value $x^+ = \arg\min_x L(x, \nu)$ we get the gradient

$$\nabla g(\nu) = Ax^+ - b,$$

which is the residual for the equality constraint. Dual ascent iterations are given by

$$x^{k+1} = \arg\min_x L(x, \nu^k) \tag{1.16}$$

$$\nu^{k+1} = \nu^k + \alpha^k (Ax^{k+1} - b), \tag{1.17}$$

where $\alpha^k > 0$ is the step size in k-th iteration. The eq. (1.16) is the x-minimization and the eq. (1.17) is the dual variable update. This algorithm is called *dual ascent*, since, with appropriate choice of x^k, the dual function increases in each step, that is $g(\nu^{k+1}) > g(\nu^k)$.

The method presented above can be used even in some cases when the dual function g is not differentiable. In such situations, the residual $Ax^+ - b$ is not the gradient, but the negative subgradient of $-g$. This imply different choice of the parameter α^k than when the dual function g is differentiable, and convergence is not monotone; it is often the case that $g(\nu^{k+1}) \not> g(\nu^k)$, then the algorithm is usually called the dual *subgradient method* [Sho85].

If α^k is chosen properly and several other assumptions hold, then α^k converges to the optimal point and ν^k converges to the optimal dual point. However, these assumptions often do not hold, so dual ascent often cannot be used. For example, if a function f is the nonzero affine function of any component of x, then the x-update (1.16) fails, since Lagrangian is unbounded below in x for most ν [Boy+10].

Dual decomposition

The strongest benefit of the dual ascent method is that in some cases it can lead to a decentralized algorithm. Suppose, that an objective function f is *separable*, that is

$$f(x) = \sum_{i=1}^{N} f_i(x_i)$$

where $x = (x_1, \ldots, x_N)$ and the variables $x_i \in \mathbb{R}^{n_i}$ are the sub-vectors of the vector x. By partitioning the matrix A in the same manner as

$$A = [A_1 \ldots A_N],$$

so that $Ax = \sum_{i=1}^{N} A_i x_i$, the Lagrangian can be written as

$$L(x, \nu) = \sum_{i=1}^{N} L_i(x_i, \nu_i) = \sum_{i=1}^{N} \left(f_i(x_i) + \nu^T A_i x_i - \frac{1}{N} \nu^T b \right). \tag{1.18}$$

The Lagrangian (1.18) is separable in x with respect to given partition. This means that the x-minimization step splits into N separate problems, each of them can be solved using algorithm

$$x_i^{k+1} = \arg\min_{x_i} L_i(x_i, \nu^k) \tag{1.19}$$

$$\nu^{k+1} = \nu^k + \alpha^k (Ax^{k+1} - b). \tag{1.20}$$

The x-minimization step (1.19) is executed independently for each $i = 1, 2, \ldots, N$, so they can be calculated in parallel. In this case, the dual ascent method is called a *dual decomposition*.

Each iteration on (1.19) and (1.20) require, so-called, a *broadcast* and a *gather* operation. In order to calculate the dual update step, the equality constraint residual inputs $A_i x_i^{k+1}$ are gathered in order to compute the residual $Ax^{k+1} - b$. Once the global dual variable ν^{k+1} is obtained, it must be broadcasted back to the N individual x_i minimization steps.

Although the dual decomposition is the old idea in optimization, and traces back at least to the early 1960s, it gives intuitions useful for ADMM.

Augmented Lagrangian and the Method of Multipliers

Now we introduce a concept of Augmented Lagrangian methods, they were developed to bring robustness to the dual ascent method, and in particular, to yield convergence without assumptions like strict convexity or finiteness of the underlying function f. An *augmented Lagrangian* $\mathcal{L}$ for the simple problem (1.15) is defined as

$$\mathcal{L}_\rho(x, \nu) = f(x) + \nu^T (Ax - b) + \frac{\rho}{2} \|Ax - b\|_2^2, \tag{1.21}$$

or in inner product form as

$$\mathcal{L}_\rho(x, \nu) = f(x) + \langle \nu, Ax - b \rangle + \frac{\rho}{2} \|Ax - b\|_2^2,$$

where $\rho > 0$ is called a *penalty parameter*. The augmented Lagrangian in the eq. (1.21) can be viewed as the ordinary Lagrangian associated with the problem

$$\begin{aligned} \text{minimize} \quad & f(x) + \frac{\rho}{2} \|Ax - b\|_2^2 \\ \text{subject to} \quad & Ax = b. \end{aligned} \tag{1.22}$$

Since for any feasible variable x the term added to the objective is zero, this problem is clearly equivalent to the original problem (1.15). An associated dual function is given by $g_\rho(\nu) = \inf_x \mathcal{L}(x, \nu)$.

The main advantage of augmenting the Lagrangian comes from the fact, that it can be shown that g_ρ is differentiable under rather mild conditions. More detailed information and derivation can be found in chapter 17 of [NW06].

The gradient of the augmented dual function is found the same way as with the ordinary Lagrangian, that is by minimizing over the variable x, and then evaluating the resulting equality constraint residual.

Applying dual ascent method to the modified problem generate variable updates

$$x^{k+1} = \arg\min_x \mathcal{L}_\rho(x, \nu^k) \tag{1.23}$$

$$\nu^{k+1} = \nu^k + \rho(Ax^{k+1} - b), \tag{1.24}$$

which are known as a *method of multipliers* for equality constraint setup. This is the same as the standard dual ascent, except that the x-minimization step uses the augmented Lagrangian, and the penalty parameter ρ is used instead of α^k in the dual update. The method of multipliers converges under far more general conditions than the dual ascent, for example, if the function f is not strictly convex and takes on ∞ value.

The choice of the parameter ρ as the step size in the eq. (1.23) and eq. (1.24) is easy to motivate. For simplicity assume that the function f is differentiable, although it is not required. The optimality conditions for (1.15) are a primal and dual feasibility. By definition, the variable x^{k+1} minimizes $\mathcal{L}_\rho(x, \nu^k)$, thus we have

$$\begin{aligned} 0 &= \nabla_x \mathcal{L}_\rho(x^{k+1}, \nu^k) \\ &= \nabla_x f(x^{k+1}) + A^T \left(\nu^k + \rho(Ax^{k+1} - b)\right) \\ &= \nabla_x f(x^{k+1}) + A^T \nu^{k+1}. \end{aligned}$$

Thus by using ρ as the step size in the dual update, the iterate (x^{k+1}, ν^{k+1}) is dual feasible. As the method of multipliers proceeds, the primal residual $Ax^{k+1} - b$ converges to zero, meeting optimality conditions.

The greatly improved convergence properties of the method of multipliers over dual ascent come at a cost. When the function f is separable, the augmented Lagrangian $\mathcal{L}_\rho$ is not separable, so the x-minimization step (1.23) cannot be carried out separately in parallel for each x_i. This means that the basic method of multipliers cannot be used for decomposition. ADMM is the solution to this limitation.

1.4.2 Alternating Direction Method of Multipliers

Alternating Direction Method of Multipliers [ADMM] is the main convex optimization algorithm used in this thesis. It is based on the two algorithms described in the last section and omits their limitations.

Algorithm

ADMM blends decomposability of dual ascent with the great convergence properties of the method of multipliers. It can solve the problems of the form

$$\begin{aligned} \text{minimize} \quad & f(x) + g(y) \\ \text{subject to} \quad & Ax + By = c \end{aligned} \tag{1.25}$$

with variables $x \in \mathbb{R}^n$ and $y \in \mathbb{R}^m$, where $A \in \mathbb{R}^{p \times n}$, $B \in \mathbb{R}^{p \times m}$, and $c \in \mathbb{R}^p$. For now, we just assume convexity of the functions f and g, more detailed assumptions are discussed later. The difference from the equality-constrained problem (1.15) is that the main variable was split into two parts, x and y, so the objective function. The optimal value of the problem (1.25) is defined as

$$p^* = \inf\{f(x) + g(y) \colon Ax + By = c\}.$$

The augmented Lagrangian with penalty parameter $\rho > 0$ is given by

$$\mathcal{L}_\rho(x, y, \nu) = f(x) + g(y) + \nu^T(Ax + By - c) + \frac{\rho}{2}\|Ax + By - b\|^2.$$

ADMM iterations are given by

$$x^{k+1} = \arg\min_x \mathcal{L}_\rho(x, y^k, \nu^k) \tag{1.26}$$

$$y^{k+1} = \arg\min_y \mathcal{L}_\rho(x^{k+1}, y, \nu^k) \tag{1.27}$$

$$\nu^{k+1} = \nu^k + \rho(Ax^k + By^{k+1} - c). \tag{1.28}$$

The algorithm is blend of the dual ascent and the method of multipliers: it consist of the x-minimization step, the y-minimization step, and the dual update. As in the method of multipliers, the dual variable update uses the step size equal to the augmented Lagrangian parameter ρ. In the method of multipliers, the augmented Lagrangian is minimized jointly, in ADMM the variables x and y are updated in an alternating way, which accounts for the term *alternating direction*.

Convergence

In [Boy+10] authors showed that under two basic and general assumptions

1. functions f and g are closed, proper, and convex,
2. an unaugmented Lagrangian L_0 has a saddle point.

The first assumption can be expressed compactly using the epigraphs of the functions, that is the function f satisfies assumption 1 if and only if its epigraph is a closed nonempty convex set. Second assumption means that there exists, not necessarily unique, triple (x^*, z^*, y^*) that for every (x, y, z) the following inequalities hold

$$\mathcal{L}(x^*, z^*, y) \leqslant \mathcal{L}(x^*, z^*, y^*) \leqslant \mathcal{L}(x, z, y^*)$$

Under those assumptions, ADMM iterates satisfy the following:

- *Residual convergence*: $r^k \to 0$ as $k \to \infty$, that is the iterates approach feasibility.
- *Objective convergence*: $f(x^k) + g(y^k) \to p^*$ as $k \to \infty$, that is the objective function of the iterates approaches the optimal value.
- *Dual variable convergence*: $\nu^k \to \nu$ as $k \to \infty$, where ν^* is a dual optimal point.

The proper proof is included in [Boy+10], appdendix A. Moreover, in [Nis+15] authors analyzed parameters tunning and relaxation of above assumptions, they showed that ADMM converge in many different settings. We do not present the results, as this is not the main point of this thesis.

Convergence in practice Simple examples show that ADMM can be very slow to converge to high accuracy, although, it is often the case that ADMM converges to modest accuracy in a few tens iterations.

While in some cases it is possible to combine ADMM with a method presented in [EF98] producing a high accuracy solution from a low accuracy solution, in the general case ADMM is practically useful mostly in cases when modest accuracy is sufficient. Also, in the case of statistical and machine learning problems, solving a parameter estimation problem to very high accuracy often yields little to no improvement in actual prediction performance, the real metric of interest in applications. As we "belong" to this branch of mathematics, ADMM seems to be the perfect choice for us.

Optimality conditions

The necessary and sufficient optimality conditions for the problem (1.25) are a primal feasibility,

$$Ax^* + By^* - c = 0, \tag{1.29}$$

and dual feasibility,

$$0 \in \partial f(x^*) + A^T \nu^* \tag{1.30}$$
$$0 \in \partial f(y^*) + B^T \nu^*, \tag{1.31}$$

where ∂ denotes the subdifferential operator. In case of differentiability of the functions f and g, dual feasibility conditions are replaced by

$$0 = \nabla f(x^*) + A^T \nu^*$$
$$0 = \nabla f(y^*) + B^T \nu^*.$$

Since y^{k+1} minimizes $\mathcal{L}_\rho(x^{k+1}, y, \nu^k)$ by definition, we have

$$\begin{aligned} 0 &\in \partial f(y^{k+1}) + B^T \nu^k + \rho B^T (Ax^{k+1} + By^{k+1} - c) \\ &= \partial f(y^{k+1}) + B^T \nu^k + \rho r^{k+1} \\ &= \partial f(y^{k+1}) + B^T \nu^{k+1}. \end{aligned}$$

This means that at each state y^{k+1} and ν^{k+1} satisfies first dual feasibility condition 1.30, and attaining optimality comes down to satisfying (1.29) and 1.31. Since x^{k+1} minimizes $\mathcal{L}_\rho(x, y^k, \nu^k)$ by definition, we have

$$\begin{aligned} 0 &\in \partial f(x^{k+1}) + A^T \nu^k + \rho A^T (Ax^{k+1} + By^k - c) \\ &= \partial f(x^{k+1}) + A^T (\nu^k + \rho r^{k+1} + \rho B(y^k - y^{k+1})) \\ &= \partial f(x^{k+1}) + A^T \nu^{k+1} + \rho A^T B(y^k - y^{k+1}), \end{aligned}$$

equivalently

$$\rho A^T B(y^{k+1} - y^k) \in \partial f(x^{k+1}) + A^T \nu^{k+1}.$$

Denote left side as

$$s^{k+1} = \rho A^T B(y^{k+1} - y^k).$$

We refer s^{k+1} as the *dual residual* at the iteration $k+1$, and to $r^{k+1} = Ax^{k+1} + By^{k+1} - c$ as the *primal residual* at the iteration $k+1$. If those residuals converge to zero as ADMM proceed, the optimality conditions are met and it guarantees convergence of the algorthm. As stated earlier, this was shown in Appendix A of [Boy+10].

Stopping criteria

The residuals of the optimality conditions s^{k+1} and r^{k+1} can be related to a bound on the objective suboptimality of the current point, that is $f(x^k) + g(y^k) - p^*$. In the proof mentioned in the last paragraph, following inequality is shown

$$f(x^k) + g(y^k) - p^* \leqslant -(\nu^k)^T r^k + (x^k - x^*)^T s^k. \tag{1.32}$$

This means that when the residuals s^k and r^k are small, the objective suboptimality also must be small. As we do not know the real optimal value x^*, we cannot use this inequality directly.

In [Boy+10] authors suggested guessing or estimating d that follows $\|x^k - x^*\|_2 \leqslant d$. we have

$$f(x^k) + g(y^k) - p^* \leqslant -(\nu^k)^T r^k + d\|s^k\|_2 \leqslant \|\nu^{k^T}\|_2 \|r^k\|_2 + d\|s^k\|_2$$

Both, middle and right terms, could be used as an approximate bound on the objective suboptimality (which depends on our guess of d). The reasonable termination criterion should require the primal and dual residuals to be small, that is

$$\|r^k\|_2 \leqslant \epsilon_{\text{primal}} \quad \text{and} \quad \|s^k\|_2 \leqslant \epsilon_{\text{dual}}, \tag{1.33}$$

where $\epsilon_{\text{primal}} \geqslant 0$ and $\epsilon_{\text{dual}} \geqslant 0$ are feasibility tolerances.

In cited paper, authors suggest a reasonable value for the relative stopping criterion. The choice depends on the scale of the typical variable values.

1.4.3 Useful patterns

ADMM might be used to solve many different settings, but it is often the case that structure in f, g, A, and B can be exploited to carry out updates more efficiently. We consider only cases that are useful in our work. All cases are written for an x-update, but also applies for a y-update. In this section the x-update step is expressed as

$$x^+ = \arg\min_x \left(f(x) + \frac{\rho}{2}\|Ax - \xi\|_2^2 \right),$$

where $\xi = -By + c - u$ is a known constant vector defined before the variable update.

Proximity operator

First, let's assume that $A = I$, then update is

$$x^+ = \arg\min_x \left(f(x) + \frac{\rho}{2}\|x - \xi\|_2^2 \right). \tag{1.34}$$

This expression, as the function of ξ is denoted by $\mathbf{prox}_{f,\rho}(\xi)$ and is called a *proximity operator* of the function f with the penalty ρ. A similar operator used in variational analysis, called *Moreau envelope* or *Moreau-Yosida regularization* of the function f, is defined as

$$\bar{f}(\xi) = \inf_x \left(f(x) = \frac{\rho}{2}\|x - \xi\|_2^2 \right).$$

The x-minimization in the proximity operator is usually referred to as *proximal minimization*, it ties the x-minimization step to other well known ideas.

This pattern is useful in our work as a proximal operator for SLOPE was presented in [Bog+15]. More information about the proximal operator and, generally, proximal minimization can be found in [Par14; PSW15].

Decomposition

The idea behind decomposition is simple, but a proper derivation of statements below needs a lot of background from convex optimization, the description is limited to most important steps, for more precise background description see [Roc] and [Par14]. We present two types of decomposition useful in our applications.

Separability Suppose that $x = (x_1, \ldots, x_N)$ is a partition of the variable into subvectors and that f is block separable with respect to this partition, that is

$$f(x) = f_1(x_1) + \cdots + f_N(x_N),$$

where $x_i \in \mathbb{R}^{n_i}$ and $\sum_{i=1}^N n_i = N$. If the term $\|Ax\|_2^2$ (called *quadriatic term*) is also separable with respect to this partition, that is A^TA is block diagonal in the same way as x is, then the augmented Lagrangian $\mathcal{L}_\rho$ is also separable. This means that x-update can be done parallel, in each block at the same time, this property is called a *block separability*.

The special case of block separability called *component separability* is the case when the decomposition extends all the way to the individual components. The x-minimization step can

then be done by n *scalar minimizations*, which can in some cases be expressed analytically (but in any case can be computed very efficiently).

Moreover, if the function f is separable, then the proximity operator of f is fully separable, that is $(\mathbf{prox}_f(\xi))_i = \mathbf{prox}_{f_i}(\xi_i)$, where $\xi_i \in \mathbb{R}^{n_i}$. This important fact for our work is proved in [Par14], chapter 5.

Soft thresholding A good example of separability is the ℓ_1 based problem. Assume that $f(x) = \lambda\|x\|_1$ ($\lambda > 0$) and $A = I$. As the ℓ_1 norm is component separable, a scalar x_i-update is given by

$$x_i^+ = \arg\min_{x_i} \left(\lambda|x_i| + \frac{\rho}{2}(x_i - \xi_i)^2\right). \tag{1.35}$$

By using differential calculus we can obtain a simple closed-form solution.

