SAPIENZA
UNIVERSITÀ DI ROMA

Dipartimento di Scienze Statistiche

Philosophy Doctor in Operations Research

Nonlinear formulation of Semidefinite Programming and Eigenvalue Optimization

Application to Integer Quadratic Problems

Ph.D. Candidate:
Mauro Piacentini

Coordinator:
Prof. Stefano Lucidi

Advisor:
Prof. Laura Palagi

Acknowledgements

To my life Daniela and Giulia, to my family Mario Laura Ale and Vale, to my friends Michele Peky Fabio Beppoccio Raffaella Lorena Roberta Alessia Carlo Simone Bianca, To my pets Linda and Paweł

To my dismates Carla Marianna Tiziana Lorenzo Simone Federico Andrea Francesco

To my Professors Grippo Palagi Piccialli Rendl

Contents

Glossary v

Preface vii

1 Semidefinite Programming 1

1.1 Semidefinite Programming problem 1

1.2 Duality Theory 4

1.3 Interior Point methods 6

1.4 Geometry 9

2 Low-Rank formulation of SDP problems 15

2.1 Low-Rank formulation of the standard primal SDP problem 16

2.2 Optimality conditions 18

2.3 Solution approach for the Low-Rank formulation 22

2.3.1 Augmented Lagrangian approach for general Low-Rank problems 24

2.3.2 Exact Penalty approach for particular Low-Rank problems . . . 25

3 Low-Rank SDP Heuristic for large-scale Max-Cut 29

3.1 Max-Cut 30

3.2 SDP relaxation 33

3.3 Low-Rank SDP relaxation 34

3.3.1 Quotient formulation 37

3.3.2 Regularized Quotient function 39

3.4 Heuristic for large-scale Max-Cut 44

3.5 Numerical results 47

4 Eigenvalue Optimization 57

4.1 Extreme Eigenvalue Optimization 57

4.1.1 SDP formulation 59

4.1.2 Low-Rank SDP formulation 63

4.2 Smallest Eigenvalues of symmetric matrices 72

4.2.1 Nonlinear constrained formulation 72

4.2.2 Nonlinear unconstrained formulation for the Smallest Eigenvalue 79

4.3 Numerical results 87

5 Integer Quadratic Programming 95

5.1 Continuous relaxations 96

5.1.1 Non-convex case 97

5.1.1.1 Solution approach 101

5.1.1.2 Choice of the Ellipsoid 103

5.1.1.3 Scaling quadratic problems 106

5.1.2 Convex case 107

5.2 Branch-and-Bound scheme 109

5.3 Numerical results 114

Appendices 120

A Subgradient method over the Simplex 121

B Backround material 123

B.1 Eigenvalues and Semidefinite matrices 123

B.2 Kronecker product 125

References 127

Glossary

$\mathbb{R}^n$ space of real n-dimensional vector

$\mathbb{S}^n$ space of real $n \times n$ symmetric matrices

$\mathbb{D}^n$ space of real $n \times n$ diagonal matrices

$\det(A)$ determinant of a matrix

$\mathrm{rank}(A)$ rank of a matrix

$\mathrm{Im}(A)$ span of the columns of a matrix

$\ker(A)$ kernel of a matrix

$\succeq$ Löwner partial order

$A \succeq 0$ A symmetric semidefinite positive matrix

$A \succ 0$ A symmetric definite positive matrix

$A \bullet B$ scalar product over matrices

$\mathcal{A}(\cdot)$ linear operator over matrices

$\mathcal{A}^T(\cdot)$ adjoint linear operator of $\mathcal{A}(\cdot)$

I identity matrix of appropriate dimension

I_n identity matrix of dimension n

U matrix of all ones of appropriate dimension

U_n matrix of all ones of dimension n

u vector of all ones of appropriate dimension

u_n vector of all ones of dimension n

e_i vector of all zero, expect 1 in position i

E_{ij} square matrix of all zero, expect 1 in position ij

$\mathrm{diag}(A)$ diagonal vector of a matrix

$\mathrm{Diag}(v)$ diagonal matrix with main diagonal the vector v

$\otimes$ Kronecker product

$\mathrm{vec}(A)$ vector obtained by stacking columns of A

GLOSSARY

$\mathrm{mat}(v)$	inverse operation of $\mathrm{vec}(\cdot)$
$(\alpha)_+$	scalar $\max(0, \alpha)$
$\mathcal{B}_\rho(x)$	sphere centered at x with ray ρ

Preface

This thesis deals with nonlinear differentiable formulations and algorithms for particular problems in Semidefinite Programming (SDP) and in Eigenvalue Optimization (EO). These nonlinear techniques will be used as a tool to solve exactly or approximately general integer quadratic problems: in particular, we will consider general integer quadratic problems, with no assumption of convexity, as well as the particular integer formulation for the Max-Cut.

SDP belongs to the class of Conic Programming and it consists in minimizing a linear function over the intersection of the cone of semidefinite matrices with an affine subspace.
Convexity and duality theory in SDP led to the definition of theoretically and practically efficient solution algorithms, such as Interior Point Methods. Nevertheless, the widespread applications of SDP have thus brought to the need of having efficient SDP solvers for large-scale problems. Among the possible approaches for large-scale SDP problems, there is the so called Low-Rank SDP formulation based on rectangular factorization of semidefinite matrices. This approach led to a special structured nonlinear non-convex problem, for which is possible to define necessary and sufficient global optimality conditions.

Others nonlinear formulations of SDP problems are based on EO. The relevance of EO is beyond the equivalence with SDP. Given $A(x)$, a symmetric affine function of $x \in \mathbb{R}^m$, the problem is to minimize some convex combination of the eigenvalues of $A(x)$ with x subject to linear constraints.

In the first part of the thesis, Chapter 3, we address the special SDP problem coming out as the SDP relaxation of the $\{-1, 1\}$ Quadratic Problem, which can represent an integer formulation for the Max-Cut. We consider the Low-Rank formulation (LRSDP), which corresponds to a nonlinear programming problem of minimizing a quadratic function subject to separable quadratic equality constraints. We prove the equivalence of this constrained non-convex problem with the unconstrained problem defined by means a continuously differentiable merit function. We define an efficient and globally convergent algorithm, called SpeeDP, for finding critical points of the LRSDP problem and integrated with the global optimality conditions known for LRSDP. We provide evidence of the effectiveness of SpeeDP by comparing it with other existing codes on an extended benchmark. We further include SpeeDP within an enhanced version of the Goemans-Williamson algorithm and we show that the corresponding heuristic SpeeDP-MC can generate high-quality cuts for very large and sparse graphs in reasonable time.

In the second part, Chapter 4, we consider the extreme eigenvalue optimization problem (EEO). Based on the Low-Rank approach, we reformulate this non-smooth problem as a smooth and non-convex optimization problem and define new global optimality conditions which can not be derived from the ones for the general Low-Rank SDP problem. This results in the definition of an efficient algorithm for the solution of EEO, for which we report preliminary computational evidence.

Using the same machinery, we also rewrite the problem of finding the k smallest eigenvalues of a symmetric matrix with a new constrained formulation. For the special case of the smallest eigenvalue, we also derive a new unconstrained formulation, suitable for large-scale instances.

In the last part, Chapter 5, we address the general integer quadratic problem, with bound on variables and with no assumption of definiteness on the quadratic term. With the aim of finding a good lower bound on the integer problem, we consider as a relaxation the minimization of the objective function over an ellipsoid enclosing the feasible region. In order to strength the relaxation, we propose to solve a non-smooth constrained problem to find a good ellipsoid. Moreover, we concentrate also on nonlinear techniques to accelerate the solution of the relaxation. Finally, we embed this relaxation in a particular branch-and-bound scheme and examine the performance of the overall algorithm by extensive experiments.

1

Semidefinite Programming

Semidefinite Programming came out in the last two decades as powerful tool in several areas, such as system and control theory, combinatorial optimization, approximation theory and robust optimization.
In Semidefinite Programming, a linear function is optimized over the space of symmetric matrices, subject to affine constraints and semidefinite constraints. In this sense, it can be viewed as the simplest nonlinear extension of Linear Programming as well as a particular case of the most general Conic Programming.
Respect to Linear Programming, Semidefinite Programming keeps convexity, duality theory, theoretically and practically efficient solution algorithms, such as Interior Point methods.

In this chapter we recall all main topics in Semidefinite Programming: standard formulation, duality, Interior Point methods and geometry. For a very exhaustive coverage of these topics for Semidefinite Programming, the book [56] is highly advised.

1.1 Semidefinite Programming problem

Given matrices $C, A_1, \dots, A_m \in \mathbb{S}^n$ and $b \in \mathbb{R}^m$, the SDP problem in primal standard form is

$$\begin{aligned} \min_X \quad & C \bullet X \\ & A_i \bullet X = b_i \quad i = 1, \dots, m, \\ & X \succeq 0. \end{aligned} \tag{P}$$

To keep statements clear, we introduce the linear operator $\mathcal{A} : \mathbb{S}^n \to \mathbb{R}^m$ defined as

$$\mathcal{A}(X) = \begin{bmatrix} A_1 \bullet X \\ \vdots \\ A_m \bullet X \end{bmatrix},$$

and its adjoint operator $\mathcal{A}^T : \mathbb{R}^m \to \mathbb{S}^n$ defined as

$$\mathcal{A}^T(y) = \sum_{i=1}^{m} y_i A_i.$$

Under this notation, we rewrite the primal problem as

$$\begin{aligned} \min_{X} \quad & C \bullet X \\ & \mathcal{A}(X) = b, \\ & X \succeq 0. \end{aligned} \tag{P}$$

The associated dual problem in standard form is given by

$$\begin{aligned} \max_{y,S} \quad & b^T y \\ & \mathcal{A}^T(y) + S = C \\ & y \in \mathbb{R}^m, S \succeq 0. \end{aligned} \tag{D}$$

The dual (D) is obtained by expressing the primal (P) as min-max problem, namely

$$\min_{X \succeq 0} \max_{y \in \mathbb{R}^m} C \bullet X + y^T \left(b - \mathcal{A}(X) \right),$$

exchanging the min with the max,

$$\max_{y \in \mathbb{R}^m} \min_{X \succeq 0} \left(C - \mathcal{A}^T(y) \right) \bullet X + b^T y,$$

and solving the inner minimization.

Although (P) and (D) look like different, they own an hidden symmetry between each other. For this purpose, we need an assumption on the linear independence of $A_1, \dots, A_m$, which is a standard requirement in Semidefinite Programming. Actually, if this were not true then we would have two cases:

(i) for any y such that $\mathcal{A}^T(y) = 0$ we have $b^T y = 0$. In this case, we could safely remove the redundant primal constraints and the corresponding dual variables, keeping the problem equivalent;

(ii) there exists $\bar{y} \in \mathbb{R}^m$ such that $\mathcal{A}^T(\bar{y}) = 0$ and $b^T y > 0$. It follows that (D) is unbounded and (P) unfeasible.

The assumption implies we can always find a $Z \in \mathbb{S}^n$ such that $\mathcal{A}(Z) = b$. Moreover, chosen a basis $G_1, \ldots, G_k$ for the orthogonal complement to the span$(A_1, \ldots, A_m)$, scalars $h_j = C \bullet G_j$, with $j = 1, \ldots, k$, are defined.

Lemma 1.1.1 *Problem* (P) *can be rewritten in standard dual form, namely as*

$$\begin{aligned} \max_{w,X} \quad & -h^T w + [C \bullet Z] \\ & \mathcal{A}^T(w) + X = Z, \\ & w \in \mathbb{R}^k, X \succeq 0. \end{aligned}$$

Problem (D) *can be rewritten in standard primal form, namely as*

$$\begin{aligned} \min_{S} \quad & -Z \bullet S + [C \bullet Z] \\ & G_i \bullet S = h_i, \quad i = 1, \ldots, k, \\ & S \succeq 0. \end{aligned}$$

Proof By definition of Z and $G_1 \ldots G_k$, a matrix X satisfies $\mathcal{A}(X) = b$ if and only if

$$X = Z - \sum_{i=1}^{k} w_i G_i,$$

for a certain $w \in \mathbb{R}^k$. This equivalence proves the first statement.
As before, from Z and $G_1 \ldots G_k$, it follows that (y, S) are such that $\mathcal{A}^T(y) + S = C$ if and only if S satisfies

$$G_j \bullet S = h_j, \quad j = 1, \ldots, k.$$

Therefore, also the second statement follows. □

Later on, we denote the primal feasible set with $\mathcal{F}_p$ and the dual feasible set with $\mathcal{F}_d$, that are

$$\begin{aligned} \mathcal{F}_p &= \{X \in \mathbb{S}^n : \ \mathcal{A}(X) = b, X \succeq 0\}, \\ \mathcal{F}_d &= \left\{(y, S) \in \mathbb{R}^m \times \mathbb{S}^n : \ \mathcal{A}^T(y) + S = C, S \succeq 0\right\}. \end{aligned}$$

In addition, we denote with $\mathcal{F}_p^0$ and $\mathcal{F}_d^0$ the interior of the above sets, namely

$$\begin{aligned} \mathcal{F}_p^0 &= \mathcal{F}_p \cap \{X \in \mathbb{S}^n : X \succ 0\}, \\ \mathcal{F}_d^0 &= \mathcal{F}_d \cap \{(y, S) \in \mathbb{R}^m \times \mathbb{S}^n : \ S \succ 0\}. \end{aligned}$$

Finally, we denote with p^* and d^* the primal and the dual optimal values, respectively.

1.2 Duality Theory

As in linear programming, (P) and (D) are by construction closely related as result of the duality theory. The first result is known as weak duality.

Proposition 1.2.1 *For any $X \in \mathcal{F}_p$ and $(y, S) \in \mathcal{F}_d$, the following condition holds*

$$C \bullet X - b^T y = X \bullet S \geq 0.$$

Proof

$$C \bullet X - b^T y = \big(\mathcal{A}^T(y) + S\big) \bullet X - b^T y = y^T \mathcal{A}(X) + S \bullet X - b^T y = S \bullet X \geq 0.$$

□

For any primal feasible solution X and any feasible dual solution (y, S), the quantity $C \bullet X - b^T y$ is called duality gap. If the duality gap is zero, namely strong duality holds, then X and (y, S) are proved to be optimal, respectively for the primal and dual problems.

As opposed to Linear Programming, in Semidefinite Programming it might not hold that duality gap is zero at an optimal solutions pair. Actually, some unpleasant situations can occur:

- $p^* = d^*$, but one of the problem does not admit solution;
- both problems admit solution, but $p^* > d^*$;
- one of the problems admits solution, but $p^* - d^*$ is unbounded.

Actually, it is easy to build small counterexamples of strong duality, as you can see in [59]. Fortunately, we can easily check problems where strong duality necessarily holds: in particular problems where Slater Constraint Qualification (SCQ) holds, namely problems where feasible set has a non-empty interior.

Theorem 1.2.2 *If $\mathcal{F}_p$ and $\mathcal{F}_d^0$ are not empty, then* (P) *admits optimal solution and $p^* = d^*$.*

Proof Let $\hat{X} \in \mathcal{F}_p$ and $(\hat{y}, \hat{S}) \in \mathcal{F}_d^0$. For any optimal solution X^*, $C \bullet X^* \leq C \bullet \hat{X}$, so that

$$\hat{S} \bullet X^* = \left(C - \mathcal{A}^T(\hat{y})\right) \bullet X^* = C \bullet X^* - b^T\hat{y} \leq (C - \mathcal{A}^T(\hat{y})) \bullet \hat{X} = \hat{S} \bullet \hat{X}.$$

Adding the redundant constraint $\hat{S} \bullet X^* \leq \hat{S} \bullet \hat{X}$, another formulation for (P) is obtained, namely

$$\begin{aligned} \min_X \quad & C \bullet X \\ & \mathcal{A}(X) = b \\ & \hat{S} \bullet X \leq \hat{S} \bullet \hat{X} \\ & X \succeq 0. \end{aligned} \tag{P'}$$

The subset $\left\{X \in \mathbb{S}^n : \ \hat{S} \bullet X \leq \hat{S} \bullet \hat{X}, \ X \succeq 0\right\}$ is compact because $\hat{S} \succ 0$. Therefore, (P') admits an optimal solution and hence also (P).

For a fixed $\epsilon > 0$, it is possible to find a feasible dual solution with objective function at least $p^* - \epsilon$. Consider the following two sets

$$\mathcal{F}_1 = \{X \in \mathbb{S}^n : \ X \succeq 0\},$$

$$\mathcal{F}_2 = \{X \in \mathbb{S}^n : \ \mathcal{A}(X) = b, C \bullet X \leq p^* - \epsilon\},$$

which are closed convex sets and also disjoint. As result of the Separating Hyperplane Theorem (see, e.g.,[12]), there exists $(\sigma, S) \in \mathbb{R} \times \mathbb{S}^n$ such that

$$\sup_{X \in \mathcal{F}_2} S \bullet X < \sigma < \inf_{X \in \mathcal{F}_1} S \bullet X. \tag{1.1}$$

Because $\mathcal{F}_1$ is also a cone, the null vector belongs surely to $\mathcal{F}_1$ and hence $\sigma < 0$. In addition it follows that $S \succeq 0$, because otherwise the second problem would be unbounded below. Moreover, because of the first inequality in (1.1), there not exists an $X \in \mathbb{S}^n$ such that

$$\begin{cases} \mathcal{A}(X) = b, \\ C \bullet X \leq p^* - \epsilon, \\ S \bullet X > \sigma. \end{cases}$$

Therefore, as a consequence of a theorem of the Alternative, there exists $(y, \eta) \in \mathbb{R}^{m+1}$ such that

$$\begin{cases} C\eta - \mathcal{A}^T(y) = S, \\ (p^* - \epsilon)\eta - b^T y \leq \sigma, \\ \eta \geq 0. \end{cases}$$

If η were zero then $-b^T y \leq \sigma < 0$, which would be in contradiction with

$$-b^T y = -y^T \mathcal{A}(X) = \mathcal{A}^T(y) \bullet \hat{X} = S \bullet \hat{X} \geq 0.$$

We can so assume $\eta = 1$ by scaling y and S. Finally, we get $(y, S) \in \mathcal{F}_d$, with $b^T \geq p^* - \epsilon - \sigma \geq p^* - \epsilon$. This result and weak duality give

$$p^* - \epsilon \leq b^T y \leq p^*.$$

For the arbitrariness in choosing ϵ, it follows $p^* = d^*$. □

Using the symmetry result given in Lemma 1.1.1, the previous result is commuted from the primal to the dual point of view.

Corollary 1.2.3 *If $\mathcal{F}_d$ and $\mathcal{F}_p^0$ are not empty, then* (D) *admits optimal solution and $p^* = d^*$.*

Of course we can combine the above results.

Corollary 1.2.4 *If $\mathcal{F}_p^0$ and $\mathcal{F}_d^0$ are not empty, then* (P) *and* (D) *both admit optimal solution and $p^* = d^*$.*

In conclusion, for problems where (SCQ) holds for both side, we can define the following necessary and sufficient optimality conditions:

$$\begin{aligned} &\mathcal{A}(X) = b,\ X \succeq 0, \\ &\mathcal{A}^T(y) + S = C,\ S \succeq 0, \\ &XS = 0. \end{aligned} \tag{1.2}$$

The last condition derives from the fact that for feasible points, $C \bullet X - b^T y = X \bullet S$, so that $C \bullet X = b^T y$ if and only if $XS = 0$.
Those optimality conditions are the basis for Interior Points algorithms, used generally for solving Semidefinite Programming problems.

1.3 Interior Point methods

Semidefinite Programming is a particular class of problems in convex optimization, so that SDP problems can be solved in polynomial time to any prescribed precision using for instance the Ellipsoid method (see, e.g., [30]). From the practical point of view,

the Ellipsoid method cannot be used for the prohibitively high running time. In turn, Interior Points methods guarantee polynomial time convergence and are practically efficient at least for small/medium SDP instances.
Interior Point Methods is a very large class of algorithms with many variants developed. For a complete survey refer to the book [39]. In order to give an idea how they work, we describe briefly the primal-dual Path Following subclass, which turns out to be one of the most efficient algorithm among the Interior Point class.

For problems where (SCQ) holds for the primal and dual side, the optimality characterization (1.2) holds. The idea of the path following algorithms is to relax the complementarity slackness condition in (1.2): given the parameter $\mu > 0$, consider the following perturbed optimality system

$$\mathcal{S}_\mu \begin{cases} \mathcal{A}(X) = b, \\ \mathcal{A}^T(y) + S = C, \\ XS = \mu I. \end{cases}$$

The system above admits always a unique solution. Indeed the above conditions can be interpreted as the KKT conditions of a convex problem with strictly convex objective function. If we denote the solution with (X_μ, y_μ, S_μ), we define the following set

$$\{(X_\mu, y_\mu, S_\mu) : \mu > 0\},$$

the central path. It can be shown that the central path is a smooth curve parametrized by μ and that it leads to the optimal primal-dual solution.
Path Following methods generate a sequence of primal-dual solution in the interior of the primal-dual feasible set, close to the central path, so that the optimal solution is reached forcing μ to 0. The way they stay close to the central path relies on applying Newton method to $\mathcal{S}_\mu$.
$\mathcal{S}_\mu$ is not a square system, so that Newton method can not be directly applied. Actually, because the product XS is not symmetric, the system $\mathcal{S}_\mu$ is overdetermined. A possible correction consists in replacing the complementarity condition with an equivalent symmetric one, such as

$$\frac{XS + SX}{2} = \mu I.$$

Given a strictly feasible (X, y, S), the idea is to reduce the violation of the complementarity condition, namely the quantity

$$c(X,S) = \frac{X \bullet S}{n},$$

by getting closer to the point in the central path with $\mu = \sigma c(X,S)$, where $\sigma \in (0,1)$. This is done by computing the Newton direction $(\Delta X, \Delta y, \Delta S)$ on system $\mathcal{S}_\mu$, namely the solution of the following linear system

$$\mathcal{S}_\mu(X,y,S) \begin{cases} \mathcal{A}(\Delta X) = 0, \\ \mathcal{A}^T(\Delta y) + \Delta S = 0, \\ \Delta X S + S \Delta X + \Delta S X + X \Delta S = 2\mu I - (XS + SX). \end{cases}$$

The solution of the above system is uniquely defined if (X, y, S) is sufficiently close to the central path, for example, if (X, y, S) belongs to the following set

$$\mathcal{N}_\gamma = \left\{ (X,y,S) \in \mathcal{F}_p^0 \times \mathcal{F}_d^0 : \ \|X^{\frac{1}{2}} S X^{\frac{1}{2}} - c(X,S) I\| \leq \gamma c(X,S) \right\},$$

with $\gamma \in \left(0, \frac{\sqrt{2}}{2}\right)$.

An outline of the algorithm is given in the following.

primal-dual Path Following algorithm

```
Parameter. $\gamma \in \left(0, \frac{\sqrt{2}}{2}\right)$, $\epsilon > 0$.

Initialization. $(X, y, S) \in \mathcal{N}_\gamma$.

While $c(X,S) > \epsilon$
    1. Choose $\sigma \in (0,1)$ and set $\mu = \sigma c(X,S)$.
    2. Compute $(\Delta X, \Delta y, \Delta S)$ as a solution of $\mathcal{S}_\mu(X,y,S)$.
    3. Choose $\alpha \in (0,1)$ such that
           $X = X + \alpha \Delta X,$
           $y = y + \alpha \Delta y,$
           $S = S + \alpha \Delta S,$
       the updated point belongs to $\mathcal{N}_\gamma$.
End While
```

Different choices for σ and α represent the trade-off between theoretical convergence and practical efficiency. For example, for a given $\delta \in (0,1)$ such that

$$\frac{2\gamma^2 + 2\delta^2}{(1-\sqrt{2}\gamma)^2} \leq \gamma\left(1 - \frac{\delta}{\sqrt{n}}\right),$$

the choice $\sigma = 1 - \frac{\delta}{\sqrt{n}}$ makes the entire sequence strictly feasible even for $\alpha = 1$ and it is possible to show that the Path Following algorithm converges in at most $\mathcal{O}(\sqrt{n}|\log \epsilon|)$ iterations. On the other side, a faster convergence in practice is obtained choosing dynamically σ and setting $\alpha = 0.99\hat{\alpha}$, with

$$\hat{\alpha} = \min\left\{\frac{1}{\max\left(0, -\lambda_{min}(X^{-\frac{1}{2}}\Delta X X^{-\frac{1}{2}})\right)}, \frac{1}{\max\left(0, -\lambda_{min}(S^{-\frac{1}{2}}\Delta S S^{-\frac{1}{2}})\right)}\right\},$$

so that the iterates are at least strictly feasible.

Without going to much in the details, main tasks per iteration include the solution of a linear system of dimension m (whose data need to be computed), forming the directions and the new points, performing two extreme spectral decomposition. So, the cost per iteration is of the order of $\mathcal{O}(mn^3 + m^2n^2 + m^3)$. Even if the data of the SDP problems are sparse, savings are not so much as expected. Actually some operations are sparse independent.

1.4 Geometry

In this section we discuss some important geometric properties of the primal and dual feasible set.
The sets we consider are given by the intersection of affine subspaces with the cone of semidefinite matrices. From the latter they inherit the geometric structure, e.g. the shape of its faces.

Definition 1.4.1 *Given a convex set S, a subset $F \subseteq S$ is a called a face of S if for any two elements $x, y \in S$ such that $\alpha x + (1-\alpha)y \in F$ for some $\alpha \in (0,1)$, then $x, y \in F$.*

The faces of the semidefinite cone can be qualified as the subset of semidefinite matrices whose eigenvectors corresponding to nonzero eigenvalues are restricted to a certain subspace of $\mathbb{R}^n$(see [35]).

Proposition 1.4.2 *Faces of the semidefinite cone fall in one of the following three cases:*

(i) $F = \emptyset$;

(ii) $F = \{0\}$;

(iii) $F = \{X \in \mathbb{S}^n : \; \mathrm{Im}(X) \subseteq \mathrm{Im}(P), \; X \succeq 0\}$.

Proof The first two cases are trivially verified.
For the third case, we first show that F defined in point (*iii*) is a face. Assume by contradiction that F is not a face. Then there exists $Z \in F$ such that

$$Z = \alpha X + (1 - \alpha)Y,$$

with $\alpha \in (0, 1)$, at least $X \notin F$, but with $X, Y \succeq 0$. This means there exists an $v \notin \mathrm{Im}(P)$ such that $Xv \neq 0$. It follows that

$$v \in \mathrm{Im}(P)^{\perp} \subseteq \mathrm{Im}(Z)^{\perp} = \ker(Z).$$

All together we have

$$0 = v^T Z v = \alpha v^T X v + (1 - \alpha) v^T Y v > 0,$$

which is a contradiction.

Now, we show that every face in the semidefinite cone can be rewritten as in (*iii*).
First of all, we observe that 0 belongs to any nonempty face, so that there is only one face with just one element.
Consider a face F with more than one element. In this situation we can find a matrix Z in the relative interior. Given $P \in \mathbb{R}^{n \times k}$ with columns the eigenvectors of Z corresponding to nonzero eigenvalues, we define the following set

$$\hat{F} = \{X \in \mathbb{S}^n : \; \mathrm{Im}(X) \subseteq \mathrm{Im}(P), \; X \succeq 0\},$$

which, from the point before, we already know it is face.
By construction, F and $\hat{F}$ are both faces and they both contain Z in the relative interior. It follows that $F = \hat{F}$. □

It is easy to verify that the characterization of a face in the SDP cone is equivalent to

$$F = \left\{X \in \mathbb{S}^{n\times n} : \; X = PSP^T, \; S \succeq 0, \; S \in \mathbb{S}^k\right\}.$$

Then F is isomorphic to $\mathbb{S}^k$ and hence with dimension $\begin{pmatrix} k+1 \\ 2 \end{pmatrix}$.

An SDP problem is given by the intersection of the semidefinite cone with an affine subspace: faces for $\mathcal{F}_p$ and $\mathcal{F}_d$ can be directly derived from Proposition 1.4.2.