The optimality condition for (1.35) is given by

$$0 = \rho(x_i - \xi_i) + \lambda\partial\|x_i\|_1$$

A first examine the case when $x_i \neq 0$, then, $\lambda\partial\|x_i\|_1 = \mathbf{sign}(x_i)$ and the optimum x_i^* is solution for

$$0 = \rho(x_i - \xi_i) + \lambda\,\mathbf{sign}(x_i)$$
$$x_i = \xi_i - \frac{\lambda}{\rho}\,\mathbf{sign}(x_i)$$

Substitute $\kappa = \frac{\lambda}{\rho}$. Note that if $x_i^* < 0$, then $\xi_i < -\kappa$ and if $x_i^* > 0$, then $\xi_i < -\kappa$, Thus $|\xi_i| > \kappa$ and $\mathbf{sign}(x_i^*) = \mathbf{sign}(\xi_i)$. Using that we get

$$x_i = \xi_i - \kappa\,\mathbf{sign}(\xi_i)$$

Coming back to the case when $x_i = 0$, the subdifferential of the ℓ_1-norm is the interval $[-1, 1]$ and the optimality condition is

$$0 \in \rho\xi_i + \lambda[-1, 1]$$
$$\xi_i \in [-\kappa, \kappa]$$
$$|\xi_i| < \kappa$$

Thus the combined solution for (1.35) is

$$x_i^+ = \mathrm{ST}_{\lambda/\rho}(\xi_i) \tag{1.36}$$

where the *soft thresholding* function ST_κ is defined as

$$\mathrm{ST}_\kappa(x) = \begin{cases} x - \kappa & \text{if } x > \kappa \\ 0 & \text{if } |x| \leqslant \kappa \\ x + \kappa & \text{if } x < -\kappa. \end{cases}$$

In the language of proximal minimization, soft thresholding is the proximity operator of ℓ_1 norm.

Constrained convex optimization

Now we proceed to a general example of ADMM usage. Consider the generic constrained convex optimization problem

$$\begin{aligned} \text{minimize} \quad & f(x) \\ \text{subject to} \quad & x \in C \end{aligned} \tag{1.37}$$

with variable $x \in \mathbb{R}^n$, where the function f and set C are convex. This problem can be rewritten in ADMM form as

$$\begin{aligned} \text{minimize} \quad & f(x) + \mathbb{I}_C(y) \\ \text{subject to} \quad & x - y = 0. \end{aligned}$$

This formulation of the problem (1.37) will be very useful in the main problem of this thesis.

1.5 Summary

We finish the chapter with the convenient pseudocode formulation of ADMM.

Algorithm 1 Alternative direction method of multipliers

$y_0 \leftarrow \tilde{y}$, $\nu_0 \leftarrow \tilde{\nu}$, $k \leftarrow 1$
$\mu \leftarrow \tilde{\rho} > 0$ ▷ initialize
while convergence criterion is not met **do**
  $x_k \leftarrow \arg\min_x L_\rho(x, y_{k-1}, \nu_{k-1})$ ▷ x-minimization
  $y_k \leftarrow \arg\min_y L_\rho(x_k, y, \nu_{k-1})$ ▷ y-minimization
  $\nu_k \leftarrow \nu_{k-1} + \rho(Ax_k + By_k - b)$ ▷ dual update
  $k \leftarrow k + 1$
end while

Recall the generalized inequalities, we will formulate an SDP problem of minimizing a convex function defined on the space of $n \times n$ matrices over a set of positive semidefinite cone S_+^n in the form of problem(1.37). The (augmented) Lagrangian can be defined using inner product and norm inducted by this product. This generalization is especially useful in our problem. It is solved in Chapter 3.

Now we proceed to graphical models theory, ADMM comes back later, in chapter regarding precision matrix estimation.

Chapter 2

Gaussian graphical models theory

The material in this part comes mainly from [Has+15; HTF09; KF09] (units regarding graphical models), nonetheless the first chapter of [Die17] was used for the graph theory background.

A graph consists of a set of vertices (nodes), along with a set of edges joining some pairs of the vertices. In graphical models, each vertex represents a random variable, and the graph gives a visual way of understanding the joint distribution of the entire set of random variables. We restrict our discussion to undirected graphical models, also known as Markov random fields or Markov networks. In these graphs, the absence of an edge between two vertices has a special meaning: the corresponding random variables are conditionally independent, given the other variables.

Usually, the structure of the graph is not known and the problem of its estimation is known as graphical model selection. We focus mainly on the sparse models and as sparse graphs have a relatively small number of edges and a regularization (SLOPE or Lasso) is especially useful in the problem of graph estimation.

2.1 Graph theory

2.1.1 Basics

Definition 2.1.1 (Graph). A *graph* is a pair $G = (V, E)$ of sets such that $E \subset [V]^2$, thus, the elements of E are 2-element subsets of V. To avoid notational ambiguities, we shall always assume that $V \cap E = \emptyset$. The elements of V are called *vertices* of G, and those of E are called *edges* of G. The vertex set of a graph G is denoted by $V(G)$ and its edge set by $E(G)$.

Definition 2.1.2 (Adjacency). Two vertices v_1 and v_2 are called *adjacent* if there is an edge joining them, that is $\{v_1, v_1\} \in E(G)$. The set of all the edges in E at a vertex v, that is adjacent to v, is denoted by $E(v)$.

In order to simplify notation we write $v \in G$ or $e \in G$, instead of $v \in V(G)$ and $e \in E(G)$. A pair $\{v_1, v_2\}$ is usually written simply as v_1v_2 or (v_1, v_2), although the last notation is sometimes used in case of directional graphs, where the direction of the edge matters. We only consider unidirectional graphs, which means $(v_1, v_2) \equiv (v_2, v_1)$.

The usual way to picture a graph is by drawing a dot for each vertex and joining two of these dots by a line if the corresponding two vertices form an edge.

Definition 2.1.3 (Graph order, size). For a graph G, the number of its vertices, denoted by $|V|$, is called an *order* of the graph G, and the number of edges, denoted by $|E|$, is called a *size*

Definition 2.1.4 (Connected graph). A graph is *connected* if for every $u, v \in G$ there is a path (i.e. there exist a connection) between u and v.

2.1.2 Subgraphs

Definition 2.1.5 (Subgraph, Supergraph). If $V \subset V'$ and $E \subset E'$, then G is a *subgraph* of G' (and G' is a *supergraph* of G), written as $G \subset G'$.

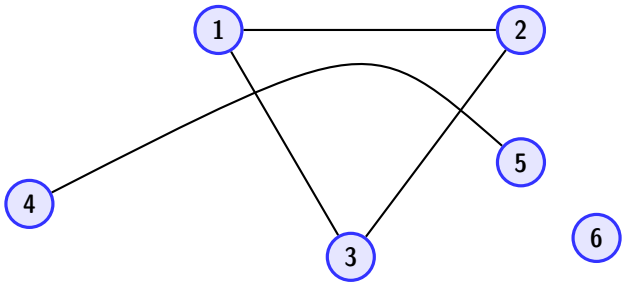


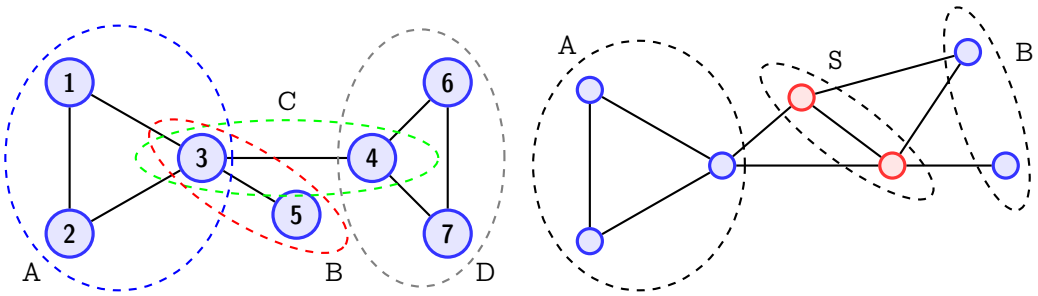
Figure 2.1 – The simple graph G with $V(G) = \{1, 2, 3, 4, 5, 6\}$ and $E(G) = \{12, 23, 31, 45\}$.

When we call a graph minimal or maximal with some property, we are referring to the subgraph relation. When we speak of minimal or maximal sets of vertices or edges, the reference is simply to set inclusion.

Definition 2.1.6 (Graph clique). A *graph clique* $C \subset V$ is a fully connected subset of the vertex set. That means $(s, t) \in E$ for all $s, t \in C$. A clique is said to be maximal if it is not strictly contained in any other clique. We use $\mathfrak{C}_G$ (or just $\mathfrak{C}$) to denote the set of all cliques in the underlying graph G.

Of course single vertex $\{s\}$ is a clique, but it is not maximal unless s is an isolated vertex, what means it participates in no edges.

Definition 2.1.7 (Cut set). *Cut set* S is a set of vertices of underlying graph G which removal broke G into two subcomponents G_1 and G_2.



(a) Illustration of cliques in a graph: Sets A and D are 3-cliques, B and C and 2-cliques, commonly called edges.

(b) Illustration of a vertex cut set S: with vertices in S removed, the graph is broken into two subcomponents A and B.

Figure 2.2 – Cliques and a cut set.

2.2 Factorization and Markov properties

As we introduced some basic concepts of the graph theory we move on to an idea of bonding random variables with graphs. The main concept uses cliques structure of the graph and constrains them with the probability distribution of the random vector indexed by graph vertices.

2.2.1 Factorization property

In this subsection, let $G = (V, E)$ be a graph with a vertex set $V = 1, 2, \ldots, p$ and $\mathfrak{C}$ be its clique set. Let $\mathbb{X} = (X_1, \ldots, X_p)$ be a random vector defined on a probability space $(\Omega, \mathcal{F}, \mathbb{P})$, indexed by the graph nodes.

Definition 2.2.1 (Compatibility function). Let $C \in \mathfrak{C}$ be a clique of the graph G and let $\mathbb{X}_C$ be a subvector of the vector $\mathbb{X}$ indexed by the elements of the clique C, that is $\mathbb{X}_C = (X_s, s \in C)$.

A real-valued function ψ_C of the vector $\mathbb{X}_C$ taking positive real values is called a *compatibility function.*

Given a collection of such compatibility functions, we say that probability distribution $\mathbb{P}$ *factorizes over* G if it has decomposition

$$\mathbb{P}(x_1, \ldots, x_n) = \frac{1}{Z} \prod_{C \in \mathfrak{C}} \psi_C(\mathbf{x}_C), \tag{2.1}$$

where Z is the normalizing constant, known as the *partition function.* It is given by

$$Z = \sum_{\mathbf{x}} \prod_{C \in \mathfrak{C}} \psi_C(\mathbf{x}_C), \tag{2.2}$$

where the sum goes over all possible realizations of $\mathbb{X}$. It ensures that $\mathbb{P}$ is properly normalized. This representation of $\mathbb{P}(x_1, \ldots, x_n)$ is called *Gibbs distribution.*

Going back to fig. 2.2a, any probability function that factorizes over that graph must have the form

$$\mathbb{P}(x_1, \ldots, x_7) = \frac{1}{Z}\psi_{123}(x_1, x_2, x_3)\psi_{34}(x_3, x_4)\psi_5(x_3, x_5)\psi_{467}(x_4, x_6, x_7), \tag{2.3}$$

for some compatibility functions $\{\psi_{123}, \psi_{34}, \psi_{35}, \psi_{467}\}$.

A factorization presented above can lead to solid savings in storage and computation if the clique sizes are not too large. Imagine X_i is a binary random variable, then the distribution of the vector $\mathbb{X} \in \{-1, 1\}^p$ over a graph requires specifying $2^p - 1$ nonnegative numbers. On the other hand, for clique factorization, the number of degrees of freedom is at most $|\mathfrak{C}| \cdot 2^{\max_{C \in \mathfrak{C}} |C|}$. It clear, that for clique factorization the complexity grows exponentially in the maximum clique size $\max_{C \in \mathfrak{C}} |C|$, but only linearly in the number of cliques $|\mathfrak{C}|$. In practice, this factorization could lead to substantial gains.

2.2.2 Markov property

The alternative way how the graph structure can be used to constrain the distribution of $\mathbb{X}$ bases on graph cuts sets (recall fig. 2.2b). It explains why graphical models on undirected graphs are also called Random Markov Fields.

In particular, consider a cut set S and let introduce a symbol $\perp\!\!\!\perp$ to denote the relation *is conditionally independent of.* With this notation, we say that the random vector $\mathbb{X}$ is Markov with respect to G if

$$\mathbb{X}_A \perp\!\!\!\perp \mathbb{X}_B \mid \mathbb{X}_S \qquad \text{for all cut sets } S \subset V, \tag{2.4}$$

where $\mathbb{X}_A$ denotes the subvector indexed by the subgraph A. Markov chains theory provide a illustration of this property. Let G be a path with edge set $E = \{(1, 2), (2, 3), \ldots, (p-1, p)\}$. Any single vertex $S \in \{2, 3, \ldots, p-1\}$ forms a cut set, separating G into the past $P = \{1, 2, \ldots, s-1\}$ and the future $F = \{s+1, 2, \ldots, p\}$. For these cut sets, the Markov property simply translates to the fact that future $\mathbb{X}_F$ is conditionally independent of the past $\mathbb{X}_P$. For more complicated graphs cut sets will be much more complex, also, the properties of conditional independence will be more interesting.

2.2.3 Equivalence of Factorization and Markov properties

The Hammersley-Clifford theorem says that for any strictly positive distribution, two above characterizations are equivalent. This fact states that the distribution of $\mathbb{X}$ factorizes according to the graph G if and only if the random vector $\mathbb{X}$ is Markov with the respect to the graph G.

The Hammersley-Clifford theorem was first announced in the unpublished note of Hammersley and Clifford (1971). Independent proofs were given by Besag [Bes74] and Grimmett [Gri73], the latter proof using the Moebius inversion formula. This remarkable theorem plays a significant role in graphical models theory, but its proof requires a lot of preliminaries we decided to not include it.

2.3 Gaussian graphical models

Generally, class of graphical models is quite broad, but we focus only on Gaussian graphical models, which uses an undirected graph to model dependencies between components in a multivariate normal distribution (MVN). The Gaussian distribution is almost always used for graphical models with all variables continuous, because of its convenient analytical properties.

2.3.1 Canonical parameters

Let $\mathbb{X} \sim \mathcal{N}(\mu, \mathbf{\Sigma})$ be a Gaussian random vector in $\mathbb{R}^p$ with the mean vector $\mu \in \mathbb{R}^p$ and the covariance matrix $\mathbf{\Sigma}$ with a density

$$\mathbb{P}_{\mu,\mathbf{\Sigma}}(x) = \left(\sqrt{\det[2\pi\mathbf{\Sigma}]}\right)^{-1} \exp\left(-\frac{1}{2}(x-\mu)^T\mathbf{\Sigma}^{-1}(x-\mu)\right).$$

If we view the multivariate Gaussian as particular type of exponential family, then μ and $\mathbf{\Sigma}$ are known as the *mean parameters* of the family, for more information see [HMC13], section 7.5. In order to introduce proper model of our interest we need to reformulate MVN into so-called canonical parameterization.

Any nondegenerated (i.e.$\mathbf{\Sigma}$ is strictly positive definite) multivariate normal distribution $\mathcal{N}(\mu, \mathbf{\Sigma})$ can reparametrized into canonical parameters $\gamma \in \mathbb{R}^p$, $\mathbf{\Theta} \in S_+^p$ of the form

$$\gamma = \mathbf{\Sigma}^{-1}\mu \quad \text{and} \quad \mathbf{\Theta} = \mathbf{\Sigma}^{-1}. \tag{2.5}$$

The canonical representation is derived as follows

$$\begin{aligned}
\mathbb{P}_{\mu,\mathbf{\Sigma}}(x) &= \left(\sqrt{\det[2\pi\mathbf{\Sigma}]}\right)^{-1} \exp\left\{(-\frac{1}{2}(x-\mu)^T\mathbf{\Sigma}^{-1}(x-\mu)\right\} \\
&= \left(\sqrt{\det[(2\pi\mathbf{\Sigma})^{-1}]}\right) \exp\left\{-\frac{1}{2}x^T\mathbf{\Sigma}^{-1}x + x^T\mathbf{\Sigma}^{-1}\mu - \frac{1}{2}\mu^T\mathbf{\Sigma}^{-1}\mu\right\} \\
&= \left(\sqrt{\det[(2\pi)^{-1}\,\mathbf{\Theta}]}\right)^{-1} \exp\left\{-\frac{1}{2}x^T\,\mathbf{\Theta}\,x + x^T\gamma - \frac{1}{2}\gamma^T\,\mathbf{\Theta}^{-1}\,\gamma\right\} \\
&= \exp\left\{-\frac{1}{2}x^T\,\mathbf{\Theta}\,x + x^T\gamma - \frac{1}{2}\left(\det[(2\pi)^{-1}\,\mathbf{\Theta}] + \gamma^T\,\mathbf{\Theta}^{-1}\,\gamma\right)\right\} \\
&= \exp\left\{-\frac{1}{2}x^T\,\mathbf{\Theta}\,x + x^T\gamma - A(\gamma,\mathbf{\Theta})\right\} \\
&= \mathbb{P}_{\gamma,\mathbf{\Theta}}(x)
\end{aligned} \tag{2.6}$$

where $A(\gamma,\mathbf{\Theta}) = -\frac{1}{2}\left(\det[(2\pi)^{-1}\,\mathbf{\Theta}] + \gamma^T\,\mathbf{\Theta}^{-1}\,\gamma\right)$. By using $\mathbb{P}_{\gamma,\mathbf{\Theta}}(x)$ notation we emphasize that we are thinking about the canonical formulation of MVN.

The inverse covariance matrix $\mathbf{\Theta}$ is called a *precision* or *concentration* matrix.

Now, we write the last line from derivation (2.6) in the entrywise form

$$\mathbb{P}_{\gamma,\mathbf{\Theta}}(x) = \exp\left\{\sum_{s=1}^{p}\gamma_s x_s - \frac{1}{2}\sum_{s,t=1}^{p}\theta_{st}x_s x_t - A(\gamma,\mathbf{\Theta})\right\}. \tag{2.7}$$

The eq. (2.7) is especially convenient in graphical model theory discussed in the next section.