Corollary 1.4.3 *Any face of $\mathcal{F}_p$ is in the form*

$$\{X \in \mathcal{F}_p : \; \mathrm{Im}(X) \subseteq \mathrm{Im}(P)\},$$

for a certain matrix $P \in \mathbb{R}^{n\times k}$.

Any face of $\mathcal{F}_d$ is in the form

$$\{(y, S) \in \mathcal{F}_s : \; \mathrm{Im}(S) \subseteq \mathrm{Im}(Q)\},$$

for a certain matrix $Q \in \mathbb{R}^{n\times r}$.

An important geometric property is related to the rank of a matrix contained in a certain face, as it can not to be too large respect to the dimension of the face (see [50]).

Proposition 1.4.4 *Let F be a face of $\mathcal{F}_p$ of dimension d and $X \in F$ of rank r, then*

$$\begin{pmatrix} r+1 \\ 2 \end{pmatrix} \leq m + d.$$

Let F be a face of $\mathcal{F}_d$ of dimension d and $(y, S) \in F$, with S of rank r, then

$$\begin{pmatrix} r+1 \\ 2 \end{pmatrix} \leq \begin{pmatrix} n+1 \\ 2 \end{pmatrix} - m + d.$$

Proof Let prove the first statement. By rank assumption, we can write $X = QEQ^T$ where $Q \in \mathbb{R}^{n\times r}$ and $E \in \mathbb{S}^r$ with $E \succ 0$. Moreover, feasibility leads to

$$Q^T A_i Q \bullet E = b_i, \quad i = 1, \ldots, m.$$

The system above has m equations in $\frac{r(r+1)}{2}$ variables. Assume by contradiction that $\frac{r(r+1)}{2} > m + d$. Therefore, there exist $\Delta_1, \ldots, \Delta_{d+1} \in \mathbb{S}^r$ linear independent such that

$$Q^T A_i Q \bullet \Delta_j = 0, \quad i = 1, \ldots, m, \; j = 1, \ldots, d+1.$$

Since $E \succ 0$, there exists an $\epsilon > 0$ such that

$$\Delta_{j,1} = E + \epsilon\Delta_j, \ \Delta_{j,2} = E - \epsilon\Delta_j \succeq 0, \quad j = 1, \dots, d+1.$$

Based on these matrices, we define

$$X_{j,1} = Q\Delta_{j,1}Q^T, \ X_{j,2} = Q\Delta_{j,2}Q^T, \quad j = 1, \dots, d+1.$$

By construction, for any $j = 1, \dots, d+1$, $X_{j,1}, X_{j,2} \in \mathcal{F}_p$ and $X = \frac{X_{j,1}+X_{j,2}}{2}$. Therefore, because F is face, $X_{j,1}, X_{j,2} \in F$. Moreover, $\Delta_1, \dots, \Delta_{d+1}$ are linear independent, so that matrices $E, \Delta_{1,1}, \dots, \Delta_{d+1,1}$ and hence also $X, X_{1,1}, \dots, X_{d+1,1}$ are affine independent. Therefore, it follows $\dim(F) \geq d+1$, which is a contradiction.

The second statement follows directly writing down (D) in standard primal form as in Lemma 1.1.1. In this form, the number of equations equals the dimension of $\text{span}(A_1, \dots A_m)^\perp$, namely $\binom{n+1}{2} - m$. So, the second statement holds as result of the first one. □

Corollary 1.4.5 *If Problem* (P) *admits an optimal solution, then there exists an optimal solution X^* such that*

$$\binom{r+1}{2} \leq m,$$

where r is the rank of X^*.

If Problem (D) *admits an optimal solution, then there exists an optimal solution (y^*, S^*) such that*

$$\binom{r+1}{2} \leq \binom{n+1}{2} - m,$$

where r is the rank of S^*.

Proof If an SDP problem admits an optimal solution, then there exits an optimal solution being an extremal matrix, namely belonging to a face with dimension zero. Statements follow from Proposition 1.4.4. □

An extension of the results in Proposition 1.4.4 and Corollary 1.4.5 can be obtained for a more general primal SDP problem (see [50]). In some situations, variables are required to be semidefinite in terms of blocks and it is convenient to make this fact explicit:

$$\begin{aligned} \min_{X_1\in\mathbb{S}^{n_1},\dots,X_p\in\mathbb{S}^{n_p}} \quad & \sum_{j=1}^{p} C_j \bullet X_j \\ & \sum_{j=1}^{p} A_{ij} \bullet X_j = b_i \quad i=1,\dots,m, \\ & X_1,\dots,X_p \succeq 0, \end{aligned} \tag{GP}$$

where data are properly defined.

Proposition 1.4.6 *Let F be a face of the feasible set of problem* (GP)*, with dimension d. For any $(X_1,\dots,X_p)\in F$, it holds*

$$\sum_{j=1}^{p} \binom{r_j+1}{2} \le m+d,$$

where $r_j = \operatorname{rank}(X_j)$, with $j=1,\dots,p$.

Proof As in Proposition 1.4.4, the rank assumption allows to rewrite, for any $j = 1,\dots,p$, $X_i = Q_iE_iQ_i^T$, where $Q_i \in \mathbb{R}^{n\times r}$ and $E_i \in \mathbb{S}^r$ with $E_i \succ 0$. It follows that

$$Q_j^T A_{ij} Q_j \bullet E_j = b_i, \quad i=1,\dots,m, \ j=1,\dots,p.$$

The system above has m equations in $\displaystyle\sum_{j=1}^{p}\frac{r_j(r_j+1)}{2}$ variables. The proof can be completed as in the first part of the proof in Proposition 1.4.4. □

Corollary 1.4.7 *If Problem* (GP) *admits an optimal solution, then there exists an optimal solution $X^* = (X_1^*,\dots,X_p^*)$ such that*

$$\sum_{j=1}^{p} \binom{r_j+1}{2} \le m$$

where $r_j = \operatorname{rank}(X_j^*)$, with $j=1,\dots,p$.

2

Low-Rank formulation of SDP problems

In Chapter 1, we refer to Interior Point methods as the main class of algorithms for Semidefinite Programming. Actually, Interior Point methods are theoretically efficient in solving linear SDP problems: they are proved to converge to an $\epsilon-$optimal solution in polynomial time. Practically, on the one side these methods show high accuracy in reasonable time for small and medium instances. On the other side they are time and memory consuming, so that large instances can not be tackled.

To overcome dimension restrictions, several solution approaches were proposed in literature based on relaxing the theoretical requirements in favor of practical efficiencies. The general recipe is to quit with second-order methods and to exploit just first-order information. Another key point is to try to get rid of semidefinite constraints, as the most challenging part in linear SDP problems.

An efficient approach is the one proposed in [32], where a first-order bundle method is used to solve a restricted class of SDP problems: first, an SDP problem, with an implicit constant trace constraint, is reformulated as an eigenvalue optimization problem and then it is applied a particular bundle method. This eigenvalue reformulation exploits the fact that a matrix is semidefinite positive if and only if the smallest eigenvalue is nonnegative.

In the spirit of bypass semidefinite constraints other approaches rely on some kind of factorization. The first work in this direction goes back to the one of Homer and Peinado

in [36]: in order to solve the SDP relaxation of the Max-Cut problem, it proposed the change of variables $X = VV^T$, where $V \in \mathbb{R}^{n \times n}$. Difficulties of this approach are mainly due to the number of variables (n^2) and also the loss of convexity.
As a way to decrease the dimension of the reformulation, Burer and Monteiro in [15] proposed to use a transformation of variables based on factorizations with triangular matrices.

In [16, 17], Burer and Monteiro improved and extended their approach to general SDP problems, applying the factorization $X = VV^T$ with $V \in \mathbb{R}^{n \times r}$ and $r < n$. This rectangular transformation reduces the number of variables to nr.
Respect to this nonlinear reformulation called Low-Rank, the most important theoretical result is the definition of some global optimality conditions [1, 16, 27], which allows to recognize which stationary points are optimal.
In practice it may happen to compute stationary points several times. Therefore, a key aspect in the Low-Rank approach is how to efficiently compute a stationary point for this nonlinear reformulation. In [14, 16, 17], for solving general linear SDP problems, an Augmented Lagrangian approach is defined, which leads to the solution of a sequence of unconstrained problems. On the other side, looking at a restricted class of SDP problems, it is possible to think of an Exact Penalty approach which allows to get a stationary point of the Low-Rank formulation with just a single unconstrained minimization.

In this chapter we recall properties of the Low-Rank formulation and related solution algorithms.

2.1 Low-Rank formulation of the standard primal SDP problem

For sake of simplicity, we recall the primal SDP problem

$$\begin{aligned} \min_{X} \quad & C \bullet X \\ & \mathcal{A}(X) = b, \\ & X \succeq 0, \end{aligned} \tag{P}$$

and the corresponding dual

$$\begin{aligned} \max_{y} \quad & b^T y \\ & C - \mathcal{A}^T(y) \succeq 0, \\ & y \in \mathbb{R}^m. \end{aligned} \tag{D}$$

Without loss of generality, we assume matrices $A_1 \dots, A_m$ (defining $\mathcal{A}(\cdot)$) being linear independent as well as the fulfillment of the Slater Constraint Qualification for the primal and the dual. By Corollary 1.2.4, these assumptions make both problems admit an optimal solution and the following conditions

$$\begin{aligned} & \mathcal{A}(X) = b, \; X \succeq 0, && \text{(primal feasibility)} \\ & C - \mathcal{A}^T(y) \succeq 0, && \text{(dual feasibility)} \\ & C \bullet X = b^T y, && \text{(zero duality gap)} \end{aligned}$$

be necessary and sufficient for primal-dual optimality.

The idea in [16] is to reformulate (P) using rectangular factorizations. The innovative aspect, respect to square factorizations used in [36], is to consider a factorization which is not valid for all primal feasible points, but simply one that holds at least for an optimal solution of (P). The transformation used is given by $X = VV^T$ with $V \in \mathbb{R}^{n \times r}$, for $r \leq n$. Feasible solutions with large rank are driven out: if $X = VV^T$ then $\text{rank}(X) \leq r$. This is the reason why this approach is referred as Low-Rank formulation for SDP problems.

The matter is to preserve at least an optimal solution. If we knew the minimal rank of an optimal solution, that is

$$r^* = \min\{\text{rank}(X^*) : \; X^* \text{optimal solution for (P)}\},$$

we could safely choose $r = r^*$. Even if r^* is generally unknown, Proposition 1.4.4 provides an upper bound, defined as

$$\hat{r} = \max\left\{ r \in \mathbb{N} : \; \frac{r(r+1)}{2} \leq m \right\} = \left\lfloor \frac{\sqrt{1+8m} - 1}{2} \right\rfloor. \tag{2.1}$$

More generally, for a fixed $r < n$ we consider the following problem

$$\begin{aligned} \min_{V} \quad & C \bullet VV^T \\ & \mathcal{A}(VV^T) = b, \\ & V \in \mathbb{R}^{n \times r}, \end{aligned} \tag{LR$_r$}$$

where the equivalence with (P) is stated in the next proposition.

Proposition 2.1.1 *Given $r \geq r^*$, problems* (P) *and* (LR$_r$) *are equivalent in terms of global optimality.*

Proof First of all, if V is feasible for (LR$_r$), then VV^T is feasible for (P) and with the same objective values. It follows that the optimal value of (LR$_r$) can not be smaller than the optimal value of (P).
Let X^* be an optimal solution for (P) with $\text{rank}(X^*) = r^*$, so that there exists $R^* \in \mathbb{R}^{n \times r^*}$ such that $X^* = R^* R^{*T}$. Then, $V^* = \begin{bmatrix} R^* & 0_{n \times r - r^*} \end{bmatrix}$ is feasible for (LR$_r$) and with objective value equals to $C \bullet X^*$. It follows that the optimal value of (LR$_r$) can not be larger than the optimal value of (P).
In conclusion, optimal values of (P) and (LR$_r$) are equal and optimal solutions can be defined from each other. □

Equivalence is guaranteed, but difficulties come out because (LR$_r$) is nonlinear and generally non-convex problem. Fortunately, on the one side global optimality conditions are available. On the other side, experimental tests in literature for the Low-Rank approach show that first-order nonlinear algorithms (developed for finding simple stationary points) provide almost directly global solutions for (LR$_r$), hence for (P).

Before to recall optimality conditions, we rewrite (LR$_r$) in nonlinear standard form using vectors instead of matrices, namely setting $v = \text{vec}(V)$. To rewrite the objective function and constraints we use properties of the Kronecker product and the vec operator: given $M \in \mathbb{S}^n$, it follows that

$$M \bullet VV^T = MV \bullet V = \text{vec}(MV)^T \text{vec}(V) = ((I_r \otimes M)\text{vec}(V))^T \text{vec}(V) = v^T (I_r \otimes M) v.$$

Therefore, the Low-Rank formulation can be rewritten in this way

$$\begin{aligned} \min_v \quad & v^T (I_r \otimes C) v \\ & v^T (I_r \otimes A_i) v - b_i = 0, \quad i = 1, \dots, m \\ & v \in \mathbb{R}^{nr}. \end{aligned} \tag{LR$_r$}$$

2.2 Optimality conditions

First, we define the Lagrangian function

$$L(v, y) = v^T (I_r \otimes C) v + \sum_{i=1}^m y_i \left(b_i - v^T (I_r \otimes A_i) v \right) = v^T \left(I_r \otimes C - \mathcal{A}^T(y) \right) v + b^T y,$$

where $y \in \mathbb{R}^m$ is the vector of multipliers associated to the constraints.
Second, we recall first and second order KKT conditions for the nonlinear programming problem (LR$_r$): if $\hat{v}$ is regular and is a local minimizer for (LR$_r$), then there exists $\hat{y} \in \mathbb{R}^m$ such that

First-Order KKT conditions (FOC)

$$\begin{aligned} &\left(I_r \otimes C - \mathcal{A}^T(\hat{y})\right) \hat{v} = 0, \\ &\hat{v}^T (I_r \otimes A_i) \hat{v} = b_i, \quad i = 1, \ldots, m. \end{aligned} \tag{2.2}$$

Second-Order KKT conditions (SOC)

$$z^T \left(I_r \otimes C - \mathcal{A}^T(\hat{y})\right) z \geq 0,$$

for any $z \in \mathbb{R}^{nr}$ such that $z^T (I_r \otimes A_i) \hat{v} = 0$, with $i = 1, \ldots, m$.