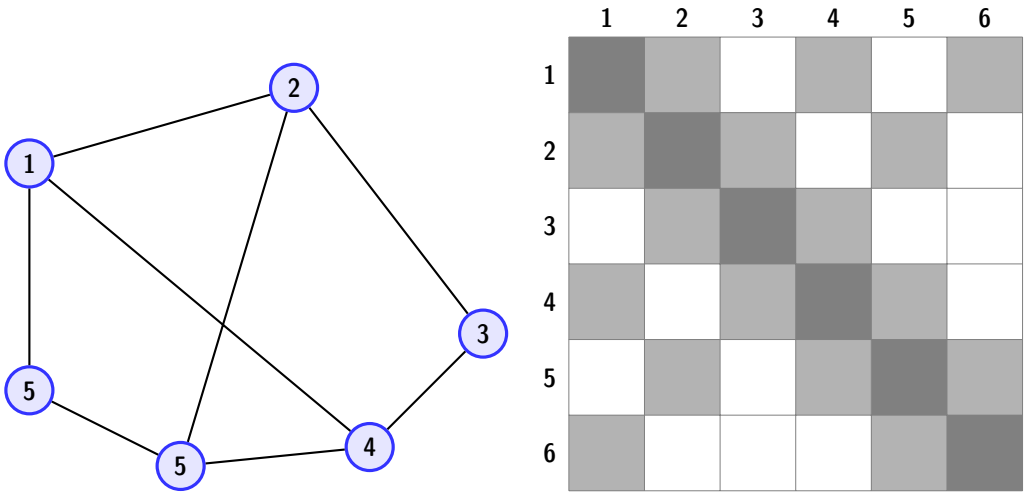
2.3.2 Model

The (2.7) allow us to discuss the factorization properties in the terms of the sparsity pattern of the precision matrix $\mathbf{\Theta}$. Especially, if $\mathbb{X}$ factorizes according to some graph G, then $\theta_{st} = 0$ for any pair $(s,t) \notin E$, which sets up the correspondence between the zero pattern of the matrix $\mathbf{\Theta}$ and pattern of the underlying graph. This comes from the fact the precision matrix $\mathbf{\Theta}$ contains information about the partial covariances between the variables, that is, the covariances between

pairs s and t, conditioned on all other variables. In particular, if the $\theta_{st} = 0$, then variables s and t are conditionally independent, given the other variables.

The proofs of this properties can be found in [Lau96], chapter 5. We do not cover them here, as they are cumbersome and need many preliminaries.

We now turn to the problem of graph selection. The problem itself is simply stated: suppose that we are given a collection $\mathbf{X} = \{x_1, \ldots, x_N\}$ of samples from a graphical model (or data we want to "fit into" graphical model), but the underlying graph structure is unknown. How to use the data to select the correct graph with high probability? There are many approaches, such as likelihood, Bayesian criteria, stability methods. We present our approach based on SLOPE in Chapter 3.



(a) The undirected graph G on six vertices.

(b) The associated sparsity pattern of the precision matrix Θ. White squares correspond to zero entries.

Figure 2.3 – The graph and corresponding precision matrix.

Chapter 3

The problem of graph selection

Now we turn to the main aspect of this thesis, graph selection. Consider a simple problem: suppose there is given a collection $\mathfrak{X} = \{x_1, \dots, x_N\}$ of samples from a graphical model with an unknown graph structure. How to select the correct graph structure with high probability? We discuss the use of likelihood-based methods combined with ℓ_1-regularization for this purpose. In Gaussian case, it leads to manageable methods for model selection. We present two different approaches in this chapter and compare them later.

3.1 Global Likelihoods for Gaussian Models

As stated earlier, we only consider Gaussian graphical models, where the problem of model selection is also known as covariance selection. Our main objective is to estimate the graph structure, so we assume the distribution has zero mean, thus we need only consider the symmetric precision matrix $\Theta \in \mathbb{R}^{p\times p}$, a vector γ is out of scope.

Let X be a sample from zero-mean multivariate Gaussian with precision matrix Θ. It's log-likelihood $\mathbb{L}(\Theta, X)$ takes the form

$$\begin{aligned}
\mathbb{L}(\Theta, X) &= \frac{1}{N}\sum_{i=1}^{N} \log \mathbb{P}_{\Theta}(x_i) \\
&= \frac{1}{N}\sum_{i=1}^{N} -\frac{1}{2} x_i^T \Theta\, x_i - A(\Theta) \\
&= \frac{1}{N}\sum_{i=1}^{N} \frac{1}{2}\log\det[\Theta/2\pi] - \frac{1}{2} x_i^T \Theta\, x_i \\
&= \frac{1}{2N}\sum_{i=1}^{N} \log\det\Theta - N\log 2\pi - x_i^T \Theta\, x_i \\
&= \frac{1}{2}\log\det\Theta - \frac{N}{2}\log 2\pi - \frac{1}{2N}\sum_{i=1}^{N} \operatorname{tr}\left(-x_i^T x_i\, \Theta\right) \\
&= \frac{1}{2}\log\det\Theta - \frac{N}{2}\log 2\pi - \frac{1}{2N}\sum_{i=1}^{N} -\operatorname{tr}\left(x_i x_i^T\, \Theta\right) \\
&= \frac{1}{2}\log\det\Theta - \frac{N}{2}\log 2\pi - \frac{1}{2}\operatorname{tr}\left(S\,\Theta\right),
\end{aligned} \tag{3.1}$$

where S is an empirical covariance matrix given by $\frac{1}{N}\sum_{i=1}^{N} x_i x_i^T$ and the log-determinant function is defined on the space of symmetric matrices as

$$\log\det A = \begin{cases} \sum_{i=1}^{p} \log(\lambda_i) & \text{if } A \succ 0 \\ -\infty & \text{otherwise,} \end{cases}$$

where λ_i denotes i-th eigenvalue of $\boldsymbol{A}$.

In derivation 3.1 we used the fact that the trace operator is invariant under cyclic permutations and $x_i^T \boldsymbol{\Theta} x_i$ is a scalar, so we can write $x_i^T \boldsymbol{\Theta} x_i = \operatorname{tr}\left(x_i^T \boldsymbol{\Theta} x_i\right)$.

Up to a constant, the result of log-likelihood function derived in (3.1) is equal to

$$\mathbb{L}(\boldsymbol{\Theta}, \boldsymbol{X}) = \log\det \boldsymbol{\Theta} - \operatorname{tr}(\boldsymbol{S}\,\boldsymbol{\Theta})\,. \tag{3.2}$$

As the objective function (3.2) is strictly concave (see [BV09], chapter 3) the maximum, when it is achieved, must be unique, and defines the precision matrix MLE $\widehat{\boldsymbol{\Theta}}_{ML}$.

Classical theory says MLE converge to the true parameter $\boldsymbol{\Theta}$ as N goes to infinity. Thus generally we could use a thresholded version of $\widehat{\boldsymbol{\Theta}}_{ML}$ to specify an edge set and find the best Gaussian graphical model.

However, especially in practice, the number of nodes p may be comparable to, or larger than, the sample size N. In such cases using MLE does not make any sense. Indeed, the empirical correlation matrix $\boldsymbol{S}$ must be rank-degenerate whenever $N < p$, which implies that MLE fails to exist. It comes from the fact that

$$\widehat{\boldsymbol{\Theta}}_{ML} \in \underset{\Theta \in \mathcal{S}_+^p}{\arg\max}\{\log\det \boldsymbol{\Theta} - \operatorname{tr}(\boldsymbol{S}\,\boldsymbol{\Theta})\}$$

when the maximum is attained. Hence the fact that maximum is attained we get

$$\boldsymbol{S}^{-1} = \widehat{\boldsymbol{\Theta}}.$$

When $N < p$, then $\operatorname{rank}(\boldsymbol{X}) \leqslant \min(N, p) = N$. As the sample covariance matrix rank is smaller than N ($\boldsymbol{S}$ is the function of $\boldsymbol{X}$), it is singular, so we get the contradiction, because $\boldsymbol{S}^{-1}$ does not exist.

Hence we must consider suitability constrained or regularized forms of MLE. Furthermore, irrespective of the sample size, we may be interested in constraining the estimated precision matrix to be sparse, what makes it easier to analyze. Moreover, in some applications, we want to control some kinds of errors, like a number of false discoveries. Then regularization methods with properly chosen penalty parameter are the solution for this needs.

3.2 Regularization

The idea of regularization is simple. If we are seeking GGMs based on relatively sparse graphs, then it is desirable to control the number of edges, which can be measured by ℓ_0-based quantity

$$\rho_0(\boldsymbol{\Theta}) = \sum_{s \neq t} \mathbb{I}[\theta_{st} \neq 0].$$

Note that $\rho_0(\boldsymbol{\Theta}) = 2|E(G)|$ for a given graph G, so we could consider the optimization problem

$$\widehat{\Theta} \in \underset{\substack{\Theta \in \mathcal{S}_+^p \\ \rho_0(\boldsymbol{\Theta}) \leqslant k}}{\arg\max}\{\log\det \boldsymbol{\Theta} - \operatorname{tr}(\boldsymbol{S}\,\boldsymbol{\Theta})\} \tag{3.3}$$

Unfortunately, the ℓ_0-based constrained defines a highly nonconvex constraint set formed as the union over all $\binom{p}{2}$ possible subsets of k edges. This makes the problem hard to solve without brute-force methods. Natural relaxation of this constraint are ℓ_1 based methods: graphical Lasso and graphical SLOPE.

3.3 Graphical Lasso

3.3.1 Graphical Lasso problem

Lasso (Least absolute shrinkage and selection operator) is mainly known as a regression regularization method, used for variable selection. We introduce *graphical Lasso*, a variation used for sparse graphical model selection.

Convex relaxation of (3.3) leads to the problem of maximizing the penalized log-likelihood of the form

$$\mathbb{L}_\lambda(\boldsymbol{\Theta}, \mathbf{X}) = \log\det\boldsymbol{\Theta} - \operatorname{tr}(\mathbf{S}\boldsymbol{\Theta}) - \lambda\|\boldsymbol{\Theta}\|_1. \tag{3.4}$$

where $\|\cdot\|_1$ states for entrywise off-diagonal ℓ_1-norm $\|A\|_1 = \sum_{i\neq j}|a_{ij}|$ (variation with penalized diagonal is sometimes considered).

Thus the problem we need to solve is given by

$$\widehat{\boldsymbol{\Theta}} \in \underset{\Theta\in\mathcal{S}_+^p}{\arg\max}\{\log\det\boldsymbol{\Theta} - \operatorname{tr}(\mathbf{S}\boldsymbol{\Theta}) - \lambda\|\boldsymbol{\Theta}\|_1\}. \tag{3.5}$$

The choice of the parameter λ is crucial for this method, we face this method in the last section of thic chapter.

3.3.2 Algorithms

We present two approaches for solving problem (3.5) which we compare in the next chapter.

Graphical Lasso algorithm

As (3.5) can be formulated as a convex problem, it is solvable with convex optimization methods. In [VBW98] generic interior point methods were proposed, but this is not an efficient solution for large problems. First-order block coordinate descent approaches, introduced by in [BEd08] and refined in [FHT08] are better for this type of problem. The latter authors call this approach the *graphical lasso algorithm*. We use it in our estimations, as it is easily available as an R package **glasso**. We decide to compare it with ADMM algorithm applied to the gLasso problem, results are presented in the next chapter.

We do not derive and discuss graphical Lasso algorithm, detailed analysis and proofs can be found in [FHT08]. We just present the algorithm in the same way as it is presented in the cited paper.

Consider the matrices partitioning into one column versus the rest, for convenience we pick the last column:

$$\boldsymbol{\Theta} = \begin{bmatrix}\boldsymbol{\Theta}_{11} & \theta_{12}\\ \theta_{12}^T & \theta_{22}\end{bmatrix}, \quad \mathbf{S} = \begin{bmatrix}\mathbf{S}_{11} & \mathbf{s}_{12}\\ \mathbf{s}_{12}^T & S_{22}\end{bmatrix} \quad \text{etc.} \tag{3.6}$$

The partitioning (3.6) is used in graphical Lasso algorithm below.

Algorithm 2 Graphical Lasso

$\boldsymbol{W} \leftarrow \boldsymbol{S}$ ▷ initialize
while convergence criterion is not met **do**
  for $i = 1, 2, \ldots, p$ **do**
    Partition the matrices $\boldsymbol{W}$ and $\boldsymbol{S}$ into i-th row and column as in (3.6)
    Solve $\widehat{\beta} = \min_\beta\left\{(1/2)\|\boldsymbol{W}_{11}^{1/2}\beta - b\|^2 + \lambda\|\beta\|_1\right\}$, where $b = W_{11}^{-1/2}s_{12}$
    Update the corresponding row and column of $\boldsymbol{W}$ using $w_{12} = \boldsymbol{W}_{11}\widehat{\beta}$
  end for
end while

If the diagonal elements are not left out of the penalty in (3.4), the first step in the algorithm changes for $\boldsymbol{W} \leftarrow \boldsymbol{S} + \mathbf{1}\lambda$, the rest of the algorithm remains the same as before.

ADMM for Graphical Lasso

Although we could use gLasso algorithm during simulations, we decided to derive its ADMM version. Firstly, it might be faster, secondly, it is very similar to ADMM for graphical SLOPE, introduced later.

For convenience and being consistent with the section about ADMM, we use different notation in this section. We use:

- X to denote $\boldsymbol{\Theta}$,
- S to denote $\mathbf{S}$,
- N, instead of ν, to denote the dual variable.
- X_k to denote variable X state during k-th iteration, unless stated differently.

Now we derive the exact ADMM algorithm for the graphical Lasso problem (3.5). The introduction to ADMM is given in the first chapter, so we do not cover its background here.

As the original log-likelihood function (3.4) is strictly concave we need to consider problem given by its additive inverse

$$\begin{aligned} &\text{minimize} && -\log\det X + \operatorname{tr}(XS) + \lambda\|X\|_1 \\ &\text{subject to} && X \in S_+^p\,. \end{aligned} \tag{3.7}$$

Recall (1.37), where constrained convex optimization problem was in the same form as (3.7). As the objective function is strictly convex, the variable is constrained to positive semidefinite cone, we can reformulate the initial problem in the same manner

$$\begin{aligned} &\text{minimize} && -\log\det X + \operatorname{tr}(XS) + \mathbb{I}[X \succeq 0] + \lambda\|Y\|_1 \\ &\text{subject to} && X = Y. \end{aligned} \tag{3.8}$$

Having proper ADMM problem formulation, we write an augmented Lagrangian with penalty parameter $\rho \in \mathbb{R}$ in the inner product form

$$\begin{aligned} &\mathcal{L}_\rho : \mathbb{R}^{p\times p} \times \mathbb{R}^{p\times p} \times \mathbb{R}^{p\times p} \to \mathbb{R} \\ &\mathcal{L}_\rho(X, Y, N) = -\log\det X + \operatorname{tr}(XS) + \mathbb{I}[X \succeq 0] + \lambda\|Y\|_1 + \rho\langle N, X - Y\rangle_F + \frac{\rho}{2}\|X - Y\|_F^2. \end{aligned} \tag{3.9}$$

Note that $\langle\cdot,\cdot\rangle$ is the Frobenius inner product define as $\langle A, B\rangle_F = \sum_{i,j} \overline{a_{ij}} b_{ij} = \operatorname{tr}\left(\overline{A^T}B\right)$ (where the overline denotes the complex conjugate) and $\|\cdot\|_F$ is the Frobenius norm inducted by this inner product defined as $\|A\|_F = \sqrt{\langle A, A\rangle_F} = \sqrt{\sum_{i,j} |a_{ij}|^2}$.

Recall algorithm 1, to solve the optimization problem we need to minimize the augmented Lagrangian as a function of X and Y and then do the dual update in order to conduct algorithm step. The update formula for X_k in ADMM algorithm is given by

$$\begin{aligned} X_k &= \arg\min_X \mathcal{L}_\rho(X, Y_{k-1}, N_{k-1}) \\ &= \arg\min_{X\succeq 0} \left\{ -\log\det X + \operatorname{tr}(XS) + \lambda\|Y_{k-1}\|_1 + \rho\langle N_{k-1}, X - Y_{k-1}\rangle_F + \frac{\rho}{2}\|X - Y_{k-1}\|_F^2 \right\} \\ &= \arg\min_{X\succeq 0} \left\{ -\log\det X + \langle X, S\rangle_F + \lambda\|Y_{k-1}\|_1 + \rho\langle N_{k-1}, X\rangle_F - \rho\langle N_{k-1}, Y_{k-1}\rangle_F + \frac{\rho}{2}\|X - Y_{k-1}\|_F^2 \right\} \\ &= \arg\min_{X\succeq 0} \left\{ -\log\det X + \langle X, S\rangle_F + \rho\langle N_{k-1}, X\rangle_F + \frac{\rho}{2}\left\|X - Y_{k-1} + \left(N_{k-1} + \frac{1}{\rho}S\right)\right\|_F^2 \right. \\ &\qquad \left. - \frac{\rho}{2}\left\|N_{k-1} + \frac{1}{\rho}S\right\|_F^2 - \rho\left\langle X - Y_{k-1}, N_{k-1} + \frac{1}{\rho}S\right\rangle_F \right\} \\ &= \arg\min_{X\succeq 0} \left\{ -\log\det X + \langle X, S\rangle_F + \rho\langle N_{k-1}, X\rangle_F + \frac{\rho}{2}\left\|X + \left(N_{k-1} - Y_{k-1} + \frac{1}{\rho}S\right)\right\|_F^2 \right. \\ &\qquad \left. - \rho\langle X, N_{k-1}\rangle_F - \rho\left\langle X, \frac{1}{\rho}S\right\rangle_F \right\} \\ &= \arg\min_{X\succeq 0} \left\{ -\log\det X + \frac{\rho}{2}\left\|X + \left(N_{k-1} - Y_{k-1} + \frac{1}{\rho}S\right)\right\|_F^2 \right\} \end{aligned}$$

The second equality uses the fact that X and S are real symmetric matrices, the third comes from the (sesqui)linearity of a inner product, the fourth uses the fact that $\|a + b\|^2 = \|a\|^2 + \|b\|^2 + 2\langle a, b\rangle$. Moreover, we dropped the expressions that do not affect minimizing over the variable X.

Now let

$$\tilde{S}_{k-1} = -N_{k-1} + Y_{k-1} - \frac{1}{\rho}S,$$

then we have

$$\arg\min_{X \succeq 0} \left\{ -\log\det X + \frac{\rho}{2} \left\| X - \tilde{S} \right\|_F^2 \right\},$$

thus the X gradient of the augmented Lagrangian (3.9) is given by

$$\nabla_X \mathcal{L}_\rho(X, Y_{k-1}, N_{k-1}) = -X^{-1} + \rho X - \rho\tilde{S}_{k-1}.$$

As the augmented Lagrangian is convex, using optimality conditions we get

$$\nabla_X \mathcal{L}_\rho(X^*, Y_{k-1}, N_{k-1}) = 0 \tag{3.10}$$

$$-(X^{*-1}) + \rho X^* - \rho\tilde{S}_{k-1} = 0, \tag{3.11}$$

that is gradient vanish for some $X^* \succeq 0$. Thus X^* is the solution for the X-update. Rewriting equation (3.11) as

$$-(X^*)^{-1} + \rho X^* = \rho\tilde{S}_{k-1},$$

we can find a matrix that meets this condition (minimizes optimization program). At first, lets take the eigenvalue decomposition of right side

$$\rho\tilde{S}_{k-1} = Q\Lambda Q^T,$$

where $\Lambda = \operatorname{diag}\{l_i\}$ and $QQ^T = Q^TQ = \mathbf{1}$. By multiplying right and left side by Q and Q^T respectively, we obtain

$$-(\tilde{X}^*)^{-1} + \rho\tilde{X}^* = \Lambda,$$

where $\tilde{X}^* = Q^T X^* Q$.

Now, we construct a solution to this equation, that is we find positive numbers $\tilde{x}^*_{ii}$ (here x^*_{ii} denotes elements of the matrix X^*) that satisfy

$$\frac{1}{\rho}(\tilde{x}^*_{ii})^2 - l_{ii}\tilde{x}^*_{ii} - 1 = 0.$$

It is obvious that

$$\tilde{x}_{ii} = \frac{l_i + \sqrt{l_i^2 + \frac{4}{\rho}}}{2}.$$

Thus X^* is given by $X^* = Q^T\tilde{X}^*Q$. All diagonals are positive since $\rho > 0$. The computational cost of this operation is that of finding eigenvalue decomposition of matrix X^*.

Define $\mathcal{F}_\rho(\Lambda)$ as

$$\mathcal{F}_\rho(\Lambda) = \frac{1}{2}\operatorname{diag}\left\{ l_i + \sqrt{l_i^2 + \frac{4}{\rho}} \right\}$$

Since that

$$X^* = Q\mathcal{F}_\rho = Q^T\tilde{X}^*Q = Q^T\mathcal{F}_\rho Q = \mathcal{F}_\rho(\tilde{S}_{k-1})$$

we obtain a formula for updating X_k in each step.

Now derive a formula for a Y_k-updating

$$\begin{aligned}
Y_k &= \arg\min_Y \mathcal{L}_\rho(X_k, Y, N_{k-1}) \\
&= \arg\min_Y \left\{ -\log\det X_k + \operatorname{tr}(X_k S) + \lambda\|Y\|_1 + \rho\langle N_{k-1}, X_k - Y\rangle_F + \frac{\rho}{2}\|X_k - Y\|_F^2 \right\} \\
&= \arg\min_Y \left\{ \lambda\|Y\|_1 + \frac{\rho}{2}\|Y - (X_k + N_{k-1})\|_F^2 \right\} \\
&= \mathrm{ST}_{(\lambda/\rho)}(X_k + N_{k-1}),
\end{aligned}$$

where ST_κ is soft thresholding function introduced earlier, see (1.36).

As we have the formula for X and Y update, we can write the proper algorithm for gLasso

Algorithm 3 Alternative direction method of multipliers for gLasso

$Y_0 \leftarrow \tilde{Y}$, $N_0 \leftarrow \tilde{N}$, $k \leftarrow 1$ ▷ initialize (loosely)
$\mu \leftarrow \tilde{\mu} > 0$ ▷ initialize
while convergence criterion is not meet **do**
  $X_k \leftarrow \mathcal{F}_\rho(N_{k-1} + Y_{k-1} - \frac{1}{\rho}S)$ ▷ x-minimization
  $Y_k \leftarrow ST_{(\lambda/\rho)}(X_k + N_{k-1})$ ▷ y-minimization
  $N_k \leftarrow N_{k-1} + \rho(X_k - Y_k)$ ▷ dual update
  $k \leftarrow k + 1$
end while

3.4 Graphical SLOPE

3.4.1 Motivation

In [Bog+15] Bogdan et. al. presented a novel approach to regularization. The concept of *Sorted ℓ_1 penalized regression* [SLOPE] is especially useful in sparse models. Instead of using single λ value for the variable selection procedure authors propose to order the coefficients in decreasing order and tie them with decreasing series of coefficients. They proved that SLOPE could control a fraction of false discoveries in orthogonal designs, they derived the algorithm for solving the optimization problem, and they compared SLOPE with Lasso. More information and background can be found in the cited paper.

The SLOPE is using OL1 instead of L1 norm for coefficient selection. For linear model $y = X\beta + z$, the problem called *ordered lasso* is stated as

$$\min_b \frac{1}{2}\|y - Xb\|_2^2 + \lambda_1|b_{(1)}| + \lambda_2|b_{(2)}| + \ldots + \lambda_p|b_{(p)}|,$$

where

$$\lambda_1 \geqslant \lambda_2 \geqslant \ldots \geqslant \lambda_p$$
$$|b|_{(1)} \geqslant |b|_{(2)} \geqslant \ldots \geqslant |b|_{(p)}.$$

By $x_{(i)}$ we denote the i-th decreasing order statistic.

Briefly, the proposed regularizer penalizes coefficients according to their rank: the higher the rank is, the larger the penalty. The proposed approach is somehow similar to the *Benjamini-Hochberg procedure* [BHq] [BH95], that compares the value of a test statistic to a critical threshold that depends on its rank in the family. Both methods are oriented on controlling a number of false discoveries.

We apply the idea of sorted ℓ_1 norm to the problem of graphical model selection. This novel approach suggested by Małgorzata Bogdan and we did some research about the capabilities of this method.

3.4.2 OL1 regularizer

The ordered ℓ_1 regularizer (known also as OL1, OWL, or OSCAR) for $\beta \in \mathbb{R}^p$ and $\lambda \in \mathbb{R}^p, \lambda_1 \geqslant \ldots \geqslant \lambda_p$ is defined by

$$J_\lambda(\beta) = \sum_{i=1}^{p} \lambda_i|\beta|_{(i)}.$$

In order to use J_λ in graphical model estimation procedure we need to reformulate it for the $p \times p$ matrix space.

Let $A \in \mathbb{R}^{p\times p}$ and $\lambda \in \mathbb{R}^{p^2}, \lambda_1 \geqslant \ldots \geqslant \lambda_{p^2}$, then

$$J_\lambda(A) = \sum_i \lambda_i|a|_{(i)}.$$

The notation might be confusing, but by an abstract indexing over (i) we point out that we go through all elements of the matrix A in decreasing order. The J_λ operator can be used differently in many different settings, for example, symmetric matrices need only penalization of the upper (lower) triangle and copying the result to the lower (upper) triangle.

From now, in our paper we use only the version of J_λ appropriate for the precision matrix $\boldsymbol{\Theta} \in S_+^p$ given by

$$J_\lambda(\boldsymbol{\Theta}) = \sum_i \lambda_i |\theta|_{(i)},$$

where θ_i denote the i-th largest component of upper triangle (without diagonal) of $\boldsymbol{\Theta}$.

As in graphical Lasso, variation with penalized diagonal might be considered.

3.4.3 Graphical SLOPE problem

Now recall the convex relaxation of the ℓ_0 penalized log-likelihood (3.4). We want to use OL1 instead of ordinary ℓ_1 norm, thus want to maximize

$$\mathbb{L}_\lambda(\boldsymbol{\Theta}, \mathbf{X}) = \log\det \boldsymbol{\Theta} - \operatorname{tr}(\boldsymbol{S}\,\boldsymbol{\Theta}) - J_\lambda(\boldsymbol{\Theta}). \tag{3.12}$$

The optimization problem is given by

$$\widehat{\boldsymbol{\Theta}} \in \underset{\Theta \in S_+^p}{\arg\max} \{\log\det \boldsymbol{\Theta} - \operatorname{tr}(\boldsymbol{S}\,\boldsymbol{\Theta}) - J_\lambda(\boldsymbol{\Theta})\}, \tag{3.13}$$

The great problem with the OL1 regularization is the proper choice of lambda series. We face this in the next section, together with the parameter choice for gLasso.

3.4.4 ADMM for Graphical SLOPE

Once again we use different notation in this section. Recall that we use:

- X to denote $\boldsymbol{\Theta}$,
- S to denote $\mathbf{S}$,
- N, instead of ν, to denote the dual variable.

As SLOPE is the modification of Lasso, the formula for X_k-update is the same as in (3.8) because it is independent of the regularization function.

A formula for Y_k is different. We have

$$\begin{aligned} Y_k &= \underset{Y}{\arg\min}\, \mathcal{L}_\rho(X_k, Y, N_{k-1}) \\ &= \underset{Y}{\arg\min} \left\{ -\log\det X_k + \operatorname{tr}(X_k S) + J_\lambda(Y) + \mu\langle N_{k-1}, X_k - Y\rangle_F + \frac{\rho}{2}\|X_k - Y\|_F^2 \right\} \\ &= \underset{Y}{\arg\min} \left\{ J_\lambda(Y) + \frac{\rho}{2}\|Y - (X_k + N_{k-1})\|_F^2 \right\} \end{aligned}$$

Now recall the proximity operator (1.34) defined as $\mathbf{prox}_{f,\rho}(\xi)$, the last line of Y-update formula is equal to

$$\underset{Y}{\arg\min} \left\{ J_\lambda(Y) + \frac{\rho}{2}\|Y - (X_k + N_{k-1})\|_F^2 \right\} = \mathbf{prox}_{J_\lambda,\rho}(X_k + N_{k-1}). \tag{3.14}$$

As stated in the introduction to this section, the algorithm for solving a SLOPE proximal operator was derived and presented in [Bog+15]. We do not include it in this paper, as it needs many preliminaries to understand its concept.

3.5 Parameter choice

As stated before, the choice of the parameter λ is crucial in both methods. As our main objective is to control the number of false discoveries (which is useful, e.g., in medical studies), the choice is based on multiple testing theory. Considerations presented in this section are based and suggested by Piotr Sobczyk and they will be carefully described and presented in his PhD thesis [Sob18]. We just present a sample if his work. All theoretical results, proofs etc. will be available there.

Algorithm 4 Alternative direction method of multipliers for gSLOPE

$Y_0 \leftarrow \tilde{Y}$, $N_0 \leftarrow \tilde{N}$, $k \leftarrow 1$ ▷ initialize (loosely)
$\mu \leftarrow \tilde{\mu} > 0$ ▷ initialize
while convergence criterion is not meet **do**
  $X_k \leftarrow \mathcal{F}_\rho(N_{k-1} + Y_{k-1} - \frac{1}{\rho}S)$ ▷ x-minimization
  $Y_k \leftarrow \mathbf{prox}_{J_\lambda,\rho}(X_k + N_{k-1})$ ▷ y-minimization
  $N_k \leftarrow N_{k-1} + \rho(X_k - Y_k)$ ▷ dual update
  $k \leftarrow k + 1$
end while

3.5.1 Introduction

Multiple comparisons occur when a statistical analysis involves multiple simultaneous statistical tests, each of which has a potential to produce a discovery. Obviously, the more inferences are made, the more likely wrong discoveries occur. A stated confidence level generally applies only to each test considered individually, but often it is desirable to have a confidence level for the whole family of simultaneous tests.

Sometimes wrong discoveries are extremely expansive; imagine a medication design industry, where false inferences might lead to tons of hours spent on a purposeless research. There are numbers of controlling procedures, which the main objective is to somehow control this kind of discoveries.

In the case of precision matrix estimation we are conducting one test per one entry of covariance matrix, thus this theory is applicable in our problem.

In each problem the lambda choice is based on the following parameters:

- dimensionality of data matrix $\mathbf{X} \in \mathbb{R}^{N \times p}$ (N observations of p dimensional random variable),
- parameter α, which denotes a desired level of an error control,
- sample covariance matrix $\mathbf{S}$.

We also assume our data are scaled and centered.

3.5.2 FWER, FDR, sensitivity, specificity

The errors can be determined with a *confusion matrix* defined as

		Real value (+)	Real value (−)
Test outcome	(+)	True positive	False positive
	(−)	False negative	True negative

Definition 3.5.1 (Familywise error rate). A *family-wise error rate* (FWER) is the probability of making one or more false discoveries, that is,

$$\mathrm{FWER} = \mathbb{P}(\text{type I error}).$$

Definition 3.5.2 (False discovery rate). *False discovery rate* [FDR] is defined as

$$\mathrm{FDR} = \mathbb{E}\left[\frac{\#[\text{False positive}]}{\#[\text{False positive}] + \#[\text{True positive}]}\right].$$

As stated earlier, methods with FDR control are useful in applications, where false discoveries are expansive.

As we consider graphical models, we can treat false discovery in a given component of the underlying graph as positive discovery, that is if procedure finds an edge between two vertices which are not adjacent, but are connected by a longer path, then we do not treat it as a false discovery.

Thus we define so-called *local FDR* for graphs as

$$\text{localFDR} = \mathbb{E}\left[\frac{\#[\text{False positive outside the component}]}{\#[\text{False positive}] + \#[\text{True positive}]}\right].$$

A local FDR converts each component of the real graph into a (minimum) clique.

Definition 3.5.3 (Sensitivity and specificity). *sensitivity* (or *empirical power*, shortly *power*) and *specificity* defined as follows

$$\text{Sensitivity} = \mathbb{E}\left[\frac{\#[\text{True positive}]}{\#[\text{True positive}] + \#[\text{False negative}]}\right],$$
$$\text{Specificity} = \mathbb{E}\left[\frac{\#[\text{True negative}]}{\#[\text{True negative}] + \#[\text{False positive}]}\right].$$

.

Although Sensitivity is not power by a strict definition, we use those notions interchangeably in this paper.

3.5.3 Lambda for gLasso

As said before, the choice of lambda is crucial for the usefulness of the graphical Lasso method. There are many approaches, we use the one proposed by Banerjee et al. in [BEd08]. The main point is the fact that for Gaussian models, it controls the FDR.

The formula proposed in [BEd08] for the parameter λ is given by

$$\lambda^{\text{Banerjee}}(\alpha) = \max_{i<j}(s_{ii}, s_{jj})\frac{\text{qt}_{n-2}(1-\frac{\alpha}{2p^2})}{\sqrt{n-2+\text{qt}^2_{n-2}(1-\frac{\alpha}{2p^2})}}, \tag{3.15}$$

where s_{ii} denotes the variance of variable i, that is i-th diagonal element of the sample covariance matrix $\mathbf{S}$. As our data are scaled and centered, the first term is negligible.

In [BEd08] a following theorem was formulated.

Theorem 3.5.1. *Using* (3.15) *as the penalty parameter in graphical Lasso problem* (3.5)*, for any fixed level* α

$$\mathbb{P}(\text{False Positive Discovery}) \leqslant \alpha,$$

where "False Positive discovery" means there is a nonzero coefficient of the estimated precision matrix, which is zero in the real precision matrix.

The detailed proof is given in appendix B of the cited paper.

3.5.4 Lambda for gSLOPE

Multiple testing procedures

We begin with a presentation of multiple testing procedures which motivated Sobczyk in [Sob18] for his choice of lambda.

Let $H_1, \ldots, H_m$ be a family of hypotheses and $p_1, \ldots, p_m$ their corresponding p-values. Let m be the total number of null hypotheses and m_0 the number of true null hypotheses. Let $\alpha \in [0, 1]$ be a demanded level of error control.

Bonferonni correction Reject the null hypothesis for each $p_i \leqslant \frac{\alpha}{m}$. Bonferonni correction controls a FWER at the level α, that is, FWER $\leqslant \alpha$. This control does not require any assumptions about dependence among the p-values or about how many of the null hypotheses are true.

Holm method

- Start by ordering the p-values in increasing order $p_{(1)}, \ldots, p_{(m)}$ and let the associated hypotheses be $H_{(1)}, \ldots, H_{(m)}$.
- For a given significance level α, let R be the minimal index such that $p_{(k)} > \frac{\alpha}{m+1-k}$.
- Reject the null hypotheses $H_{(1)}, \ldots, H_{(k-1)}$ and do not reject $H_{(k)}, \ldots, H_{(m)}$.
- If $k = 1$ then do not reject any of the null hypotheses and if no such k exist then reject all of the null hypotheses.

The Holm method ensures that FWER is controlled at the level α.

Benjamini-Hochberg procedure

- Start by ordering the p-values in increasing order $p_{(1)}, \ldots, p_{(m)}$ and let the associated hypotheses be $H_{(1)}, \ldots, H_{(m)}$.
- Find the largest k such that $p_{(k)} \leqslant \frac{k}{m}\alpha$.
- Reject the null hypothesis for all $H_{(1)}, \ldots, H_{(k)}$.

The Benjamini–Hochberg procedure (BH procedure) controls the FDR at the level α.

Lambda series based on the Holm correction

Recall that gSLOPE penalize only upper triangle of the estimated matrix, thus we need series of $p(p-1)/2$ values.