In the following, we call $\hat{v}$ a stationary point for (LR$_r$) if condition (FOC) is satisfied. If also (SOC) holds then we call $\hat{v}$ a second-order stationary point.

Proposition 2.2.1 *If $\hat{v}$ a stationary point for* (LR$_r$) *with multiplier $\hat{y}$, then*

$$b^T \hat{y} = \hat{v}^T (I_r \otimes C) \hat{v}.$$

Proof Exploiting first condition in (2.2) and feasibility of $\hat{v}$, we have

$$\begin{aligned} 0 = \hat{v}^T \left(I_r \otimes C - \mathcal{A}^T(\hat{y})\right) \hat{v} &= \hat{v}^T \left(I_r \otimes C\right) \hat{v} - \hat{v}^T \left(I_r \otimes \mathcal{A}^T(\hat{y})\right) \hat{v} \\ &= \hat{v}^T \left(I_r \otimes C\right) \hat{v} - \sum_{i=1}^m \hat{y}_i \hat{v}^T \left(I_r \otimes A_i\right) \hat{v} \\ &= \hat{v}^T \left(I_r \otimes C\right) \hat{v} - \sum_{i=1}^m \hat{y}_i b_i, \end{aligned}$$

so that the thesis follows. □

Therefore, if the multiplier $\hat{y}$ associated to a stationary point $\hat{v}$ results to be feasible for (D), then $\hat{v}$ is optimal for (LR$_r$) and $\hat{y}$ is dual optimal for (D), [16]. Truly, this condition is not only sufficient for optimality but also necessary, [27].

Proposition 2.2.2 *Given $r \geq r^*$, a point v^* is a global minimizer for* (LR$_r$) *if and only if v^* is stationary point for* (LR$_r$) *with multiplier y^* and*

$$C - \mathcal{A}^T(y^*) \succeq 0. \tag{2.3}$$

Proof As for the sufficient part, the multiplier y^* is dual feasible for (D) by condition (2.3) and

$$X^* = \mathrm{mat}(v^*)\mathrm{mat}(v^*)^T$$

is primal feasible for (P) because v^* is feasible for (LR$_r$). Moreover, Proposition 2.2.1 proves optimality of X^* and y^* by strong duality, namely

$$b^T y^* = v^{*T}(I_r \otimes C)v^* = C \bullet X^*.$$

It follows optimality of v^* by Proposition 2.1.1.
Let us now prove the necessity part. By assumption v^* solves problem (LR$_r$) and by strong duality between (P) and (D), combined with results in Proposition 2.1.1, there exists a dual feasible solution y^* such that

$$b^T y^* = v^{*T}(I_r \otimes C)v^*. \tag{2.4}$$

Because feasibility of y^* recasts(2.3), it remains only to show that v^* is a stationary point for (LR$_r$) with multiplier y^*. From feasibility of v^* it follows that

$$b^T y^* = \sum_{i=1}^{m} y_i b_i = \sum_{i=1}^{m} y_i v^{*T}(I_r \otimes A_i)v^* = v^{*T}(I_r \otimes \mathcal{A}^T(y^*))v^*. \tag{2.5}$$

Subtracting (2.5) from (2.4) we get

$$v^{*T}(I_r \otimes C - \mathcal{A}^T(y^*))v^* = 0. \tag{2.6}$$

By (2.3) and by properties of Kronecker products, we have also

$$(I_r \otimes C - \mathcal{A}^T(y^*)) \succeq 0,$$

so that equation (2.6) implies that v^* belongs to the Kernel of $(I_r \otimes C - \mathcal{A}^T(y^*))$, namely

$$(I_r \otimes C - \mathcal{A}^T(y^*))v^* = 0.$$

This finally proves that v^* is a stationary point with the right multiplier y^*. □

Observe that an assumption on the regularity of the feasible set of (LR$_r$) is not required, but just conditions on the primal-dual SDP pair: conditions for strong duality for (P) and (D) are sufficient to ensure that any global minimizer of (LR$_r$) is a stationary point.

As we said before, another global optimality condition for (LR$_r$) has been presented: the sufficient side was presented in [16], while the extension as necessary condition was proved in [27].

Proposition 2.2.3 *Given $r \geq r^*$, a point v^* is a global minimizer for* (LR$_r$) *if and only if*

(i) *v^* stationary point for* (LR$_r$) *with multiplier y^*;*

(ii) *the point $\hat{v} \in \mathbb{R}^{n(r+1)}$ defined as*

$$\hat{v} = \begin{pmatrix} v^* \\ 0_n \end{pmatrix},$$

is a second-order stationary point for (LR$_{r+1}$) *with the same multiplier y^*.*

Proof In the sufficient part, for a given vector $w \in \mathbb{R}^n$, we define the vector

$$z = \begin{pmatrix} 0_{nr} \\ w \end{pmatrix}.$$

By construction, for any $i = 1, \dots, m$, z is such that

$$z^T(I_{r+1} \otimes A_i)\hat{v} = \begin{pmatrix} 0_{nr}^T & w^T \end{pmatrix} \begin{pmatrix} I_r \otimes A_i & 0 \\ 0 & A_i \end{pmatrix} \begin{pmatrix} v^* \\ 0_n \end{pmatrix} = 0,$$

so that, because $\hat{v}$ is a second-order stationary point for (LR_{r+1}), we have

$$\begin{aligned} 0 \leq z^T \left(I_{r+1} \otimes C - \mathcal{A}^T(y^*)\right) z &= \begin{pmatrix} 0_{nr}^T & w^T \end{pmatrix} \begin{pmatrix} I_r \otimes C - \mathcal{A}^T(y^*) & 0 \\ 0 & C - \mathcal{A}^T(y^*) \end{pmatrix} \begin{pmatrix} 0_{nr} \\ w \end{pmatrix} \\ &= w^T(C - \mathcal{A}^T(y^*))w \end{aligned}$$

For the generality of w it follows that $C - \mathcal{A}^T(y^*)$ is semidefinite positive. Combining this result with the assumption on v^*, Proposition 2.2.2 ensures that v^* is a global minimum for (LR$_r$).

As the necessary part, by Proposition 2.2.2, if v^* is a global solution for (LR$_r$), then it is a stationary point with multiplier y^*, optimal solution for (D). Hence, we need just to show that $\hat{v}$ is a second-order stationary point for (LR_{r+1}).

Feasibility of v^* forces $\hat{v}$ to be feasible (LR$_{r+1}$), namely

$$\hat{v}^T(I_{r+1} \otimes A_i)\hat{v} = \begin{pmatrix} v^* & 0_n \end{pmatrix} \begin{pmatrix} I_r \otimes A_i & 0 \\ 0 & A_i \end{pmatrix} \begin{pmatrix} v^* \\ 0_n \end{pmatrix} = (v^*)^T(I_r \otimes A_i)v^* = b_i,$$

for any $i = 1, \dots, m$. Likewise

$$\begin{aligned}\left(I_{r+1} \otimes C - \mathcal{A}^T(y^*)\right) \hat{v} &= \begin{pmatrix} I_r \otimes C - \mathcal{A}^T(y^*) & 0 \\ 0 & C - \mathcal{A}^T(y^*) \end{pmatrix} \begin{pmatrix} v^* \\ 0_n \end{pmatrix} \\ &= \begin{pmatrix} \left(I_r \otimes C - \mathcal{A}^T(y^*)\right) \hat{v} \\ 0 \end{pmatrix} = 0,\end{aligned}$$

so that $\hat{v}$ is at least a simple stationary point for (LR_{r+1}). The second-order KKT condition for $\hat{v}$ is given by dual feasibility of y^*, so that also

$$\left(I_{r+1} \otimes C - \mathcal{A}^T(y^*)\right) \succeq 0.$$

□

Finally, we recall a sufficient global condition appeared in [1] for a slightly more general convex SDP problem, which includes as a special case problem (P): any local minimizer $\widehat{V}$ for (LR_r) is global if $\widehat{V}$ is rank deficient, namely if $\mathrm{rank}(\widehat{V}) < r$. Actually, looking at the proof, it turns out that the assumption of $\widehat{V}$ being a local minimizer can be relaxed to satisfying the second-order KKT conditions for (LR_r) problem.

Proposition 2.2.4 *Given $r \leq r^*$, let v^* a second order stationary point for* (LR_r)*:*

1. *if $r < n$ and the vectors*

$$(e_1 \otimes I_n)^T v^*, \dots, (e_r \otimes I_n)^T v^*,$$

are linear dependent, then v^* is a global solution for (LR_r)*;*

2. *if $r = n$ then v^* is a global solution for* (LR_r)*.*

The above proposition is very important because it ensures that when $r = n$ there not exists a second-order stationary point which is not global.

2.3 Solution approach for the Low-Rank formulation

Generally speaking, algorithms for nonlinear programming provide (approximately) simple stationary points with the associated multiplier vector. Therefore, without convexity, there is no guarantee that the provided point solves globally the problem. Fortunately, for problem(LR_r), thanks to Proposition 2.2.2, it is possible to recognize stationary points which are global from those they are not. If the condition (2.3) is satisfied then the global solution for (LR$_r$) provides a solution for the original SDP problem (P). On the other side, if the condition is not true then the current stationary point is not global.

In the latter case Proposition 2.2.3 gives an hint how to escape from it: going from (LR$_r$) to (LR$_{r+1}$), the injection of the computed stationary point can not be a local minimum, so that it can be escaped from. Nevertheless, from the practical point of view, it is better to increase arbitrarily r (not just by one) and to start each minimization from scratch.

An hidden difficulty in the Low-Rank approach is how to choose the initial rank r^0. We remind that equivalence of (LR$_r$) with (P) holds if $r \geq r^*$. The fact that r^* is typically unknown could make think to start from $\hat{r}$ (the upper bound on r^*). By the way $\hat{r}$ is probably much bigger than r^*, so that we would solve a problem much bigger than needed. Moreover, if $r < r^*$, condition (2.3) could never be satisfied by stationary point of (LR$_r$) and then we would never stop too early.

Therefore, the general recipe is to choose the initial r reasonably small and to increase it until it is found a stationary point for the current problem, such that condition (2.3) is satisfied.

More formally, we summarize the Low-Rank approach in the following scheme.

```
Low-Rank Approach for problem (P)

Parameter. $r^0 \in [1, \hat{r}]$, $\beta > 1$.
Initialization. $r = r^0$.
While $r \leq n$
        1. Compute stationary point $\hat{v}$ for (LR$_r$) with multiplier $\hat{y}$.
        2. If $Q - \mathcal{A}^T(\hat{y}) \succeq 0$ then return with $\hat{X} = \mathrm{mat}(\hat{v})\mathrm{mat}(\hat{v})^T$ and $\hat{y}$,
        3. otherwise set $r = \lceil \beta r \rceil$.
End While
```

In the unlikely case we reach $r = n$ without getting an optimal solution, we can use a second-order algorithm (see, e.g., [4]) applied to (LR$_n$), which guarantees convergence to a second-order stationary point and hence to an optimal solution by Proposition 2.2.4.

The only point which needs more explanation is, once r is fixed, how to practically compute a stationary point and the corresponding multiplier.

2.3.1 Augmented Lagrangian approach for general Low-Rank problems

If we consider general SDP problem, and consequently general Low-Rank formulation, the solution approach proposed in [14, 16, 17] is based on the Augmented Lagrangian function. For problem (LR$_r$) this function is defined as

$$L_a(v, y; \epsilon) = v^T(I_r \otimes C)v + \sum_{i=1}^{m} y_i \left(b_i - v^T(I_r \otimes A_i)v\right) + \frac{1}{\epsilon}\sum_{i=1}^{m} \left(b_i - v^T(I_r \otimes A_i)v\right)^2,$$

hence given by adding to the Lagrangian function a term that measures the violation of the constraints scaled by the penalty parameter $\epsilon > 0$.

Motivations for using this function are related to the fact if we knew the multiplier y^* associated to an optimal solution v^*, then there would exist a threshold value $\epsilon^* > 0$ such that

$$v^* = \arg\min_v L_a(v, y^*; \epsilon)$$

for any $\epsilon \in (0, \epsilon^*]$(see [8]). Then a single minimization would be enough to find the stationary point v^*. This result is very important because it mitigates phenomena of ill-conditioning typical in penalty methods (because of the need of forcing the penalty parameter to 0).

Because the optimal multiplier is unknown as well as the threshold value ϵ^*, in the Augmented Lagrangian approach it is generated a sequence $\{y^k, \epsilon^k\}$ converging to (y^*, ϵ^*) and consequently a sequence $\{v^k\}$ where each element is defined as the unconstrained stationary point of

$$\min_v L_a(v, y^k; \epsilon^k).$$

At each iteration, current solution v^k is used to update multipliers and the penalty parameter: if the feasibility has sufficiently increased, then the multiplier y^k is updated as

$$y^{k+1} = y^k + \frac{1}{\epsilon^k}\left(b_i - {v^k}^T(I_r \otimes A_i)v^k\right) \quad i = 1, \ldots, m. \tag{2.7}$$

Otherwise, the penalty parameter is reduced. Update (2.7) stands for one gradient-type iteration to maximize locally the dual function (see [8]).
Under suitable assumptions, $\{v^k\}$ converges to $\hat{v}$ stationary point for (LR_r) and $\{y^k\}$ to the multiplier $\hat{y}$ associated to $\hat{v}$.

Based on works [14, 16, 17] it has been developed a very efficient code called SDPLR. In particular, a key aspect is the use of an Augmented Lagrangian algorithm specialized to the Low-Rank SDP structure. Practical performances are promising and behaves quite well for SDP problems with no very particular structure.

2.3.2 Exact Penalty approach for particular Low-Rank problems

The Augmented Lagrangian approach allows to cope with large dimension and sparsity, but it suffers on the need of solving a sequence of unconstrained problems.
Restricting the range of applications, the properties of a particular problem can be exploited to find in an easier way a stationary point of the Low-Rank formulation. The key point is the possibility to express multipliers in closed-form expression for a stationary point.