The lambda series based on this correction is constructed as follows.

$$
\begin{aligned}
m &= \frac{p(p-1)}{2}, \\
\lambda_k^{\text{Holm}} &= \frac{qt_{n-2}(1-\frac{\alpha k}{m})}{\sqrt{n-2+qt_{n-2}^2(1-\frac{\alpha k}{m})}}, \\
\lambda^{\text{Holm}} &= \{\lambda_1^{\text{Holm}}, \lambda_2^{\text{Holm}}, ..., \lambda_m^{\text{Holm}}\}.
\end{aligned}
$$

Lambda series based on the BH correction

The lambda series based on the BH correction is constructed as follows

$$
\begin{aligned}
m &= \frac{p(p-1)}{2}, \\
\lambda_k^{\text{BH}} &= \frac{qt_{n-2}(1-\frac{\alpha}{m+1-k})}{\sqrt{n-2+qt_{n-2}^2(1-\frac{\alpha}{m+1-k})}}, \\
\lambda^{\text{BH}} &= \{\lambda_1^{\text{BH}}, \lambda_2^{\text{BH}}, ..., \lambda_m^{\text{BH}}\}.
\end{aligned}
$$

Summary

Although we do not present theoretical results for gSLOPE, we conducted a number of simulations and present them in the next chapter. There is visible connection between lambda (series) construction for each method with multiple testing procedures, what might help the reader to understand the concept. gLasso is special case of gSLOPE, where series is constructed using only one value. The graphical representation of concepts are depicted on fig. 3.1. Figure depicts the idea behind gSLOPE - it is relaxation of gLasso, which should achieve the same error rate, but higher discovery rate than gLasso.

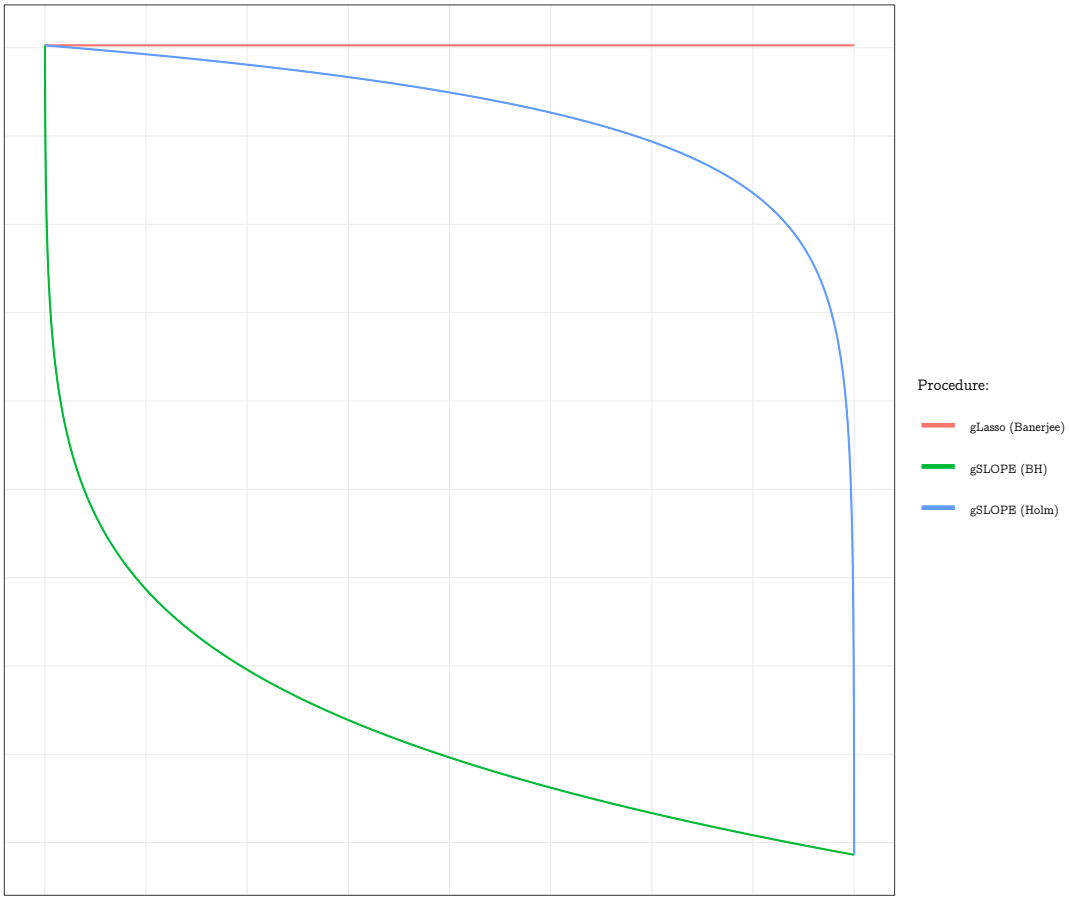


Figure 3.1 – Lambdas for gSLOPE and gLasso: for the given alpha, values of lambdas in each method are sorted in decreasing order on the y-axis. In case of gLasso it is constant-valued series.

Chapter 4

Simulations and results

To verify the concepts and methods proposed in the last chapter we have done the number of simulations under different settings. We did a set of synthetic experiments for which we used R programming language.

4.1 Data

We used package **huge** [RLW12] for data simulation, as it provides a convenient framework for simulating data from high dimensional undirected graphical models. The **huge** package provides a multivariate normal distribution generator for models with different graph structures. There are several parameters to set up:

- the number of observations n (sample size),
- the number of variables p (dimension),
- the graph structure (we focus only on "cluster", "hub" and "scale-free"), see fig. 4.1,
- the value of the off-diagonal, nonzero entries of precision matrix, denoted as v
- the value added to the diagonal elements of precision matrix after transformation to positive semidefinite matrix, denoted as u,
- the number of hubs/clusters,
- in cluster graph: the probability that a pair of nodes has an edge in the cluster, that is $\mathbb{P}(\theta_{ij} \neq 0 \mid \text{vertices } i \text{ and } j \text{ are connected})$.

More detailed information about **huge** is included in [RLW12] and package documentation. As the covariance matrix must be symmetric and positive semidefinite, the package authors used an approach proposed in [CW97], which modifies off-diagonal entries to assure semidefinite positiveness. As this transformation could change precision matrix structure (in terms of magnitudes, not nonzero entries), we mainly use measure denoted as *Magnitude ratio* [MR] defined by

$$\mathrm{MR} = \frac{\text{diagonal entries magnitude}}{\text{off-diagonal entries magnitude}} = \frac{u+v}{v}.$$

Exact arguments used for data generation for each value of MR are given in a table 4.1. The

MR	-1	0	1.25	1.43	1.67	2.5	3.33	5
u	1	1	0.2	0.3	0.4	0.6	0.7	0.8
v	-0.5	-1	0.8	0.7	0.6	0.4	0.3	0.2

Table 4.1 – Break-down of generator arguments.

visualization of the mainly used MR settings is presented on fig. 4.2. In the next sections we refer to $MR = 0$ as a strong setting, to $MR = 1.33$ as a medium setting, and to $MR = 3.33$ as a weak setting. This is motivated by the covariance matrix structure, and the off-diagonal entries which are (relatively) dominated, or not, by the diagonal entries.

4.2 ADMM gLasso or graphical Lasso?

In the section concerning graphical Lasso, we derived the formula for ADMM for graphical Lasso problem (3.5). Now we compare the performance of the original algorithm proposed in [FHT08] with our ADMM algorithm.

First comes the timing comparison. We have done 100 evaluations of each algorithm for two settings: cluster graph and hub graph. The augmented Lagrangian parameter was equal to one (careful parameter choice could speed up the procedure $5 - 10\%$). Results are presented below

	Minimum	Mean	Median	Maximum	# of evaluations
ADMM	178.509	188.8862	183.9140	335.2727	100
graphical Lasso	1.330	1.6455	1.4309	4.3721	100

Table 4.2 – Algorithms execution times comparison for the hub graph structure.

	Minimum	Mean	Median	Maximum	# of evaluations
ADMM	942.9565	995.2295	974.2730	1096.2850	100
graphical Lasso	6.2584	8.1663	6.7732	105.4057	100

Table 4.3 – Algorithms execution times comparison for the cluster graph structure.

The algorithm presented by Friedman et al. is more than ten times faster than ADMM version. It is understandable as the graphical Lasso algorithm is optimized for gLasso problem and ADMM is the general algorithm for solving convex problems.

The second comparison concerns the accuracy of the algorithms. If ADMM for gLasso will be more accurate the graphical Lasso algorithm it might be desirable to use it. The objective function (3.4) value for optimal point found by each algorithm is presented below. The lambda was calculated using the method proposed by Banerjee et al.

	hub	cluster
ADMM	121.3453	266.8644
graphical Lasso	121.3374	266.8423

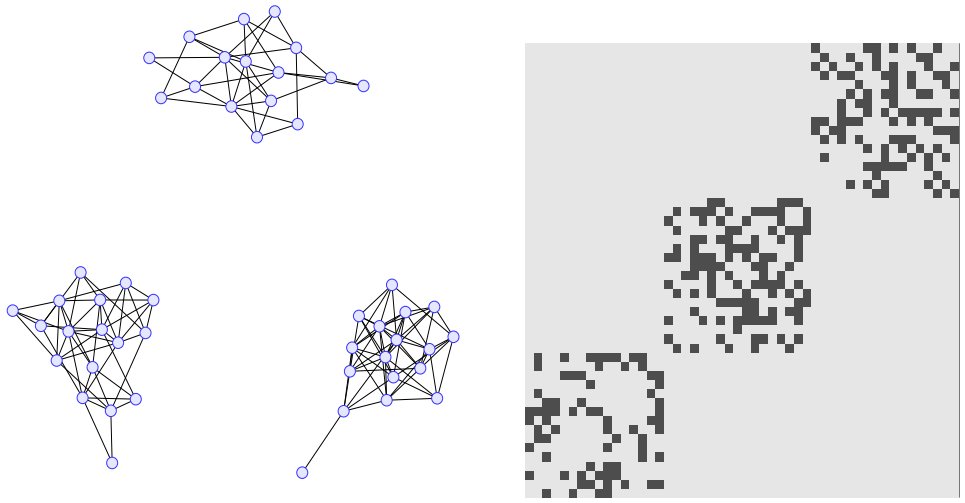
Table 4.4 – Objective function value for optimal point found by each algorithm in two different settings.

The difference between values is insignificant, thus in all calculations we use the graphical Lasso algorithm, as it is faster. ADMM algorithm is still useful in case of graphical SLOPE, where the generic algorithm was not developed earlier.

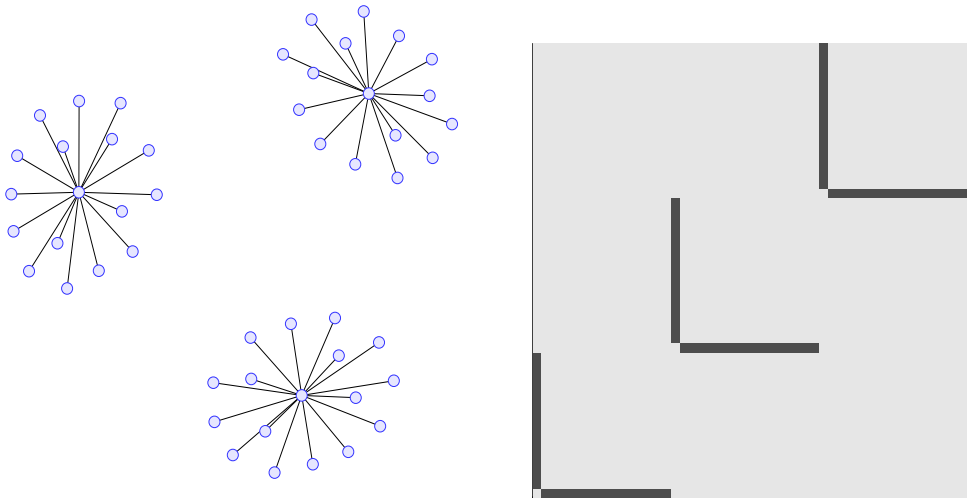
4.3 gSLOPE or gLasso?

The main objective of this thesis is to compare the performance of graphical Lasso and graphical SLOPE. We compare $\lambda^{Banerjee}$ for gLasso and λ^{Holm}, λ^{BH} for gSLOPE in a number of different settings.

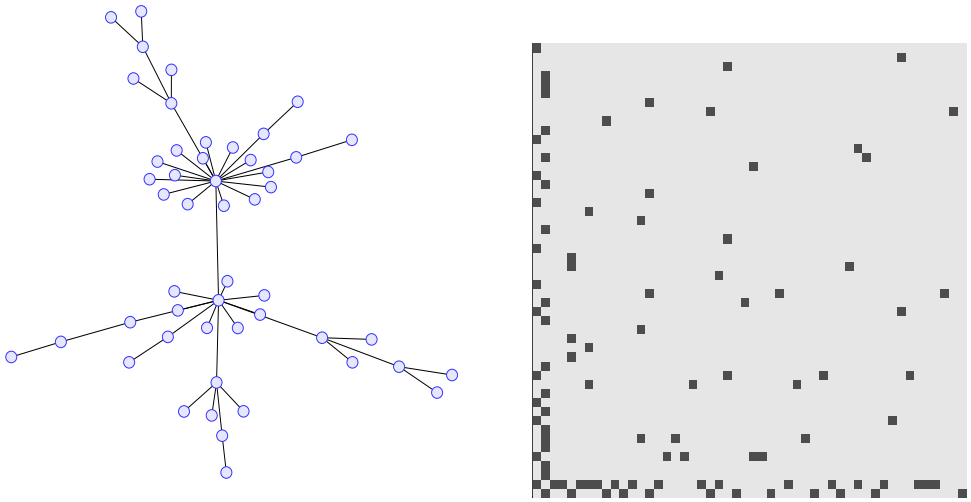
Usually, a number of variables p are equal to 100, a number of observations n vary from 50 to 400, an alpha parameter is equal to 0.05 or 0.2 and there are a set of settings for partial correlations



(a) The cluster graph



(b) The hub graph



(c) The scale-free graph

Figure 4.1 – Comparison of different types of graphs and corresponding precision matrices (nonzero entries are dark).

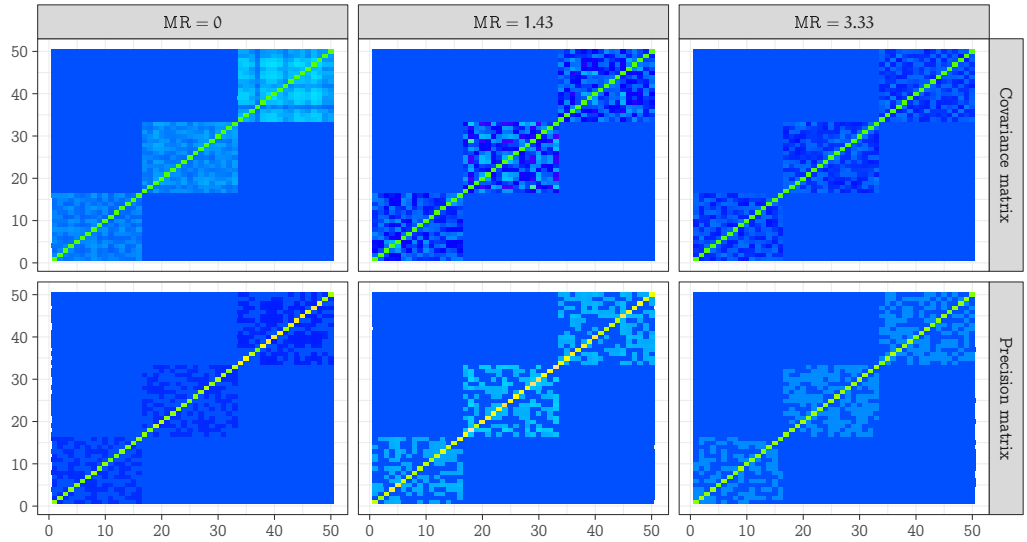


Figure 4.2 – Visualization of various MR settings which helps imagine how change of MR affect the matrix values; the lighter the color, the greater the value of corresponding entry in each matrix.

(diagonal entries are almost always equal to 1). Each simulation was done using 5000 iterations. We divide this sections into subsections, each for different graph structure.

We compared methods in terms of FDR and localFDR control, as this is the motivation behind gLasso and gSLOPE. At the end of each subsection we present ROC curve, described below.

ROC

Using Sensitivity and Specificity defined in the last chapter we could define a *receiver operating characteristic* [ROC] curve. This is a graphical method that illustrates the diagnostic ability of a classifier. The ROC curve is created by plotting the Sensitivity against the $1 - \text{Specificity}$ at various threshold settings. Generally, the larger area under the ROC curve is, the better classifier we have.

As definitions of lambda for each procedure does not guarantee to obtain extreme values (where one of the metrics attain zero or one) by manipulation of parameter alpha α, we propose the different approach. We construct lambda for the parameter α equal to 0.05 and then multiply it by scalars varying from 10^{-4} to 1.75.

4.3.1 Cluster

Cluster density dependence

In this section we examine how sparsity of the cluster affect an effectiveness of procedures. A probability that two nodes $v_i \neq v_j$ in a given cluster $C \subset G$ are adjacent is given by parameter p, that is $\mathbb{P}(\{v_1, v_2\} \in E(C) \mid v_1, v_2 \in C) = p$. Parameter p vary between simulations, it was set to 0.25, 0.5, 0.75 or 1. The value of the parameter affects the density of each cluster in the graph, in the case where $p = 1$ we obtain "full" cluster, that is our graph is made out of disjoint cliques. Each value of the parameter p was simulated in three scenarios, with different MR value. Results are presented on fig. 4.3 and fig. 4.4.

The setting with MR equal to zero differs from two other in terms of performance. In this case, the cliques (i.e. $p = 1$) are pretty well estimated; the power is high, the errors are low. In contrary, for MR equal to 1.43 or 3.33, cliques are badly estimated; although the FDR is seemed to be controlled, the rate of positive discoveries is very small. That might be caused by the fact that diagonal entries strongly dominates over off-diagonal entries of the estimated matrix.

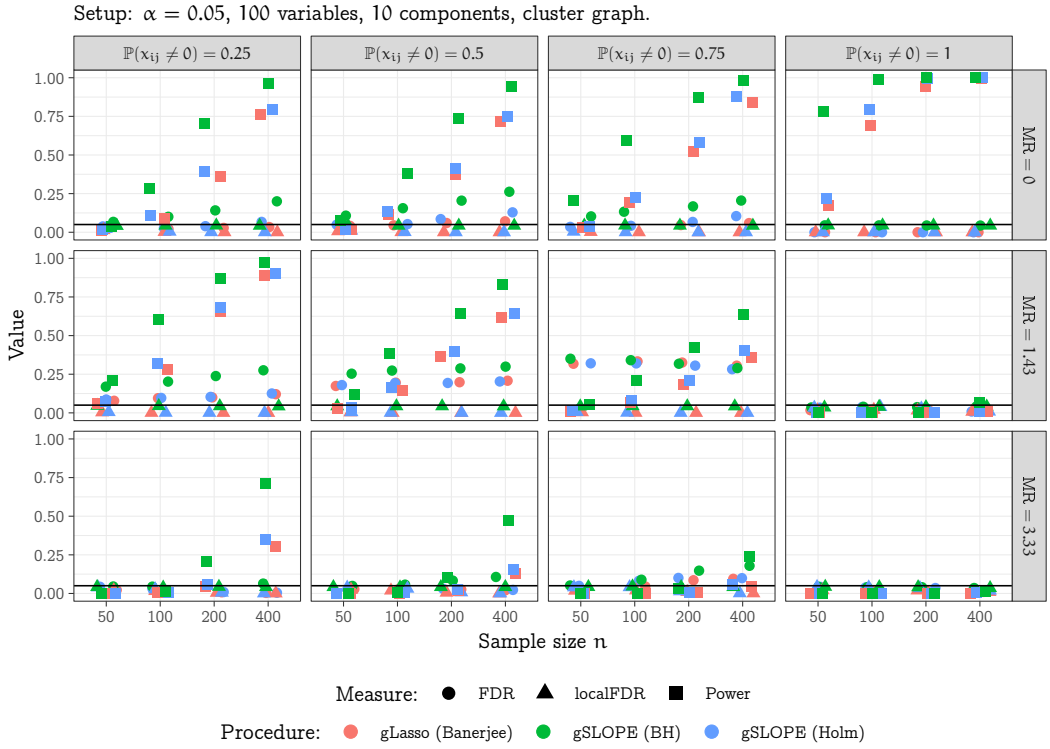


Figure 4.3 – Cluster graphs: The component density comparison with the parameter $\alpha = 0.05$.

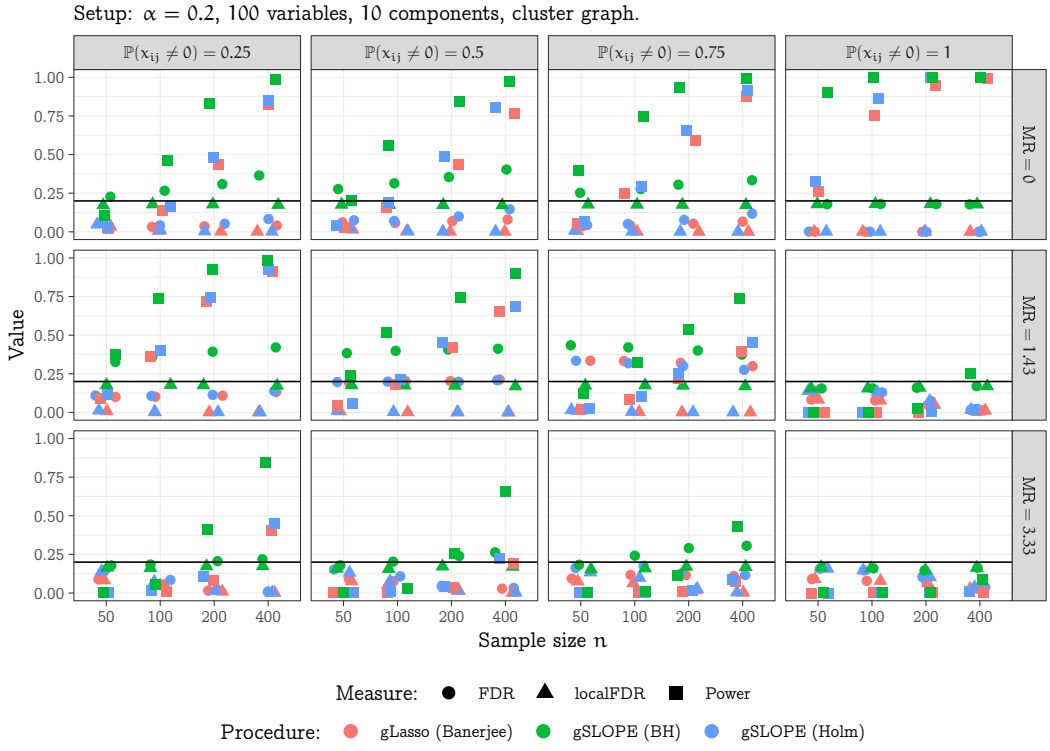


Figure 4.4 – Cluster graphs: The component density comparison with the parameter $\alpha = 0.2$.

Interesting is that in weaker settings (MR equal to 1.33 or 3.33) sparser graphs obtain better results in terms of power. In the stronger setting, the best results in terms of power are achieved for the graph made of cliques. In all best results in terms of power are achieved by gSLOPE with the parameter λ^{BH}. Although FDR is not always controlled, the localFDR is always below the

desired level.

Graphical SLOPE with the parameter λ^{Holm} usually have higher power and than gLasso, and it controls FDR at the same level.

Results between $\alpha = 0.05$ and $\alpha = 0.2$ does not vary.

Partial correlation dependence

We examine how a partial correlation between variables affect the effectiveness of procedures. We compare eight different settings, results are visualized on fig. 4.5 and fig. 4.6.

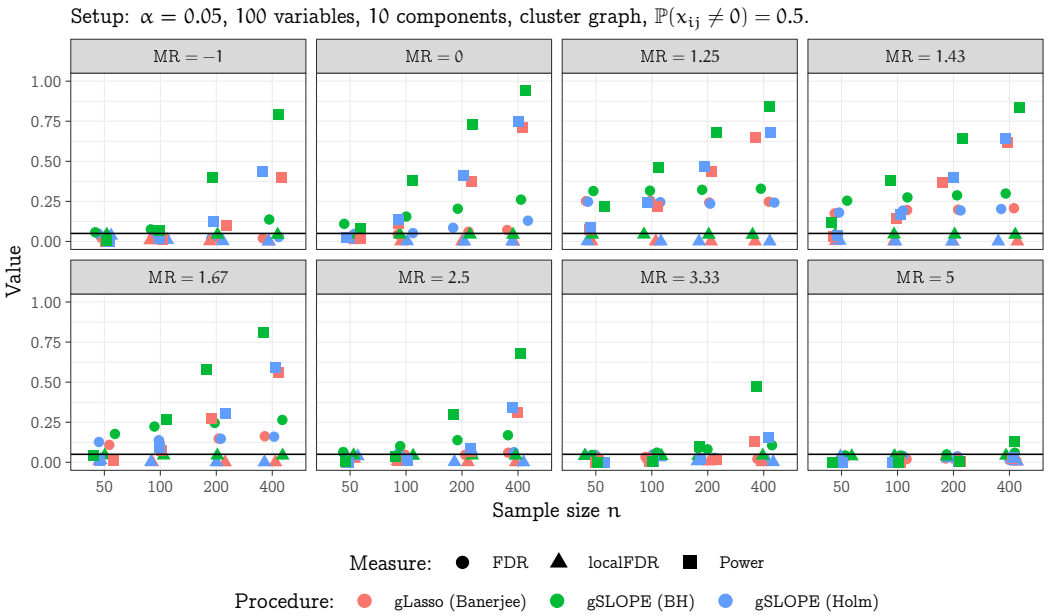


Figure 4.5 – Cluster graphs: MR comparison with the parameter $\alpha = 0.05$.

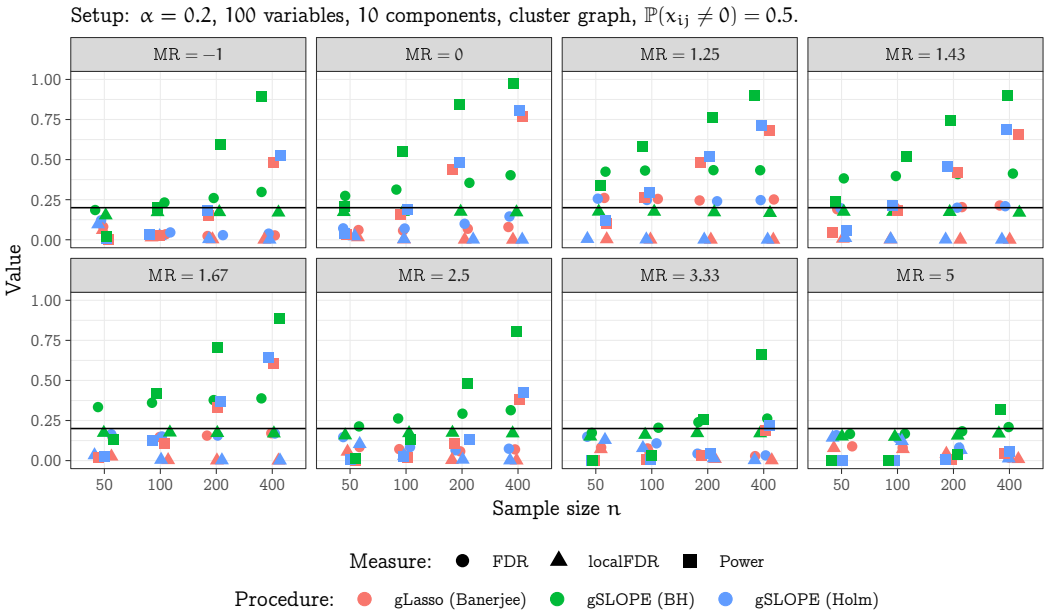


Figure 4.6 – Cluster graphs: MR comparison with the parameter $\alpha = 0.2$.

The partial correlations significantly affect obtained results. In strong settings (MR below 1.5) the power is high, the FDR is usually not controlled, the localFDR is controlled. In weaker setting power is much lower, but errors seem to be controlled.

Basing on this simulations we could suppose that gSLOPE with the parameter based on the BH correction usually gives better results than other methods. In each setting procedure based on λ^{BH} does control localFDR at the level alpha, it also has higher power than other methods.

Results for the parameter $\alpha = 0.05$ seems to be little worse than for $\alpha = 0.2$.

Cluster size dependence

The size of each cluster was changed in the next experiment, graphs with 20 and 5 clusters were introduced. See fig. 4.7 and fig. 4.8 for visualization.

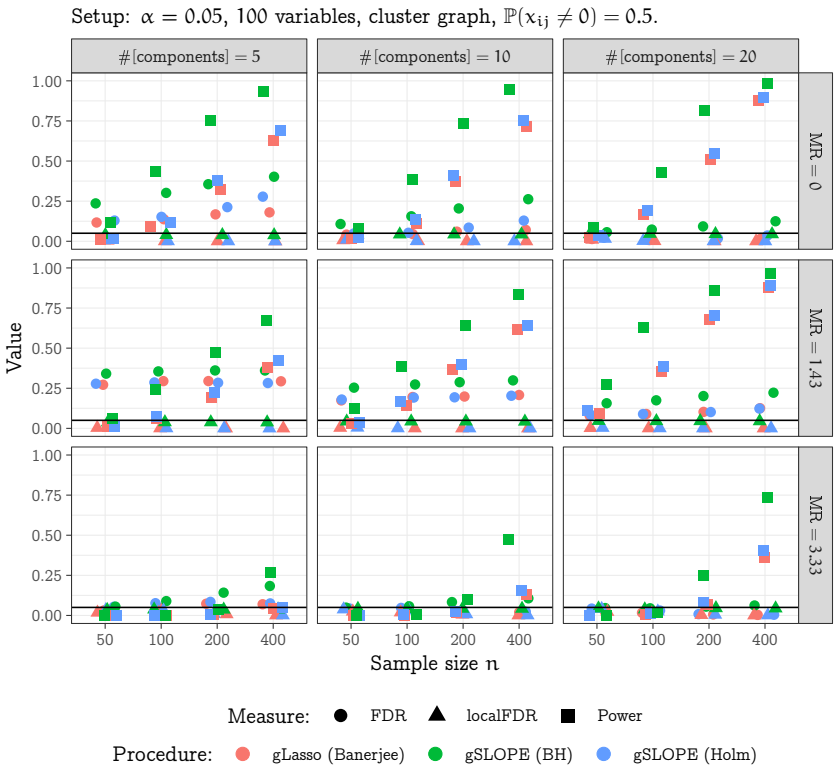


Figure 4.7 – Cluster graphs: The number of components comparison with the parameter $\alpha = 0.05$.

In the case when MR $= 0$ the number of components in a graph does not affect results strongly, the MR is the much more important factor. Although, we can suppose that for a larger number of graph components methods tend to control FDR better and has a higher power.

Results between $\alpha = 0.05$ and $\alpha = 0.2$ does not vary at all.

ROC

We conducted experiments for two connectivity and three MR settings.

On fig. 4.9 we compared settings with MR $= 0$. This is the only case when sparser graphs obtained better results than fully connected ones, but this is in accordance with findings from the first subsection about cluster-type graphs. The non-smooth curve in case when $\mathbb{P}(x_{ij} \neq 0 \mid i, j$ are in the same component$) = 1$ suggest that there is a threshold below which gLasso starts to discover false entries.

For MR $= 1.43$ the results also comply with findings from the first subsection, for the sparser graph the ROC curve is steeper than for fully connected components, see fig. 4.10. In both cases, gSLOPE-based methods outperform gLasso, especially when power is high.

ROC curves for MR $= 3.33$ presented on fig. 4.11 confirm earlier observations - in weak setting all methods cope better with graphs with some sparsity within each cluster. The obtained ROC curve is similar to the one for MR $= 1.43$, so de conclusions.

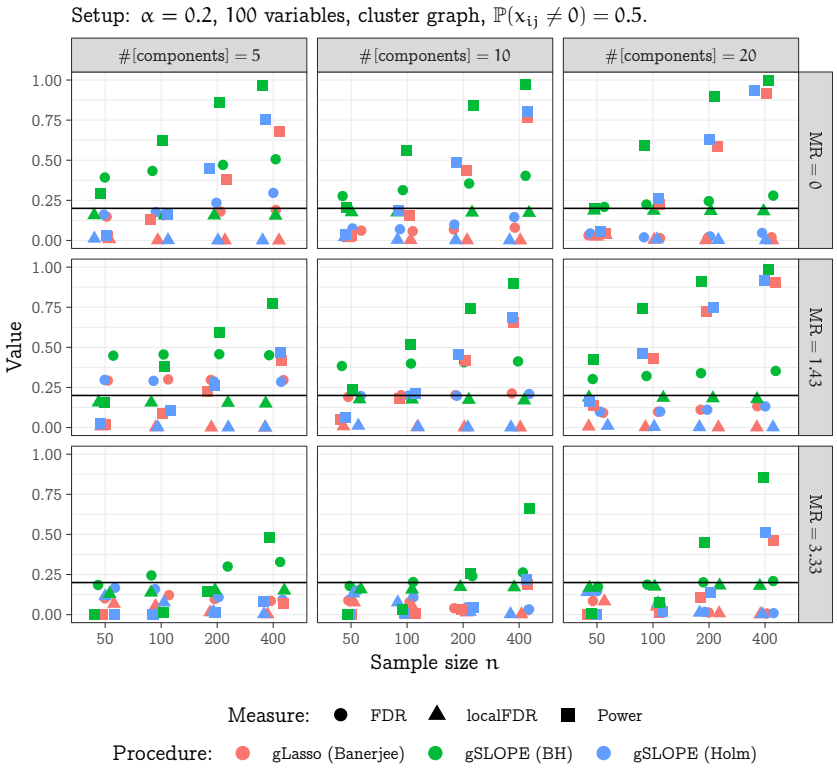


Figure 4.8 – Cluster graphs: The number of components comparison with the parameter $\alpha = 0.2$.

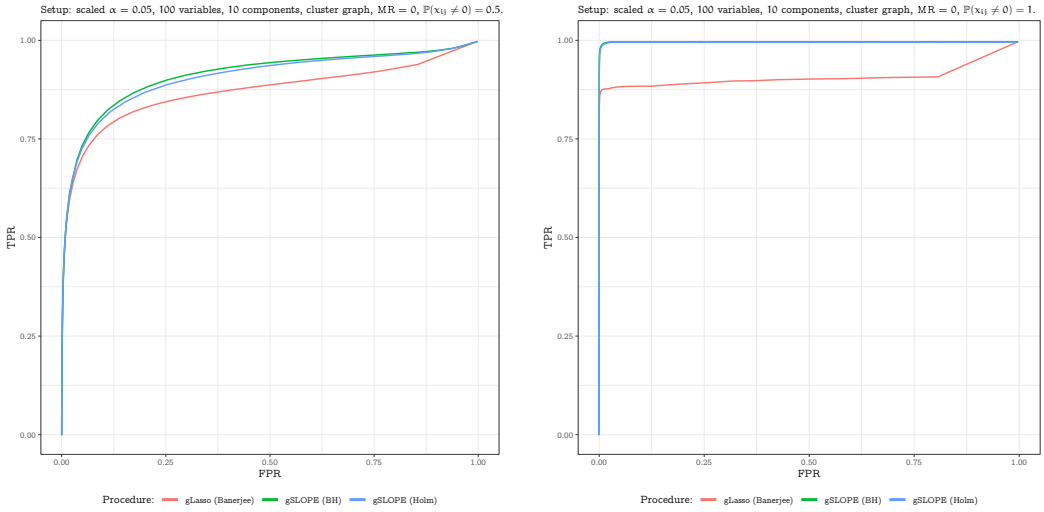


Figure 4.9 – Cluster graphs: ROC curves, MR equal to zero, tho sparsity settings. Lambdas were constructed for alpha $\alpha = 0.05$ and then scaled over appropriate range to achieve proper ROC curve.