From now on, we focus on problems satisfying the following assumption.

Assumption 2.3.1 *Constraints matrices $A_1, \dots, A_m$ satisfy*

$$A_i A_j = 0, \quad i = 1, \dots, m, \ j = i+1, \dots, m.$$

As usual in nonlinear programming, we require the feasible set in (LR_r) being regular, namely with $(I_r \otimes A_1)v, \dots, (I_r \otimes A_m)v$ linear independent for any feasible v.

Proposition 2.3.2 *If $\hat{v}$ is a stationary point for (LR_r), then the associated multiplier $\hat{y}$ can be expressed as a function of $\hat{v}$, namely as*

$$\hat{y}_l = \frac{\hat{v}^T (I_r \otimes A_l C)\hat{v}}{\hat{v}^T (I \otimes A_l^2)\hat{v}} \quad l = 1, \dots, m. \tag{2.8}$$

Proof Regularity on the feasible set of (LR_r) implies that

$$\hat{v}^T (I_r \otimes A_l^2)\hat{v} = \|(I_r \otimes A_l)v\|^2 \neq 0, \quad l = 1, \dots, m.$$

More, exploiting (2.2), for any $l = 1, \dots, m$, we have

$$\begin{aligned} 0 = \hat{v}^T (I_r \otimes A_l)\left(I_r \otimes (C - \sum_{i=1}^m \hat{y}_i A_i)\right)\hat{v} &= \hat{v}^T (I_r \otimes A_l C)\hat{v} - \sum_{i=1}^m \hat{y}_i \hat{v}^T (I_r \otimes A_l A_i)\hat{v} \\ &= \hat{v}^T (I_r \otimes A_l C)\hat{v} - \hat{y}_l \hat{v}^T (I_r \otimes A_l^2)\hat{v}, \end{aligned}$$

so that (2.8) follows. □

Therefore, we can define a multiplier function $y(v)$ in this way

$$y_l(v) = \frac{v^T(I_r \otimes A_l C)v}{v^T(I \otimes A_l^2)v + (v^T(I_r \otimes A_l)v - b_l)^2} \quad l = 1, \ldots, m. \tag{2.9}$$

Function $y(v)$ is continuously differentiable over $\mathbb{R}^{nr}$. Moreover, if $\hat{v}$ is a stationary point for (LR$_r$), then $y(\hat{v})$ gives the right multiplier vector associated to $\hat{v}$.
These properties are the most important to define an Exact Penalty function for (LR$_r$), namely a function whose minimization gives directly a stationary point for (LR$_r$). Indeed, function (2.9) is a simple reduction of the multiplier function proposed in [42, 43] for general regular constrained problems. Assumption 2.3.1 gives an easier form for the multiplier function, simplifying theoretical properties and practical aspects.
As a particular case, we exploit entirely the theory in [42, 43]. Hence, for all propositions that follow we do not report any proof, just refer to [42, 43].

For simplicity, we assume that the primal feasible set in (P) is bounded (or at least the primal optimal solution set). So there exists $\alpha > 0$ such that $I_n \bullet X < \alpha$ for any primal feasible X. Then, we define the function

$$A(v) = \alpha - v^T(I_r \otimes I_n)v,$$

and the open set $\mathcal{A} = \{v \in \mathbb{R}^{nr} : A(v) > 0\}$. By assumption, $\mathcal{A}$ contains the feasible set of (LR$_r$) and its closure is compact.
For a fixed parameter $\epsilon > 0$, we consider the merit function

$$P_\epsilon(v) = v^T(I_r \otimes C)v + \frac{1}{a(v)} \sum_{i=1}^{m} y_i \left(b_i - v^T(I_r \otimes A_i)v\right) + \frac{1}{\epsilon a(v)} \sum_{i=1}^{m} \left(b_i - v^T(I_r \otimes A_i)v\right)^2,$$

defined over $\mathcal{A}$ and the optimization problem

$$\min_{v \in \mathcal{A}} \; P_\epsilon(v). \tag{ULR$_r$}$$

In the following we report the properties of this problem

Proposition 2.3.3 *Given any $\epsilon > 0$ and a feasible point $\hat{v}$, then*

(i) $\mathcal{L}_\epsilon(\tilde{v}) = \{v \in \mathcal{A} : P_\epsilon(v) \le P_\epsilon(\tilde{v})\}$ *is compact and* $P_\epsilon(v)$ *admits a global minimizer in* $\mathcal{L}_\epsilon(\tilde{v})$*;*

(ii) $P_\epsilon(v)$ *is continuously differentiable in* $\mathcal{A} \supseteq \mathcal{L}_\epsilon(\tilde{v})$*.*

Proposition 2.3.4 *There exists an $\epsilon^* > 0$ such that for any $\epsilon \in (0, \epsilon^*]$ the following equivalences hold:*

(i) $\hat{v}$ *is stationary point for* (LR_r) *if and only if* $\hat{v}$ *is a stationary point for* (ULR_r).

(ii) $\hat{v}$ *is local minimizer for* (LR_r) *if and only if* $\hat{v}$ *is a local minimizer for* (ULR_r).

(iii) $\hat{v}$ *is global minimizer for* (LR_r) *if and only if* $\hat{v}$ *is a global minimizer for* (ULR_r).

Proposition 2.3.4 states the equivalence between (LR_r) and (ULR_r) for ϵ sufficiently large and so (ULR_r) can be used instead of (LR_r). In this sense we refer $P_\epsilon(v)$ as an Exact Penalty function for (LR_r).
Consider standard first-order unconstrained algorithms, such that the generated sequence belongs to level set of the initial point. These algorithms can be used to find a stationary point for (LR_r): properties in Proposition 2.3.3 make any standard unconstrained optimization algorithms applied to (ULR_r) be globally convergent at least to a stationary point for (ULR_r).

In conclusion, we underline a nice property of Low-Rank algorithms. As first-order algorithms use only function and gradient evaluations, Low-Rank approaches are able to fully exploit sparsity of the data. Actually, given any $M \in \mathbb{S}^n$ with k nonzero entries, the matrix $(I_r \otimes M)$ has exactly rk nonzero entries. Hence costs to compute of $(I_r \otimes M)v$ and $v^T(I_r \otimes M)v$ are both of the order of rk.

3

Low-Rank SDP Heuristic for large-scale Max-Cut

The Max-Cut is one of the most important in Graph Partitioning problems and it is a good example of difficult problem in Combinatorial Optimization. Main applications of Max-Cut come out from Circuit Design Problems and Statistical Physics (see [5]). For a detailed survey on Max-Cut refer to [51].
In the Max-Cut the aim is to partition the nodes of a graph in two subsets such that the sum of the weights of the edges crossing from one subset to the other is maximized.

Main effective solution approaches for Max-Cut are based on an SDP relaxation. Actually, producing efficiently a solution to this SDP problem is then of great interest for solving exactly the Max-Cut or for designing good heuristics.
One possibility is to reformulate this SDP problem through Low-Rank factorizations as a nonlinear programming problem. This approach allows to solve large instance in a reasonable time.

In this chapter, we recall the general integer formulation, the SDP relaxation and its Low-Rank Formulation. Then, after giving a fast overview of the Low-Rank approaches proposed in literature, we define a new unconstrained formulation and the corresponding algorithm to solve the SDP relaxation. Afterwards, based on this fast large-scale algorithm, we define a new heuristic algorithm to find good feasible solutions to large instances of Max-Cut. Finally, in last section we report numerical results on the solution of the SDP relaxation and on the approximated solution of huge instances of Max-Cut.

3.1 Max-Cut

Consider an edge-weighted undirected graph $\mathcal{G}(N, \mathcal{A})$, where N is the set of nodes, with cardinality n, and $\mathcal{A} \subseteq N \times N$ is the set of edges between nodes, with cardinality m. The connectivity of the graph is represented by the weighted adjacency matrix $A \in \mathbb{S}^n$: if $ij \in \mathcal{A}$ then a_{ij} is equal to weight associated to the edge ij, otherwise a_{ij} is 0.

Any subset $S \subseteq N$ induces a bipartition in the graph, namely $\{S, N - S\}$. We denote by $\delta(S)$ the cut defined by S, namely the set of edges with an endpoint in S and the other in $N - S$. So the Max-Cut problem can be formulated in this way

$$m^* = \max_{S \subseteq N} \sum_{ij \in \delta(S)} a_{ij}. \tag{MC}$$

In order to have an algebraic formulation of this problem, we represent a bipartition with a vector $x \in \{-1, 1\}^n$, where each x_i is equal to 1 if $i \in S$, -1 otherwise. For a fixed partition x, the vector $z(x) \in \mathbb{R}^m$, defined as

$$z(x) = \left[\frac{1 - x_i x_j}{2}\right]_{ij \in \mathcal{A}},$$

gives the incidence vector of the cut $\delta(x)$. So the weighted sum of the edges in the cut induced by x can be written in this way

$$\begin{aligned}\sum_{ij \in \delta(x)} a_{ij} &= \frac{1}{2}\sum_{i=1}^{n}\sum_{j=1}^{n} a_{ij} z_{ij}(x) = \frac{1}{4}\sum_{i=1}^{n}\sum_{j=1}^{n} a_{ij}(1 - x_i x_j) \\ &= \frac{1}{4}\sum_{i=1}^{n}\left(\sum_{j=1}^{n} a_{ij} x_i x_i - \sum_{j=1}^{n} a_{ij} x_i x_j\right) = \frac{1}{4} x^T \left(\mathrm{Diag}(Au) - A\right) x.\end{aligned}$$

The matrix $L = \mathrm{Diag}(Au) - A$ is called Laplacian of G. Finally, the Max-Cut can be rewritten as an integer quadratic problem

$$\begin{aligned} m^* = \max_{x} \quad & \tfrac{1}{4} x^T L x \\ & x \in \{-1, 1\}^n. \end{aligned} \tag{MC}$$

In order to tackle this $\mathcal{NP}-$Hard problem, several approaches have been developed. Roughly speaking, we can separate them in exact and heuristic algorithms.
Exact algorithms are based on some pseudo-enumeration techniques, with the growth of the number of subproblems limited by computation of lower and upper bounds on the Max-Cut Problem. Upper bounds are obtained by the definition and the solution of a suitable relaxation for (MC) (or some equivalent formulations), while lower bounds

are given by finding "good" cut. An exact algorithm stops as soon as the best lower bound equals the best upper bound and it outputs an optimal solution of (MC). Of course, as consequence of the implicit workload, the dimension of the problems that can be effectively solved is limited to few hundreds.
On the opposite, heuristic algorithms provide only feasible cuts, obtained by applying some heuristic approach (such as evolutionary, simulated annealing, tabu and local search) to the simple structure of bi-partitions. On the one side these algorithms can tackle large instances and on the other side there is no safeguard on the quality of the provided solutions. For a reference of these approaches see [11].
In truth, among the heuristic approaches, there are some based on some relaxation of the Max-Cut, so that a lower and an upper bound are both provided, namely a bound on the optimality error for the generated solutions. Of course, exact algorithms, if interrupted at an early stage, provide an heuristic cut and as a side product an upper bound on the optimal value.

First exact approaches proposed for Max-Cut were based on Linear Programming relaxations. In particular, the polyhedral relaxation based approach in [41] seems to work quite well for very sparse instances.
More recently, Semidefinite Programming came at hand as more effective relaxation for the Max-Cut. In this direction the most significant work is the one of Goemans and Williamson in 1995, [24], where it has been developed an approximation algorithm based on a SDP relaxation. The relevance of their work derives from two facts: first, they drastically improved the best known approximation for the Max-Cut, from 0.5 to 0.87856. Second, they put a light of interest in Semidefinite Programming, as a useful tool in Combinatorial Optimization.

Next, we recall the key points in the approach proposed by Goemans and Williamson, the SDP relaxation, the heuristic procedure and the relative performance guarantee.
Given $x \in \{-1, 1\}^n$ the matrix $X = xx^T$ is semidefinite positive and with diagonal entries equal to 1. In the space of matrices the objective function can be written as

$$x^T Lx = L \bullet xx^T = L \bullet X.$$

Dropping the implicit rank one constraint we get the following SDP relaxation for the Max-Cut

$$\begin{aligned} m' = \max_X \quad & \tfrac{1}{4} L \bullet X \\ & \operatorname{diag}(X) = u, \\ & X \succeq 0, \end{aligned} \qquad (\mathrm{SDP}_{MC})$$

so that $m' \geq m^*$. This SDP problem and the relative dual admit strict feasible points, so they both admit optimal solution and with equal optimal values. Therefore, an approximated solution can be obtained in polynomial time by Interior Point methods. The randomized approximation algorithm proposed in [24] can be summarized in the following steps.

Goemans-Williamson algorithm (`GWA`)

Data. L.

1. *SDP Relaxation*: compute optimal solution X^* for (SDP$_{MC}$).
2. *Gramiam Matrix*: factorize X^* with $V = [v_1 \dots v_n] \in \mathbb{R}^{r \times n}$ with $r \leq n$, such that $X^* = V^T V$.
3. *Random Hyperplane*: generate randomly $h \in \mathbb{R}^r$ with $\|h\| = 1$.
4. *Rounding*: set x as

$$x_i = \begin{cases} 1 & \text{if } h^T v_i \geq 1, \\ -1 & \text{otherwise,} \end{cases} \qquad i = 1, \dots, n.$$

End For

Return x, $z_{lb} = \frac{x^T L x}{4}$ and $z_{ub} = m'$.

The hyperplane defined by h separates the space in two half-spaces and an edge ij belongs to the $cut(x)$ if v_i and v_j belong to different half-spaces. Randomness of h implies randomness of x and of the corresponding cut. Therefore, we need to refer about expected value for the cut. Goemans and Williamson showed that their randomized algorithm produces a cut with an expected value not too far from the optimal one: if $A \geq 0$ then

$$m^* \geq \frac{1}{4} E[x^T L x] \geq \alpha_0 m' \geq \alpha_0 m^*,$$

where $\alpha_0 = 0.87856$. We stress that the above result just holds for graphs with non-negative weights. More generally, graphs with weights such that the Laplacian L is semidefinite positive, α_0 reduces to $\frac{2}{\pi}$ (see [46]).

Problem (SDP$_{MC}$) and `GWA` form the basis for other SDP approaches for the Max-Cut. For example, in [31, 53], exact algorithms are defined using these two ingredients, where

(SDP$_{MC}$) is even strengthened by adding triangle inequalities, directly to the feasible set or to the objective function in a dual Lagrangian fashion. Of course these approaches can not tackle large-instances: the solution of (SDP$_{MC}$) and the computation of the Gramiam matrix represent bottlenecks of the method. Actually, (SDP$_{MC}$) can be solved efficiently by Interior Point methods just for small and medium instances. Moreover, the Gramiam matrix is computed through a Cholesky factorization, which has a cost proportional to $\mathcal{O}(n^3)$. A lot of effort has been done in literature in order to make lighter these two tasks.

3.2 SDP relaxation

To be slightly more general, we consider the SDP relaxation of the Max-Cut in this form

$$\begin{aligned} \min_{X} \quad & C \bullet X \\ & \operatorname{diag}(X) = u, \\ & X \succeq 0. \end{aligned} \tag{SDP}$$

Actually (SDP$_{MC}$) corresponds to (SDP) letting $C = -\frac{1}{4}L$.
The dual of (SDP) is given by

$$\begin{aligned} \max_{y} \quad & u^T y \\ & C - \operatorname{Diag}(y) \succeq 0 \\ & y \in \mathbb{R}^n. \end{aligned} \tag{DSDP}$$

Observe that $X = I$ is strictly feasible for the primal and $y = (\lambda_{min}(C) - 1)u$ is strictly feasible for the dual. Therefore, thanks to Corollary 1.2.4, (SDP) and (DSDP) both admit optimal solution and strong duality holds.

Under those conditions, Interior Point methods are the natural candidates to solve (SDP). Unfortunately, these methods typically become less attractive for instances with n very large: those that make use of an explicit representation of the matrix X, because of the evident excessive space requirements; those that are based on iterative methods (usually based on the dual), because are still too much slow for large instances.

These limitations have motivated searching for methods that are less demanding in terms of memory allocation and of computational cost. In this perspective, the constraint structure of problem (SDP) has been exploited in the literature to define special purpose algorithms.

One possibility is to reformulate (DSDP) as an eigenvalue optimization problem. Actually, for any primal feasible solution X the trace is equal to n, so that adding this redundant constraint and passing to dual we get

$$\begin{aligned} \max_{y,\lambda} \quad & n\lambda + u^T y \\ & C - \mathrm{Diag}(y) - \lambda I \succeq 0, \\ & y \in \mathbb{R}^n, \lambda \in \mathbb{R}, \end{aligned}$$

which is equivalent to

$$\max_{y,\lambda} n\lambda_{min} \left(C - \mathrm{Diag}(y)\right) + u^T y.$$

The objective function is concave, non-smooth and with a particular structure. To maximize this function a Spectral Bundle method was designed in [32]. This algorithm involves per iteration an extreme spectral decomposition of a (sparse) matrix and the solution of a small quadratic SDP problem. As usual in bundle methods, the algorithm is very fast at the beginning and it suffers asking for high accuracy. We refer to the inherent algorithm with `SB`.

The other option is based on the Low-Rank approach as described in Chapter 2. Nonlinear formulations are obtained by means of rectangular factorizations.

3.3 Low-Rank SDP relaxation

With a look to the Max-Cut, we think of factorizations with $r \times n$ matrices instead of $n \times r$. All the results given in Chapter 2 continue to hold, just the form is slightly different.
Every $X \succeq 0$ with rank $r > 0$ can be factorized with $V \in \mathbb{R}^{r\times n}$, that is $X = V^T V$.
Under this notation and applying the vector representation $v = \mathrm{vec}(V)$, we get the Low-Rank formulation for (SDP)

$$\begin{aligned} \min_{v} \quad & v^T (C \otimes I_r) v \\ & v^T (E_{ii} \otimes I_r) v = 1 \quad i = 1, \ldots, n \\ & v \in \mathbb{R}^{nr}. \end{aligned} \tag{LR$_r$}$$

Originally, Goemans and Williamson in [24] were the first proposing this formulation as relaxation of (MC) and then they exploited the equivalence with (SDP) to compute a solution. They did the way round because at that time the convex problem (SDP) seemed to be of easier solution than the nonlinear non-convex problem (LR$_r$).

Main improvements respect to that time were brought by Burer and Monteiro in [16], where they gave a better understanding of the Low-Rank approach for general linear SDP problem. In particular, one of the results implies the equivalence of (LR$_r$) with (SDP) just for r reasonably large (see Proposition 2.1.1).
Moreover, although problem (LR$_r$) is non-convex, they provided sufficient conditions for stationary points to be global, which they turned out to be also necessary conditions, see [27, 29]. We refer to global optimality conditions given in Propositions 2.2.2 and 2.2.3.
In particular, a point $\hat{v}$ is stationary point of problem (LR$_r$) if there exists a Lagrange multiplier $\hat{y} \in \mathbb{R}^n$ such that

$$\begin{aligned} &(C - \mathrm{Diag}(\hat{y}) \otimes I_r)\, \hat{v} = 0, \\ &\hat{v}^T (E_{ii} \otimes I_r) \hat{v} = 1 \quad i = 1, \dots, n \end{aligned} \tag{3.1}$$

In [29], based on the above conditions, it is provided a multiplier function $y(v)$ defined as

$$y_i(v) = v^T (E_{ii} C \otimes I_r), v \quad i = 1, \dots, n, \tag{3.2}$$

such that it gives the right Lagrange multiplier for a stationary point. This fact can be also derived specializing results in Proposition 2.3.2.
Finally, as direct result of Proposition 2.2.2, a point v^* is a global minimizer for (LR$_r$) if and only if v^* is stationary point for (LR$_r$) and $y(v^*)$ is dual feasible for (DSDP).

In the spirit of Low-Rank approach, (LR$_r$) is solved in terms of stationary points for increasing value of r, until the related multiplier is dual feasible. The initial rank is never chosen larger than

$$\hat{r} = \left\lfloor \frac{\sqrt{8n+1} - 1}{2} \right\rfloor,$$

as it represents an upper bound on the minimal rank of an optimal solution of (SDP). This bound derives from the general bound given in (2.1).
More formally, all these results lead to the Incremental Rank Algorithm (IRA) for the solution of problem (SDP), that encompasses most of the Low-Rank methods proposed in the literature.

```
Incremental Rank Algorithm (IRA)

Data. $C$, integer $p$.

Initialization. set integers $2 \leq r^1 < r^2 < \cdots < r^p$, with $r^p \in [\hat{r}, n]$.

For $i = 1, \ldots, p$
        1. Set $r = r^i$.
        2. Compute stationary point $\hat{v}$ for (LR$_r$).
        3. Compute smallest eigenvalue $\hat{\lambda}_{min}$ of $C - \text{Diag}(y(\hat{v}))$.
        4. If $\hat{\lambda}_{min} \geq 0$ then exit.
End For

Return $\hat{\lambda}_{min}$, $\hat{V} = \text{mat}(\hat{v})$, $\hat{X} = \hat{V}^T\hat{V}$, $\hat{y} = y(\hat{v}) + \min(0,\ \hat{\lambda}_{min})u$, $m^{''} = u^T\hat{y}$.
```

The (IRA) algorithm always outputs a primal feasible solution $\hat{X}$, its Gramian matrix $\hat{V}$ and a dual feasible solution $\hat{y}$. If $\hat{\lambda} \geq 0$ then $\hat{X}$ and $\hat{y}$ are respectively optimal for (SDP) and (DSDP), and hence $m^{''} = m^{'}$. Otherwise, $\hat{y}$ remains by construction dual feasible and then $m^{''} < m^{'}$ by weak duality. It follows that at least the (IRA) algorithm provides a lower bound on (SDP). In truth, in practice it always happens to find optimal solutions.

Generally speaking, all the Low-Rank methods proposed in literature for solving (SDP) vary just on the way a stationary point for (LR$_r$) is computed, generally by means of an unconstrained reformulation. We give a brief overview of the different approaches, focusing more on the Quotient formulation because it represents the starting point for our work.

The first method relies on the Augmented Lagrangian approach proposed in [14, 16, 17] for general Low-Rank problems and explained in Subsection 2.3.1: based on

$$L_a(v, y; \epsilon) = v^T(C \otimes I_r)v + \sum_{i=1}^{n} y_i \left(1 - v^T(E_{ii} \otimes I_r)v\right) + \frac{1}{\epsilon} \sum_{i=1}^{n} \left(1 - v^T(E_{ii} \otimes I_r)v\right)^2,$$

several unconstrained minimizations, for different values of the multiplier y and the penalty parameter ϵ, are required in order to get a stationary point of (LR$_r$). We refer to the inherent algorithm with `SDPLR`.

In [29] the structure of the problem is exploited to define the multiplier function $y(v)$ given in (3.2). The use of this function in the Augmented Lagrangian allows to define the merit function

$$P(v,\epsilon) = v^T(C \otimes I_r)v + \sum_{i=1}^{n} v^T(E_{ii}C \otimes I_r)v\left(1 - v^T(E_{ii} \otimes I_r)v\right) + \frac{1}{\epsilon}\sum_{i=1}^{n}\left(1 - v^T(E_{ii} \otimes I_r)v\right)^2,$$

which has been proved to be exact respect to (LR$_r$) for ϵ sufficiently small. Therefore, just an unconstrained minimization of a continuously differentiable function is required. We refer to the inherent algorithm with `EXPA`.

More recently, in [1] it has been proposed a trust region method for the optimization over a manifold, which relies on a particular quotient manifold. We refer to the inherent algorithm with `GenRTR`.

3.3.1 Quotient formulation

This formulation heavily exploits the simple structure of the constraints and the shape of the objective function. It is useful to express explicitly the n vectors $v_1, \ldots, v_n \in \mathbb{R}^r$ which compose the vector $v \in \mathbb{R}^{nr}$, so that the problem becomes

$$\begin{aligned} \min_v \quad & f_r(v) = \sum_{i=1}^{n}\sum_{j=1}^{n} c_{ij} v_i^T v_j \\ & \|v_i\|^2 = 1 \quad i = 1, \ldots, n. \end{aligned} \tag{LR$_r$}$$

Under this notation, respect to the single blocks, we rewrite the KKT conditions as

$$\begin{aligned} \sum_{j=1}^{n} c_{ij} v_j - y_i(v) v_i = \sum_{j=1}^{n} c_{ij}\left(v_j - v_i^T v_j v_i\right) = 0, \quad & i = 1, \ldots, n \\ \|v_i\|^2 = 1 \quad & i = 1, \ldots, n \end{aligned}$$

where it is explicitly used the closed form expression for the multiplier, namely

$$y_i(v) = \sum_{j=1}^{n} q_{ij} v_i^T v_j \quad i = 1, \ldots, n.$$

The idea in this formulation is to force the feasibility of $v_1 \ldots, v_n$: by direct normalization of these vectors on $f_r(v)$, the following function is obtained

$$q_r(v) = \sum_{i=1}^{n}\sum_{j=1}^{n} c_{ij} \frac{v_i^T v_j}{\|v_i\|\|v_j\|}.$$

Based on the so-called Quotient function, there have been some interesting works and it represents also the starting point for ours. The first time of an unconstrained model based on $q_r(v)$ came out in [36] for $r = n$, but the dimension of the resulting problem makes the method prohibitive for large-scale problems. Later on, in [16] function $q_r(v)$ has been proposed again, but in a Low-Rank framework. We refer to the inherent algorithm with `SDPLR-MC`. On the one side, numerical results presented in [16] were quite promising, but, on the other side, some theoretical and practical issues were not considered.

As initial part of our work we deeply analyzed the Quotient function as a basis to solve (LR$_r$). First, to justify the use of the Quotient function, it should be proved the equivalence with (LR$_r$). In this sense, in [28] we proved such correspondence, in terms of stationary points, local and global minimizers.
Nevertheless, practical pitfalls reside in the application of an optimization method to the Quotient function. Generally speaking, a standard unconstrained procedure can be proven to be globally convergent if it is applied to the minimization of a continuously differentiable function with compact level sets. As opposed, $q_r(v)$ is defined only over

$$\mathcal{Q} = \{v \in \mathbb{R}^{nr} : \|v_i\| \neq 0,\ i = 1, \dots, n\}.$$

A workaround for this problem is to restrict to the class of gradient-type methods. Function $q_r(v)$ is continuously differentiable over $\mathcal{Q}$, with gradient components

$$\nabla_{v_i} q_r(v) = \frac{2}{\|v_i\|} \left[\sum_{j=1}^{n} c_{ij} \left(I_r - \frac{v_i}{\|v_i\|} \frac{{v_i}^T}{\|v_i\|} \right) \frac{v_j}{\|v_j\|} \right], \quad i = 1, \dots, n,$$

For the gradients the following property holds

$$v_i^T \nabla_{v_i} q_r(v) = \frac{2}{\|v_i\|} \left[\sum_{j=1}^{n} q_{ij} \left(\frac{v_j^T v_i}{\|v_j\|} - \frac{{v_i}^T v_j}{\|v_i\| \|v_j\|} \frac{{v_i}^T v_i}{\|v_i\|} \right) \right] = 0, \quad i = 1, \dots, n. \tag{3.3}$$

In a gradient-type step for a point $v \in \mathcal{Q}$, the next point v^+ is computes as

$$v_i^+ = v_i - \alpha \nabla_{v_i} q_r(v), \quad i = 1, \dots, n,$$

with a suitable $\alpha > 0$. The orthogonality condition in (3.3) gives

$$\|v_i^+\|^2 = \|v_i^+\|^2 + \alpha^2 \|\nabla_{v_i} q_r(v)\|^2 - 2\alpha {v_i}^T \nabla_{v_i} q_r(v) = \|v_i\|^2 + \alpha^2 \|\nabla_{v_i} q_r(v)\|^2 \geq \|v_i\|^2, \tag{3.4}$$

for any $i = 1, \dots, n$. This condition shows that a gradient-type method keeps the entire sequence in $\mathcal{Q}$. As opposed, with a Newton-type step, with $d = -H \nabla_v q_r(v^k)$, it should be forced the following condition to be true

$$\alpha^2 \|d_i\|^2 - 2\alpha v_i^T d_i \geq 0 \quad i = 1, \dots, n.$$

Unfortunately another problem is left: in order to prove convergence of an unconstrained standard algorithm, it is required that the entire sequence belongs to a certain compact set, so that the sequence admits at least an accumulation point. In the minimization of the Quotient function $q_r(v)$, the sequence generated is very probably to be unbounded (look at (3.4)).