4.3.2 Hub

Although the theoretical results for gLasso and gSLOPE do not guarantee any results in for hub graphs, we decided to compare methods in this setting.

Partial correlation dependence

As in the case of cluster type graphs, we examine how the partial correlation between variables affect the effectiveness of procedures. We compare eight different settings, results are presented on fig. 4.12 and fig. 4.13.

Despite the lack of theoretical results all methods seem to recognize the graph pattern without

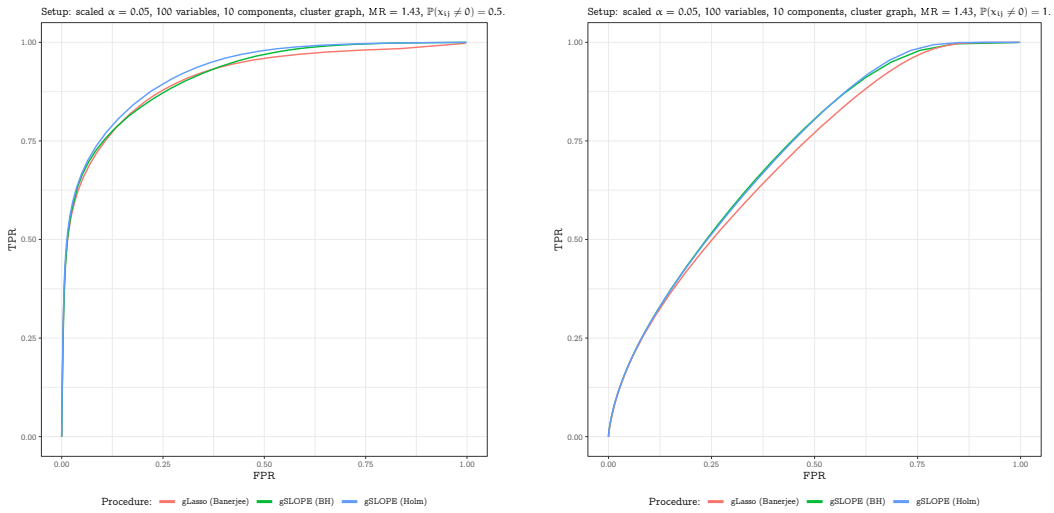


Figure 4.10 – Cluster graphs: ROC curves, MR $= 1.43$, tho sparsity settings. Lambdas were constructed for alpha $\alpha = 0.05$ and then scaled over appropriate range to achieve proper ROC curve.

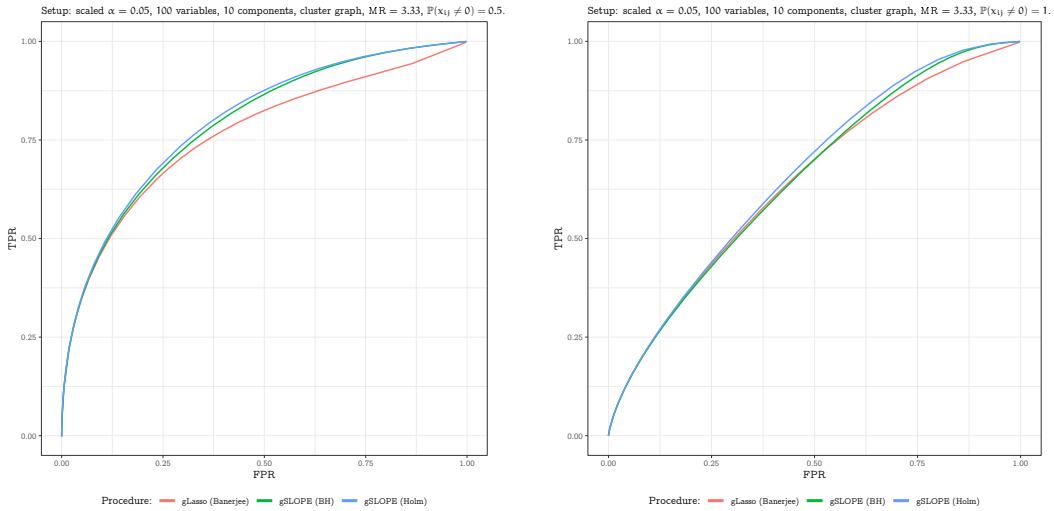


Figure 4.11 – Cluster graphs: ROC curves, MR $= 3.33$, tho sparsity settings. Lambdas were constructed for alpha $\alpha = 0.05$ and then scaled over appropriate range to achieve proper ROC curve.

many local errors. It is not surprising when we think of hub-type graphs. In contrary, the control of FDR is not always the case. Graphical gSLOPE with Holm lambdas and gLasso achieve better results than gSLOPE with BH based on BH, but still, the false discovery rate is high.

In stronger settings, more false discoveries are present, and the obtained power is much higher.

For our research important fact is that the gSLOPE with Holm-based lambda achieve higher power and similar FDR comparing to gLasso.

Hub size dependence

We compare procedures on graphs with a different number of components, see fig. 4.14 and fig. 4.15.

Similarly to the cluster graphs, the number of hubs does not affect results strongly. Graphs with more components achieve better results, but this only occurs when we have more observations. Much more important is the relevance of MR, what is much more visible.

Interesting is that in case of weak setting the FDR is being controlled by all procedures, in case of medium setting the false discoveries are controlled by gLasso and gSLOPE with Holm lambdas, but only when the number of observations is smaller than variables, and the strong setting is somehow similar to the weak why, but has higher power.

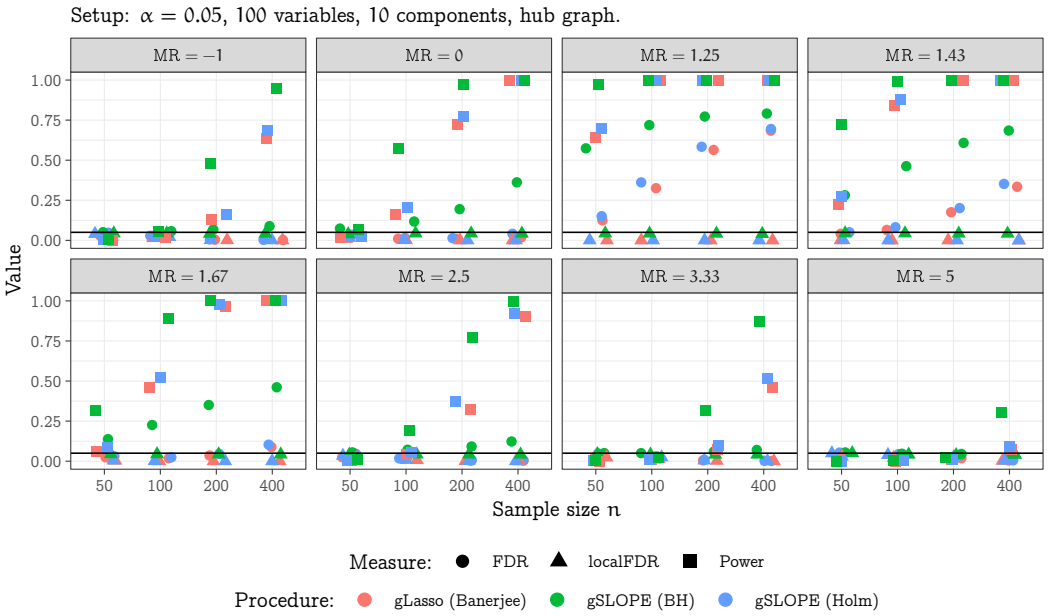


Figure 4.12 – Hub graphs: MR comparison with the parameter $\alpha = 0.05$.

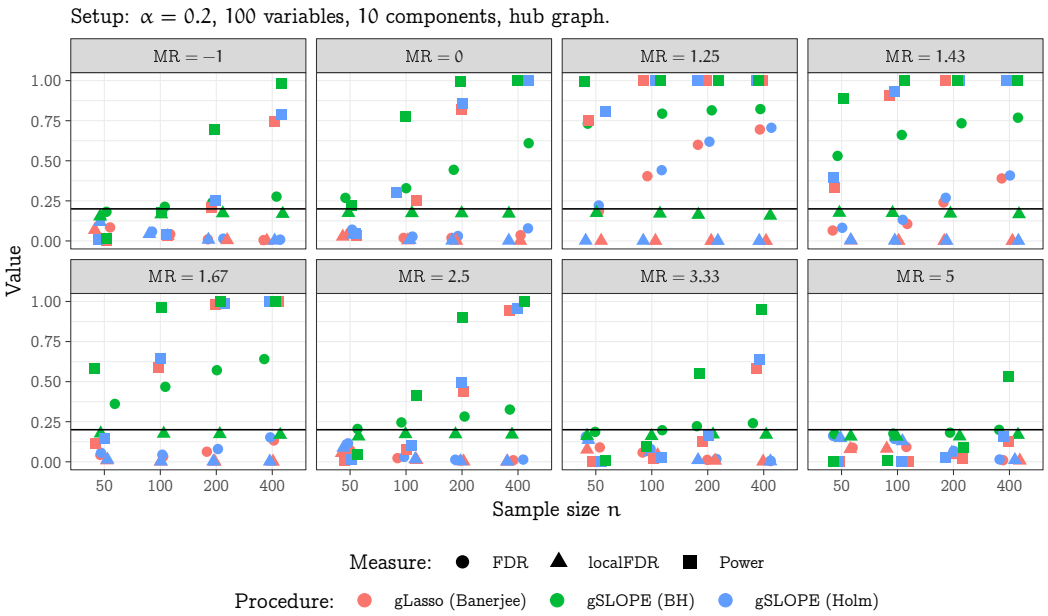


Figure 4.13 – Hub graphs: MR comparison with the parameter $\alpha = 0.2$.

ROC

ROC curves (fig. 4.16 and fig. 4.17) confirm earlier observations - all methods cope better with graphs with smaller MR. The ROC curves for stronger settings are almost rectangular, what suggests that parameter tuning in this cases is crucial, the curves are indistinguishable between the methods. In the case of weaker noise curves based on gSLOPE outperform curve based on gLasso, especially when power is high.

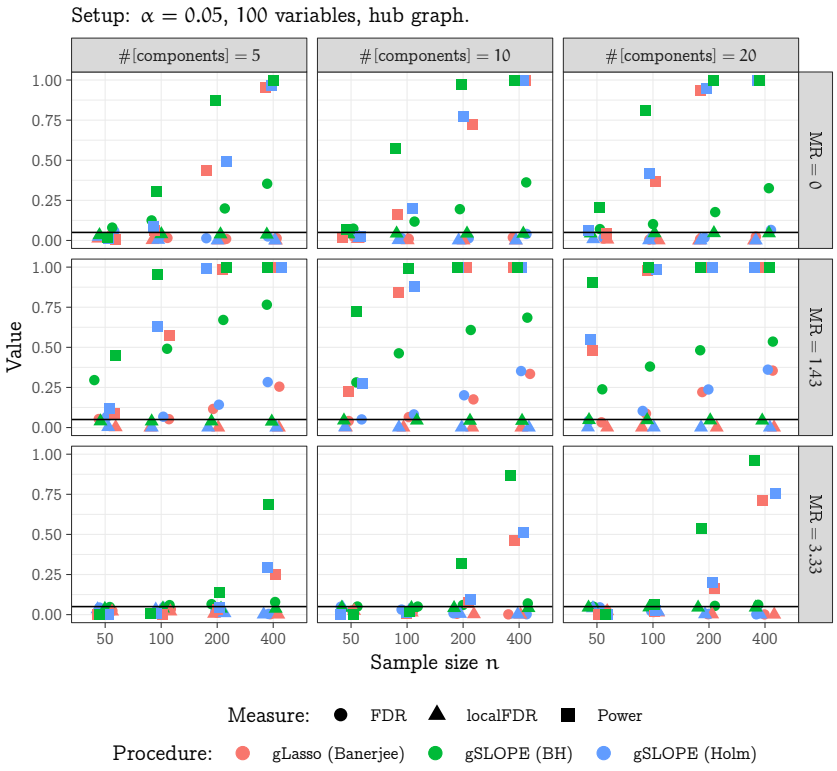


Figure 4.14 – Hub graphs: The number of components comparison with the parameter $\alpha = 0.05$.

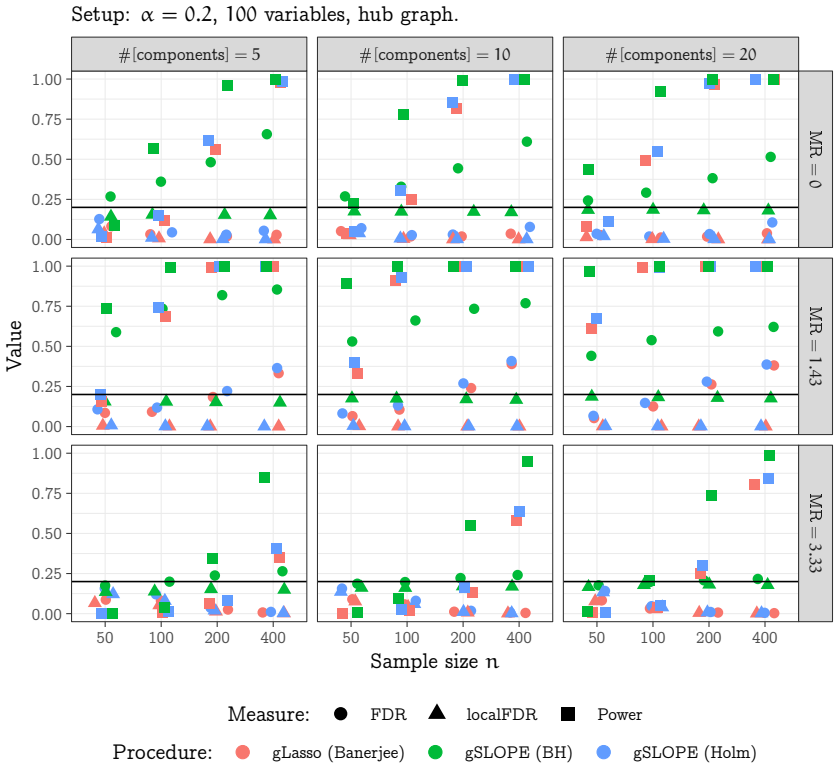


Figure 4.15 – Hub graphs: The number of components comparison with the parameter $\alpha = 0.2$.

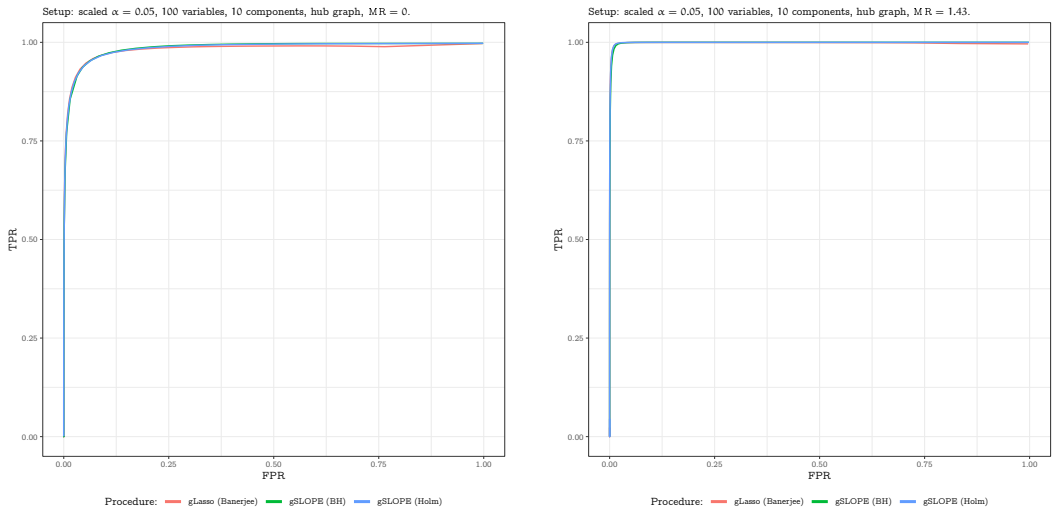


Figure 4.16 – Hub graphs: ROC curves, on the left the MR is low, on the right is medium. Lambdas are constructed for alpha $\alpha = 0.05$ and then scaled over appropriate range to achieve proper ROC curve.

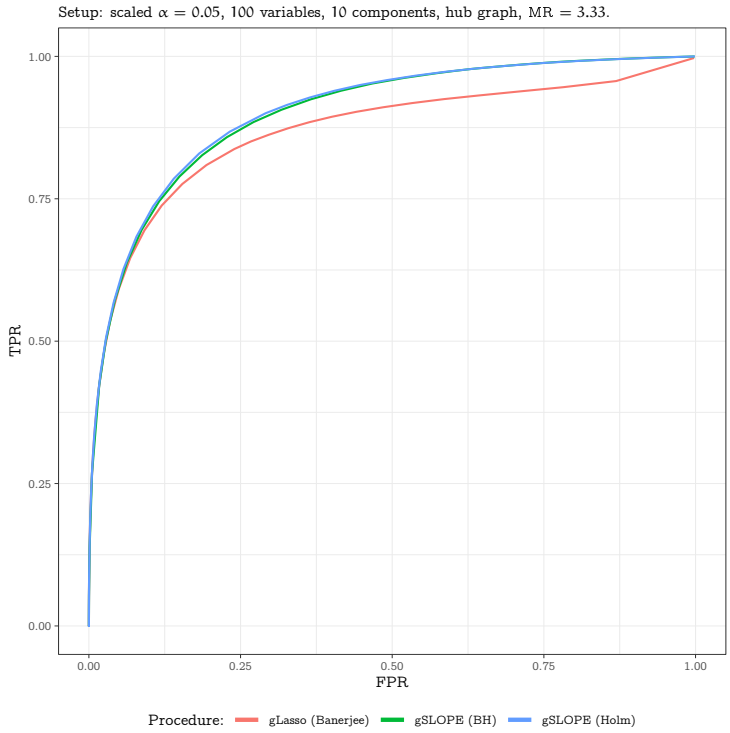


Figure 4.17 – Hub graphs: ROC curve for MR $= 3.33$. Lambdas are constructed for alpha $\alpha = 0.05$ and then scaled over appropriate range to achieve proper ROC curve.