In conclusion, we feel that convergence of an unconstrained method applied to the Quotient function cannot be guarantee. Nevertheless, this approach appeared to be very efficient in practice, so that we propose a regularization of the Quotient function that retains efficiency and allows to prove global convergence for any standard algorithm.

3.3.2 Regularized Quotient function

We modify the objective function $q_r(v)$ adding the regularization term

$$\sum_{i=1}^{n} \frac{(\|v_i\|^2 - 1)^2}{d(v_i)},$$

where the term

$$d(v_i) = \delta^2 - \left(1 - \|v_i\|^2\right)_+^2, \quad 0 < \delta < 1, \tag{3.5}$$

acts as a shifted barrier on the open set

$$\mathcal{S}_\delta = \left\{ v \in \mathbb{R}^{nr} \,:\, \|v_i\|^2 > 1 - \delta, \quad i = 1, \dots, n \right\}.$$

For fixed $\epsilon > 0$ and $r \geq 1$, we consider the unconstrained minimization problem

$$\min_{v \in \mathcal{S}_\delta} \quad r_\epsilon(v) = q_r(v) + \frac{1}{\epsilon} \sum_{i=1}^{n} \frac{(\|v_i\|^2 - 1)^2}{d(v_i)} \quad . \tag{QLR$_r$}$$

Both r_ϵ and $\mathcal{S}_\delta$ depend on r, but we omit the explicit indication of this dependency to simplify notation. Moreover, we use $\mathcal{F}$ to denote the feasible set of problem (LR$_r$), i.e.

$$\mathcal{F} = \left\{ v \in \mathbb{R}^{nr} \,:\, \|v_i\|^2 = 1, \ i = 1, \dots, n \right\}.$$

Observe that $\mathcal{F} \subseteq \mathcal{S}_\delta \subseteq \Omega$.

We start by investigating the theoretical properties of r_ϵ. This function is continuously differentiable on the open set $\mathcal{S}_\delta$, with gradient components

$$\nabla_{v_i} r_\epsilon(v) = \nabla_{v_i} q_r(v) + \frac{4}{\epsilon} \frac{(\|v_i\|^2 - 1)}{d(v_i)} \left[1 - \frac{(\|v_i\|^2 - 1)(1 - \|v_i\|^2)_+}{d(v_i)} \right] v_i.$$

The first important property is the compactness of the level sets of function r_ϵ, which guarantees the existence of a solution for problem (QLR$_r$).

Proposition 3.3.1 *Given $\epsilon > 0$ and $v^0 \in \mathcal{F}$, the level set*

$$\mathcal{L}_\epsilon(v^0) = \left\{v \in \mathcal{S}_\delta : \; r_\epsilon(v) \leq r_\epsilon(v^0)\right\}$$

is compact and

$$\mathcal{L}_\epsilon(v^0) \subseteq \left\{v \in \mathbb{R}^{nr} : \|v_i\|^2 \leq \left(2\overline{C}\epsilon\delta^2\right)^{\frac{1}{2}} + 1, \quad i = 1, \dots, n\right\}, \tag{3.6}$$

with $\overline{C} = \sum_{i=1}^{n}\sum_{j=1}^{n} |c_{ij}| > 0.$

Proof First, for every $v \in \mathcal{S}_\delta$, we have that

$$q_r(v) \;=\; \sum_{i=1}^{n}\sum_{j=1}^{n} c_{ij}\frac{v_i^T v_j}{\|v_i\|\|v_j\|} \geq -\sum_{i=1}^{n}\sum_{j=1}^{n} |c_{ij}|\frac{|v_i^T v_j|}{\|v_i\|\|v_j\|} \geq -\overline{C}.$$

Hence we get

$$r_\epsilon(v) \geq -\overline{C} + \frac{1}{\epsilon}\frac{(\|v_i\|^2 - 1)^2}{\delta^2}, \quad i = 1, \dots, n. \tag{3.7}$$

For every given $v \in \mathcal{L}_\epsilon(v^0)$, as $v^0 \in \mathcal{F}$, it follows

$$r_\epsilon(v) \leq r_\epsilon(v^0) = q_r(v^0) \leq \overline{C},$$

so that using (3.7) it holds

$$\|v_i\|^2 \leq \left(2\overline{C}\epsilon\delta^2\right)^{\frac{1}{2}} + 1 \quad i = 1, \dots n.$$

This implies that (3.6) holds and hence $\mathcal{L}_\epsilon(v^0)$ is bounded.
On the other hand, any limit point $\hat{v}$ of a sequence of points $\{v^k\}$ in $\mathcal{L}_\epsilon(v^0)$ cannot belong to the boundary of $\mathcal{S}_\delta$. Indeed, if $\|\hat{v}_i\|^2 = 1 - \delta$ for some i, then (3.5) implies $d(\hat{v}_i) = 0$ and hence $\lim_{k\to\infty} r_\epsilon(v^k) = \infty$. This fact contradicts $v^k \in \mathcal{L}_\epsilon(v^0)$ for k sufficiently large. Therefore, the level set $\mathcal{L}_\epsilon(v^0)$ is also closed and the claim follows. □

The following theorem states the correspondence between (QLR$_r$) and (LR$_r$) in terms of stationary points, local and global minimizers. This result exploits an interesting property of the Quotient function, the orthogonality condition expressed by (3.3).

Theorem 3.3.2 *Given $\epsilon > 0$ and $r \geq 1$, the following correspondences hold:*

(i) *$\hat{v}$ is a stationary point of* (QLR$_r$) *if and only if $\hat{v}$ is a stationary point of* (LR$_r$).

(ii) *$\hat{v}$ is a global minimizer of* (QLR$_r$) *if and only if $\hat{v}$ is a global minimizer of* (LR$_r$).

(iii) $\hat{v}$ *is a local minimizer of* (QLR$_r$) *if and only if* $\hat{v}$ *is a local minimizer of* (LR$_r$).

Proof First, for every $v \in \mathcal{S}_\delta$, by definition of $\nabla_{v_i} r_\epsilon(v)$ and by (3.3), we get

$$v_i^T \nabla_{v_i} r_\epsilon(v) = \frac{4}{\epsilon} \frac{(\|v_i\|^2 - 1) v_i^T v_i}{d(v_i)} \left(1 - \frac{(\|v_i\|^2 - 1)(1 - \|v_i\|^2)_+}{d(v_i)}\right).$$

for every $i = 1, \dots, n$. Therefore, if $\|v_i\|^2 \geq 1$ then

$$v_i^T \nabla_{v_i} r_\epsilon(v) = \frac{4}{\epsilon} \frac{(\|v_i\|^2 - 1)\|v_i\|^2}{\delta^2}, \tag{3.8}$$

otherwise

$$v_i^T \nabla_{v_i} r_\epsilon(v) = \frac{4}{\epsilon} \frac{(\|v_i\|^2 - 1)\|v_i\|^2}{d(v_i)} \left(1 + \frac{(\|v_i\|^2 - 1)^2}{d(v_i)}\right). \tag{3.9}$$

Furthermore, if $v \in \mathcal{F}$ then

$$\begin{aligned} r_\epsilon(v) &= f_r(v), && (3.10)\\ \nabla_{v_i} r_\epsilon(v) &= 2\sum_{j=1}^{n} c_{ij} \left(I_r - v_i {v_i}^T\right) v_j = \nabla_{v_i} L\left(v, y(v)\right), \quad i = 1, \dots, n. && (3.11) \end{aligned}$$

Now we prove the correspondences stated in our claims.

(*Correspondence of stationary points*).

Necessity. By (3.8) and (3.9), if $\hat{v} \in \mathcal{S}_\delta$ is a stationary point of r_ϵ then $\hat{v} \in \mathcal{F}$. Hence, as a result of (3.11), $\hat{v}$ is a stationary point also for problem (LR$_r$).

Sufficiency. Let $\hat{v}$ be a stationary point for problem (LR$_r$). Then $\hat{v} \in \mathcal{F}$ and by (3.11) it follows $\nabla r_\epsilon(\hat{v}) = 0$.

(*Correspondence of global minimizers*).

Necessity. By Proposition 3.3.1, the function r_ϵ admits a global minimizer $\hat{v}$, which is obviously a stationary point of r_ϵ. Hence we have $\hat{v} \in \mathcal{F}$ and $r_\epsilon(\hat{v}) = f_r(\hat{v})$. We proceed by contradiction. Assume that a global minimizer $\hat{v}$ of r_ϵ is not a global minimizer of problem (LR$_r$). Then there exists a point v^*, global minimizer of problem (LR$_r$), such that

$$r_\epsilon(\hat{v}) = f_r(\hat{v}) > f_r(v^*) = r_\epsilon(v^*),$$

but this contradicts the assumption that $\hat{v}$ is a global minimizer of r_ϵ.

Sufficiency. The claim is true by similar arguments.

(*Correspondence of local minimizers*).

Necessity. Since $\hat{v}$ is a local minimizer of r_ϵ, it is also a stationary point of r_ϵ, so that $\hat{v} \in \mathcal{F}$ and $r_\epsilon(\hat{v}) = f_r(\hat{v})$. Furthermore, there exists a $\rho > 0$ such that

$$f_r(\hat{v}) = r_\epsilon(\hat{v}) \leq r_\epsilon(v) \quad \forall\ v \in \mathcal{S}_\delta \cap \mathcal{B}_\rho(\hat{v}).$$

Therefore, by using (3.10), we have

$$f_r(\hat{v}) \leq r_\epsilon(v) = f_r(v) \quad \forall\ v \in \mathcal{F} \cap \mathcal{B}_\rho(\hat{v}),$$

namely $\hat{v}$ is a local minimizer for problem (LR$_r$).

Sufficiency. Since $\hat{v}$ is a local minimizer of (LR$_r$), there exists a $\rho > 0$ such that

$$f_r(\hat{v}) = r_\epsilon(\hat{v}) \leq f_r(v) = r_\epsilon(v) \quad \forall\ v \in \mathcal{F} \cap \mathcal{B}_\rho(\hat{v}).$$

We want to show that there exists $\gamma > 0$ such that

$$r_\epsilon(\hat{v}) \leq r_\epsilon(v) \quad \forall\ v \in S_\delta \cap B_\gamma(\hat{v}).$$

Nevertheless, it is sufficient to show that there is a $\gamma > 0$ such that for any $v \in \mathcal{S}_\delta \cap \mathcal{B}_\gamma(\hat{v})$ we have $p(v) \in \mathcal{F} \cap B_\rho(\hat{v})$, where $p(v)$ has components

$$p_i(v) = \frac{v_i}{\|v_i\|} \quad i = 1, \dots, n.$$

Indeed, in this case we would have

$$f_r(\hat{v}) = r_\epsilon(\hat{v}) \leq f_r(p(v)) = r_\epsilon(p(v)) \leq r_\epsilon(v).$$

Given any $v_i \neq 0 \in \mathbb{R}^r$, the projection over the unit norm set is simply $\frac{v_i}{\|v_i\|}$, that is

$$\|v_i - \bar{v}_i\| \geq \left\| v_i - \frac{v_i}{\|v_i\|} \right\| \quad \forall \bar{v}_i :\ \|\bar{v}_i\| = 1.$$

Hence, chosen $\gamma < \rho/2$, we can write

$$\begin{aligned} \|p(v) - \hat{v}\|^2 &= \sum_{i=1}^n \left\| \hat{v}_i - \frac{v_i}{\|v_i\|} \right\|^2 = \sum_{i=1}^n \left\| \hat{v}_i - \frac{v_i}{\|v_i\|} + v_i - v_i \right\|^2 \\ &\leq \sum_{i=1}^n \left(\|\hat{v}_i - v_i\|^2 + \left\| v_i - \frac{v_i}{\|v_i\|} \right\|^2 + 2\|\hat{v}_i - v_i\| \left\| v_i - \frac{v_i}{\|v_i\|} \right\| \right) \\ &\leq \sum_{i=1}^n 4\|\hat{v}_i - v_i\|^2 = 4\|\hat{v} - v\|^2 < 4\gamma^2 < \rho^2. \end{aligned}$$

Therefore, we have $r_\epsilon(\hat{v}) \leq r_\epsilon(v)$ for all $v \in S_\delta \cap B_\gamma(\hat{v})$, so that $\hat{v}$ is a local minimizer also for (QLR$_r$).

□

In summary, Theorem 3.3.2 states total equivalence between (LR$_r$) and (QLR$_r$), while Proposition 3.3.1, together with the differentiability of r_ϵ, makes any standard optimization algorithm globally convergent at least to a stationary point of problem (QLR$_r$) and hence of (LR$_r$).

Concerning the specific algorithm for finding a stationary point of problem (QLR$_r$), our choice falls in a gradient-type method. This is motivated by the fact that a gradient-type method keeps the generated sequence far from the boundary of $\mathcal{S}_\delta$, which might negatively affect the behavior of the algorithm.
To clarify this aspect, we consider a gradient-type method, defined by an iteration of the form

$$v_i^{k+1} = v_i^k - \alpha^k \nabla_{v_i} r_\epsilon(v^k) \quad i = 1, \ldots, n, \tag{3.12}$$

where $\alpha^k \in (0, \alpha_M]$ with $\alpha_M > 0$. The step-length α^k is obtained by a suitable line-search procedure satisfying

$$r_\epsilon(v^{k+1}) \leq r_\epsilon(v^0), \tag{3.13}$$

with $v^0 \in \mathcal{F}$. These choices guarantee for ϵ sufficiently large that the whole sequence stays in the set $\{v \in \mathbb{R}^{nr} : \|v_i\|^2 \geq 1,\, i = 1, \ldots, n\}$. Hence, for ϵ sufficiently large, the barrier term (3.5) reduces to a constant.