4.3.3 Scale free

Scale-free graphs completely differ from two previous. According to the **huge** manual they are constructed as follows.

"*The graph is generated using B-A algorithm. The initial graph has two connected nodes and each new node is connected to only one node in the existing graph with the probability proportional to the degree of each node in the existing graph. It results in* p *edges in the graph.*"

One more time, the theoretical results for gLasso and gSLOPE do not guarantee any results in this case.

Partial correlation dependence

The generator does not allow to any modifications to the B-A algorithm, we only specified the magnitude of diagonal and off-diagonal entries of the precision matrix. The results are visualized on fig. 4.18 and fig. 4.18.

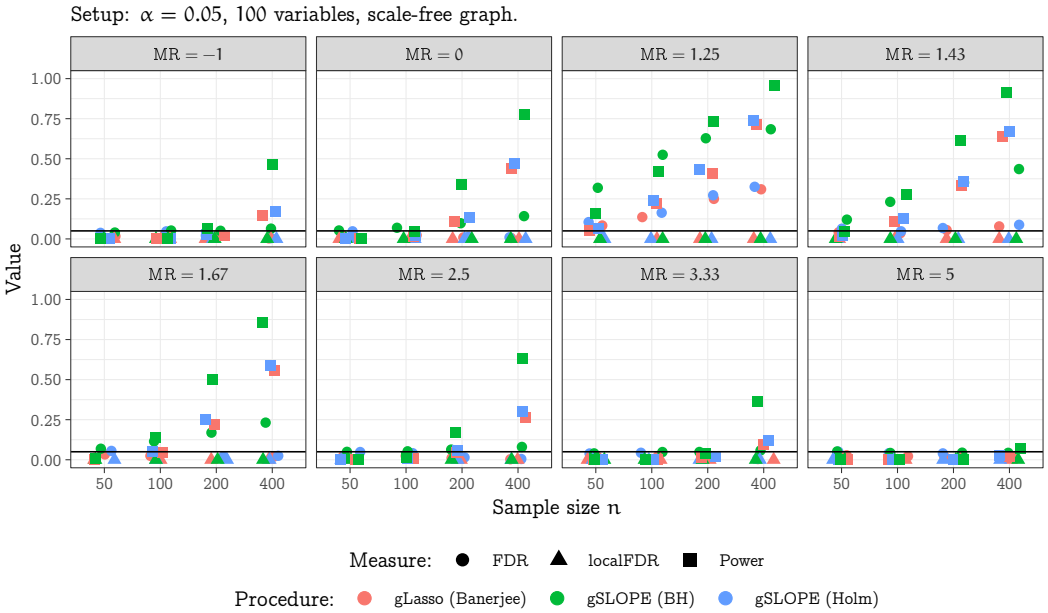


Figure 4.18 – Scale-free graphs: MR comparison with the parameter $\alpha = 0.05$.

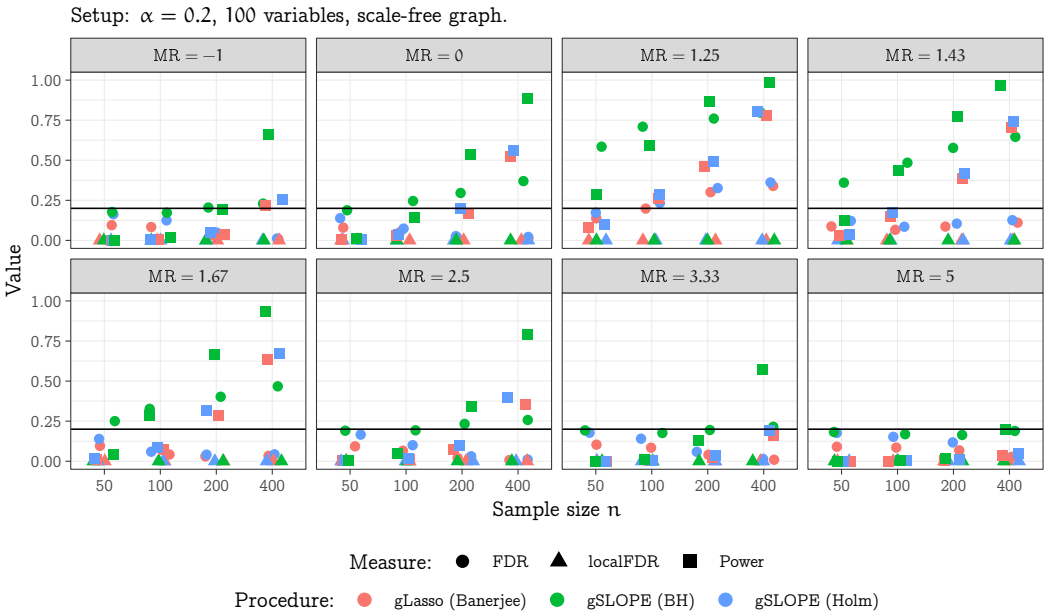


Figure 4.19 – Scale-free graphs: MR comparison with the parameter $\alpha = 0.2$.

All methods seem to commit little errors. As the noise is getting stronger, more false discoveries are present. The gSLOPE with the BH does not control FDR for MR $\leqslant$ 2.5, two other methods does not control false discoveries for MR $\leqslant$ 1.25. A power achieved by gSLOPE with Holm correction is higher than achieved by gLasso. The localFDR is almost always equal to zero, but this is implied by the graph structure; almost always there is a path between two chosen vertices.

ROC

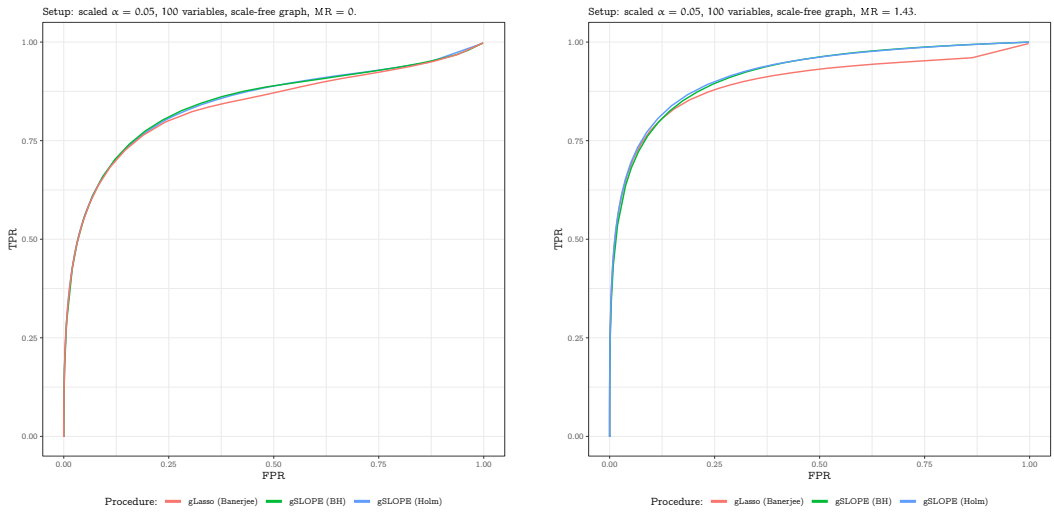
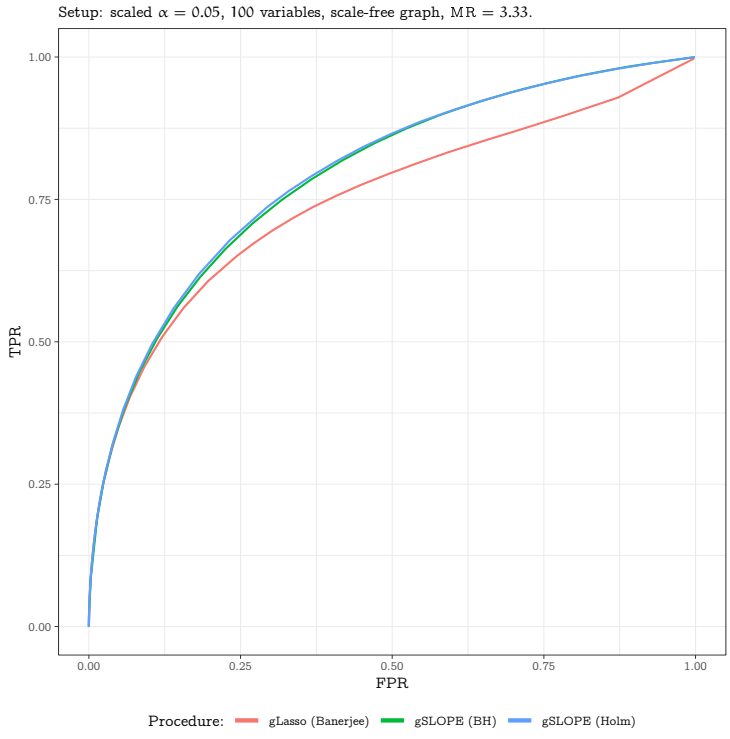


Figure 4.20 – Scale-free graphs: ROC curves, on the left the MR is low, on the right is medium. Lambdas are constructed for alpha $\alpha = 0.05$ and then scaled over appropriate range to achieve proper ROC curve.



(a)

Figure 4.21 – Scale-free graphs: ROC curve for MR $= 3.33$. Lambdas are constructed for alpha $\alpha = 0.05$ and then scaled over appropriate range to achieve proper ROC curve.

ROC curves visualized on fig. 4.20 and fig. 4.21 are similar to those obtained for hub graphs. The curve is steeper for smaller MR, it is reasonable as the number of positive discoveries was higher, but it was occupied by the number of false discoveries.

Chapter 5

Summary and conclusion

In the thesis, we faced the problem of precision matrix estimation in sparse Gaussian graphical models. We presented some basics definitions of convex optimization theory which were necessary to understand ADMM algorithm and its formulation for our usage. We gave the little background of graph theory, which was crucial to understanding the main subject of the thesis. At the beginning of the first two chapters, there might be parts boring for a more experienced reader, but we believe that little introduction, especially in a Master's thesis, is not an issue. The part regarding ADMM for graphical SLOPE is our work, as well as all simulations which were presented in the fourth chapter. The ADMM method was never used earlier for this application.

Chapters three and four are the most import in this paper. We proposed 3 different approaches, two of them are novel methods. We compared them in many different settings. Although we do not prove any theoretical results on paper, the outcomes of the simulations gave promising findings. For all three methods the choice of regularizer is crucial for the performance. We found that gSLOPE could outperform gLasso in many settings, but the one, universal regularizer is hard to point out. The lambda series based on the BH correction usually have greater power than two other methods, but it sometimes does not hold the false discovery rate. On the other hand, gSLOPE with Holm correction outperforms gLasso by a margin, but it did have FDR at the same level. There ware settings in which none of them controls false discoveries, but then, the BH correction achieved the highest power. In terms of localFDR all methods perform very well, of course gSLOPE with BH lambdas obtain the highest power.

We believe there should be more research work done around theoretical principles of gSLOPE, although the results obtained during experiments for this thesis could suggest the proper method for applications. Having some background about data, the proper choice of the method might be based on our findings.

Bibliography

[BEd08] Onureena Banerjee, Laurent El Ghaoui, and Alexandre d'Aspremont. "Model Selection Through Sparse Maximum Likelihood Estimation for Multivariate Gaussian or Binary Data". In: *The Journal of Machine Learning Research* 9 (2008), pp. 485–516. URL: arxiv.org/abs/0707.0704.

[Bes74] Julian Besag. *Spatial Interaction and the Statistical Analysis of Lattice Systems*. 1974. DOI: 10.2307/2984812. URL: https://www.jstor.org/stable/2984812.

[BH95] Yoav Benjamini and Yosef Hochberg. *Controlling the False Discovery Rate: A Practical and Powerful Approach to Multiple Testing*. Tech. rep. 1. 1995, pp. 289–300.

[Bog+15] Małgorzata Bogdan et al. "SLOPE - Adaptive variable selection via convex optimization". In: *The annals of applied statistics* 9.3 (2015), pp. 1103–1140. DOI: 10.1214/15-AOAS842. URL: http://www.ncbi.nlm.nih.gov/pubmed/26709357%20http://www.ncbi.nlm.nih.gov/pubmed/26709357.

[Boy+10] Stephen Boyd et al. "Distributed Optimization and Statistical Learning via the Alternating Direction Method of Multipliers". In: *Machine Learning* 3.1 (2010), pp. 1–122. DOI: 10.1561/2200000016. URL: https://web.stanford.edu/~boyd/papers/pdf/admm_distr_stats.pdf.

[BV09] Stephen Boyd and Lieven Vandenberghe. *Convex Optimization*. Seventh. Cambridge University Press, 2009, p. 730. ISBN: 978-0-521-83378-3. URL: http://web.stanford.edu/~boyd/cvxbook/bv_cvxbook.pdf.

[Can15] Emmanuel Candes. *Advanced Topics in Convex Optimization*. 2015. URL: http://statweb.stanford.edu/~candes/math301/index.html.

[CW97] Grace Chan and Andrew T.A. Wood. "Algorithm AS 312: An Algorithm for Simulating Stationary Gaussian Random Fields". In: *Journal of the Royal Statistical Society: Series C (Applied Statistics)* 46.1 (Feb. 1997), pp. 171–181. ISSN: 0035-9254. DOI: 10.1111/1467-9876.00057. URL: http://doi.wiley.com/10.1111/1467-9876.00057.

[Die17] Reinhard Diestel. *Graph Theory*. Ed. by Sheldon Axler and Kenneth Ribet. Fifth. Springer-Verlag Berlin Heidelberg, 2017, pp. XVIII, 428. ISBN: 978-3-662-53621-6. URL: https://www.springer.com/gp/book/9783662536216.

[EF98] Jonathan Eckstein and Michael C. Ferris. "Operator-Splitting Methods for Monotone Affine Variational Inequalities, with a Parallel Application to Optimal Control". In: *INFORMS Journal on Computing* 10.2 (May 1998), pp. 218–235. ISSN: 1091-9856. DOI: 10.1287/ijoc.10.2.218. URL: http://pubsonline.informs.org/doi/abs/10.1287/ijoc.10.2.218.

[FHT08] Jerome Friedman, Trevor Hastie, and Robert Tibshirani. "Sparse inverse covariance estimation with the graphical lasso". In: *Biostatistics (Oxford, England)* 9.3 (July 2008), pp. 432–41. DOI: 10.1093/biostatistics/kxm045. URL: http://www.ncbi.nlm.nih.gov/pubmed/18079126%20http://www.ncbi.nlm.nih.gov/pubmed/18079126.

[Gri73] G. R. Grimmett. "A Theorem about Random Fields". In: *Bulletin of the London Mathematical Society* 5.1 (Mar. 1973), pp. 81–84. ISSN: 00246093. DOI: 10.1112/blms/5.1.81. URL: http://doi.wiley.com/10.1112/blms/5.1.81.

[Has+15] Trevor Hastie et al. *Statistical Learning with Sparsity: The Lasso and Generalizations*. Chapman and Hall/CRC, 2015, p. 367. ISBN: 9781498712163. URL: https://web.stanford.edu/~hastie/StatLearnSparsity_files/SLS.pdf.

[HMC13] Robert V. Hogg, Joseph W. McKean, and Allen T. Craig. *Introduction to mathematical statistics*. Seventh. Pearson, 2013, p. 694. ISBN: 978-0-321-79543-4.

[HTF09] Trevor Hastie, Robert Tibshirani, and Jerome Friedman. *The Elements of Statistical Learning*. Second. Springer-Verlag New York, 2009, p. 745. ISBN: 978-0-387-84858-7. DOI: 10.1007/978-0-387-84858-7.

[KF09] Daphne Koller and Nir Friedman. *Probabilistic graphical models: principles and techniques*. First. MIT Press, 2009, p. 1270. ISBN: 9780262013192.

[Lau96] Steffen L. Lauritzen. *Graphical models*. Clarendon Press, 1996, p. 298. ISBN: 9780198522195. URL: https://global.oup.com/academic/product/graphical-models-9780198522195?cc=pl&lang=en&.

[Nis+15] Robert Nishihara et al. "A General Analysis of the Convergence of ADMM". In: (2015). ISSN: 14757516. DOI: 10.1109/CEC.2014.6900235.

[NW06] Jorge Nocedal and Stephen Wright. *Numerical Optimization*. Second. New York: Springer-Verlag, 2006, pp. XXII, 664. ISBN: 978-0-387-40065-5. DOI: 10.1007/978-0-387-40065-5.

[Par14] Neal Parikh. "Proximal Algorithms". In: *Foundations and Trends® in Optimization* (2014). ISSN: 2167-3888. DOI: 10.1561/2400000003.

[PSW15] Nicholas G. Polson, James G. Scott, and Brandon T. Willard. "Proximal Algorithms in Statistics and Machine Learning". In: (2015). ISSN: 0883-4237. DOI: 10.1214/15-STS530.

[RLW12] Kathryn Roeder, John Lafferty, and Larry Wasserman. *The huge Package for High-dimensional Undirected Graph Estimation in R*. 2012. URL: https://github.com/cran/huge.

[Roc] R. Tyrrell Rockafellar. *Convex analysis*, p. 451. ISBN: 9780691015866. URL: https://press.princeton.edu/titles/1815.html.

[Sho85] Naum Zuselevich Shor. *Minimization methods for non-differentiable functions*. Springer-Verlag, 1985, p. 162. ISBN: 0-387-12763-1.

[Sob18] (not published yet) Piotr Sobczyk. "Identifying low-dimensional structures through model selection in high-dimensional data". PhD thesis. Wrocław University of Science and Technology, 2018.

[VBW98] Lieven Vandenberghe, Stephen Boyd, and Shao-Po Wu. "Determinant Maximization with Linear Matrix Inequality Constraints". In: *SIAM Journal on Matrix Analysis and Applications* 19.2 (Apr. 1998), pp. 499–533. ISSN: 0895-4798. DOI: 10.1137/S0895479896303430. URL: http://epubs.siam.org/doi/10.1137/S0895479896303430.