Proposition 3.3.3 *Given $\epsilon > 0$ and $v^0 \in \mathcal{F}$, let $\{v^k\}$ be the sequence generated with the iterative scheme* (3.12)*, where each α^k satisfies* (3.13) *and $\alpha^k \leq \alpha_M$. Then, there exists $\bar{\epsilon} > 0$ such that*

$$\|v_i^k\| \geq 1, \quad i = 1, \ldots, n,\; k = 0, 1, 2 \ldots$$

if $\epsilon \geq \bar{\epsilon}$.

Proof The sequence $\{v^k\}$ stays in the compact level set $\mathcal{L}_\epsilon(v^0)$ for a fixed value $\epsilon > 0$ as result of (3.13).
The proof is by induction. Assume that there exists $\bar{\epsilon} > 0$ such that for any $\epsilon \geq \bar{\epsilon}$ it is true that $\|v_i^k\|^2 \geq 1$. We show that the same is true also for k replaced by $k+1$. We can write

$$\begin{aligned} \|v_i^{k+1}\|^2 &= \|v_i^k\|^2 + (\alpha^k)^2 \|\nabla_{v_i} r_\epsilon(v^k)\|^2 - 2\alpha^k (v_i^k)^T \nabla_{v_i} r_\epsilon(v^k) \\ &\geq \|v_i^k\|^2 - \frac{8\alpha_M}{\epsilon\delta^2}(\|v_i^k\|^2 - 1)\|v_i^k\|^2, \end{aligned}$$

where we use (3.8). If $\|v_i^k\| = 1$ then $\|v_i^{k+1}\|^2 \geq 1$. Otherwise, we need to verify that a value of $\bar{\epsilon}$ exists such that for all $\epsilon \geq \bar{\epsilon}$ we have

$$(\|v_i^k\|^2 - 1) - \frac{8\alpha_M}{\epsilon\delta^2}(\|v_i^k\|^2 - 1)\|v_i^k\|^2 \geq 0,$$

namely

$$1 - \frac{8\alpha_M}{\epsilon\delta^2}\|v_i^k\|^2 \geq 0. \tag{3.14}$$

By Proposition 3.3.1 we have $\|v_i^k\|^2 \leq (2\overline{C}\epsilon\delta^2)^{\frac{1}{2}}+1$ for all k, which combined with (3.14), implies that ϵ has to satisfy

$$\epsilon - 8\frac{\alpha_M}{\delta^2}\left((2\overline{C}\delta^2\epsilon)^{\frac{1}{2}} + 1\right) \geq 0,$$

which is possible for a sufficiently large value of ϵ. □

In the following, we refer with `SpeeDP` the algorithm given by the combination of IRA scheme and the regularized Quotient function.

Finally, we want to stress the advantages of `SpeeDP` respect to the most efficient Low-Rank codes for problem (SDP), `SDPLR-MC` and `EXPA`. Respect to the former, as already explained before, `SpeeDP` covers all the theoretical and practical lacks. While, respect to the latter, we make the following considerations:

- The theoretical properties of the exact penalty function (3.3) of `EXPA` depend on the penalty parameter ϵ, that is required to be smaller than a certain threshold value. However, choosing a small value of ϵ may negatively affects both the efficiency and the accuracy of the algorithm `EXPA`. On the contrary, the equivalence properties of r_ϵ hold for any value of the parameter $\epsilon > 0$.

- Each computation of the penalty function (3.3) requires the evaluation the multiplier function $y(v)$ as in (3.2), which is not needed in the function r_ϵ. This implies that the computation of r_ϵ requires less matrix-vector products than the evaluation of the function (3.3), with a significant reduction of computational time.

All these advantages are supported by the numerical results reported in Section 3.5.

3.4 Heuristic for large-scale Max-Cut

Our heuristic is essentially the one due to Goemans and Williamson and described in Section 3.1, integrated with `SpeeDP` and a few simple additional details.

In Section 3.1 we underlined that the most expensive tasks in the randomized algorithm are the computation of the solution X^* of problem (SDP$_{MC}$) and the computation of the gramian vectors $v_1 \ldots, v_n$ of X^*. So far, Interior Point methods and a Cholesky factorization were used to accomplish these tasks. This fact has limited the use of GWA just for small and medium graphs.
On the contrary, SpeeDP makes possible to apply GWA to very large graphs since, on the one hand, it is able to solve problem (SDP$_{MC}$) in a reasonable amount of time also for very large graphs and, on the other hand, it outputs the vectors $v_1 \ldots, v_n$ without any additional cost.

In our procedure the cut provided by GWA is then improved by means of a 1-opt local search: for every single vertex, we modify the partition moving one node to the opposite set, checking if this leads to an improvement in terms of cut. If the cut does not improve, the node is moved back to the original subset. Then, we pass to the next node. The procedure is repeated several times respect to entire set of nodes, until no further improvement is possible.

In [23], where a similar heuristic is described but with problem (SDP$_{MC}$) solved by an Interior Point algorithm, a particularly successful step is proposed to further improve on the solution. The whole procedure is repeated a few times where the solution matrix X^* of problem (SDP$_{MC}$) is replaced by the convex combination

$$X' = \alpha X^* + (1 - \alpha)\hat{x}\hat{x}^T, \quad 0 < \alpha < 1,$$

where $\hat{x}$ is the representative vector of the current best cut. The idea behind this step is to bias the Goemans-Williams rounding with the current best cut and to force the rounding procedure to generate a cut in a neighborhood of the current best solution. This step does not require to solve problem (SDP$_{MC}$) again, but the Cholesky factorization of the matrix X' is needed.

We use a similar technique in our procedure. However, to avoid the Cholesky factorization, which is not suitable for very large instances, we solve a new problem (SDP$_{MC}$) after perturbing the objective function. Matrix L is replaced by the perturbed matrix L' given by

$$L' = L - \beta\hat{x}\hat{x}^T,$$

with $\beta > 0$. Such perturbation has again the effect of moving the solution of problem (SDP$_{MC}$). Actually, given $\hat{x}$ the current best integral solution, weights of the edges in $cut(\hat{x})$ are increased by β, whilst weights of those edges out of the cut are decreased by the same quantity. Therefore, we expect the solution of the perturbed (SDP$_{MC}$)

and hence the successive Goemans-Williamson rounding are somehow constrained in the neighborhood of $\hat{x}$. Based on new objective function $C = -L'/4$ we solve problem (SDP) with SpeeDP and perform the rounding, while we use the original L in the 1-opt phase. The whole procedure is repeated a few times with different values of β. It is important to give a clarification on the perturbed matrix L'. The perturbation $\beta\hat{x}\hat{x}^T$, as it is written, would imply as adding edges with no correspondence to the original graph. In truth, perturbation is meant just on the support of the graph, so that only real edges are modified. In this way we preserve the degree of sparsity of the graph.

Summarizing, the scheme of our heuristic algorithm is as follows.

```
SpeeDP-MC Algorithm

Data. A graph $G = (N, \mathcal{A})$, its Laplacian matrix $L$, $\alpha > 0$, an integer $k_{\max} > 0$.

Initialization. Set $\hat{x} = u$, and $\overline{\beta} = \alpha \sum_{i,j} |L_{ij}|/|E|$.

For $k = k_{max}, \dots, 0$
    1. Set $\beta = k\overline{\beta}$ and $L' = L - \beta(\hat{x}\hat{x}^T)$.
    2. Apply SpeeDP to problem (SDP) with $C = -L'/4$ and let $v_1, \dots v_n$ be
       the returned solution and $z_{LB}$ the corresponding computed lower bound.
    3. Apply the GWA hyperplane rounding phase to the vectors $v_1, \dots v_n$. This
       gives a bipartition representative vector $\bar{x}$.
    4. Apply the 1-opt improvement to $\bar{x}$. This gives a new bipartition repre-
       sentative vector $\tilde{x}$. If $\tilde{x}^T L \tilde{x} > \hat{x}^T L \hat{x}$, set $\hat{x} = \tilde{x}$.
End For

Return Best cut $\hat{x}$, lower bound $\frac{\hat{x}^T L \hat{x}}{4}$, upper bound $-z_{LB}$.
```

Note that the amount of perturbation decreases after each iteration until it gets to zero. We stress that repeating Step 1 several times is not as expensive at it may appear. The key aspect is to use warm start techniques: beginning from the second iteration, we start SpeeDP from the solution found at the previous step, so that each computation of the minimum is sensibly cheaper than the first one.

Besides the ability of treating graphs of very large sizes, another advantage of SpeeDP-MC is that it also provides a solution with a guaranteed optimality error bound, since it outputs an upper and lower bound on the value of the optimal cut.

3.5 Numerical results

In this section, we describe our computational experience both with algorithm SpeeDP for solving problem (SDP) and with the heuristic SpeeDP-MC for finding good Max-Cut solutions for large graphs.

SpeeDP is implemented in Fortran 90 and all the experiments have been run on a PC with 2 Gb of RAM.
The parameters δ and ϵ in (QLR$_r$) have been set to 0.25 and $10^3 \cdot \delta^{-1}$, respectively.
For the unconstrained optimization procedure we use a Fortran 90 implementation of the gradient-type method proposed in [25]. This algorithm is a non-monotone version of the Barzilai-Borwein method which satisfies (3.12) and (3.13).
As for the choice of the starting value r^1 of the rank, we use the same rule based on n as in [29], using values $8 \leq r^1 \leq 30$ for n from 100 up to more than 20 000. The updating rule for the rank r^j is simply $r^{j+1} = \min\left\{\lfloor r^j \cdot 1.5 \rfloor, \widehat{r}\right\}$.
Condition $Q - \mathrm{Diag}(y(\hat{v})) \succeq 0$ is checked by means of the subroutines dsaupd and dseupd of the ARPACK library, which compute smallest eigenvalues for sparse symmetric matrices.

As a first test, we consider SpeeDP for solving problem (SDP). We compare the performance of SpeeDP with the best codes in literature in the main classes of methods for solving this problem: Interior Point methods, Spectral Bundle method and Low-Rank methods.
As Interior Point method we select the dual-scaling algorithm defined in [7] and implemented in the software DSDP (version 5.8) downloaded from the web page[1]. The code DSDP is considered particularly efficient for solving problems with low-rank structure and sparsity in the data (as is the case for Max-Cut instances). In addition, DSDP has relatively low memory requirements for an Interior Point method and it is indeed able to solve instances up to around 10 000 nodes.
We also include the Spectral Bundle method SB that can be found in [32] and it is downloadable from the web page[2].
Among the NLP based methods, we test the code SDPLR-MC, proposed by Burer and

[1] http://www-unix.mcs.anl.gov/DSDP/

[2] http://www-user.tu-chemnitz.de/∼helmberg/

Monteiro in [16], downloadable from the web page[1], and the code EXPA proposed in [29]. Both EXPA and SDPLR-MC have a structure similar to SpeeDP. Indeed, the main scheme differs in the way of finding a stationary point for problem (LR$_r$).
In our comparison we do not include neither the code SDPLR defined in [14, 16, 17] nor the manifold optimization method GenRTR defined in [1], because the analysis of the computational results reported in [16] and [1] highlights that they are both outperformed by SDPLR-MC.

Our benchmark set is given by the SDP relaxation of standard Max-Cut instances, with number of nodes ranging from 100 to 20 000 and different degrees of sparsity. The first set of problems belongs to the SDPLIB collection of semidefinite programming test problems (hosted by B. Borchers) that can be downloaded from the web page[2]. The second set of problems belongs to the group Gset of randomly generated problems by means of the machine-independent graph generator *rudy* [54]. These problems can also be downloaded from Burer's web page[3].

SpeeDP, EXPA, SDPLR-MC, and SB solve all the test problems, whereas DSDP runs out of memory on the two largest problems (G77 and G81 of the Gset collection).

We compare the different codes on the basis of the level of accuracy and of the computational time. Besides reporting detailed results in Tables 3.1 and 3.2 (at the end of the chapter), we use a graphical description of the results using the *performance profile* proposed in [19]: given a set of solvers s and a set of problems p, we compare the performance of solver s on problem p against the best performance obtained by any solver on the same problem. To this end, it is defined the performance ratio

$$r_{p,s} = \frac{t_{p,s}}{\min\{t_{p,s'} : s' \in S\}},$$

where $t_{p,s}$ is the performance criterion used and afterwards it is derived a cumulative distribution function

$$\rho_s(\tau) = \frac{1}{n_p}\text{size}\{p \in P : r_{p,s} \leq \tau\}.$$

Then $\rho_s(\tau)$ is plot respect to the parameter τ in a logarithmic scale. The higher is the resulting curve, the better is the corresponding method respect to the chosen criterion; efficiency is measured by how fast the curve reaches the value of 1. If a method is able to solve all the test problems, then it will eventually reach the performance value 1 for sufficiently large τ.

[1] http://dollar.biz.uiowa.edu/∼sburer/software/SDPLR

[2] http://euler.nmt.edu/∼brian/sdplib

[3] http://dollar.biz.uiowa.edu/∼sburer/software/SDPLR

As for the accuracy, we report in Table 3.1 the primal and/or the dual objective function values obtained by the five methods: DSDP reports both primal and dual values as output; SpeeDP , EXPA and SDPLR-MC only report the primal objective value; SB produces a value for the dual objective function that is a bound on the optimal value of problem (SDP). Further, we plot in Figure 3.1 the performance profile setting the performance criterion to the relative duality gap, as suggested in [44]. In particular, we use the relative difference between the primal value $f^*_{p,s}$ of any solver s on problem p and dual value $f^*_{p,\texttt{DSDP}}$ given by DSDP, namely we set

$$t_{p,s} = \frac{f^*_{p,s} - f^*_{p,\texttt{DSDP}}}{1 + |f^*_{p,s}| + |f^*_{p,\texttt{DSDP}}|}.$$

It emerges that SpeeDP can be considered comparable with DSDP in terms of accuracy, whereas EXPA, SDPLR-MC, and SB are usually worse.

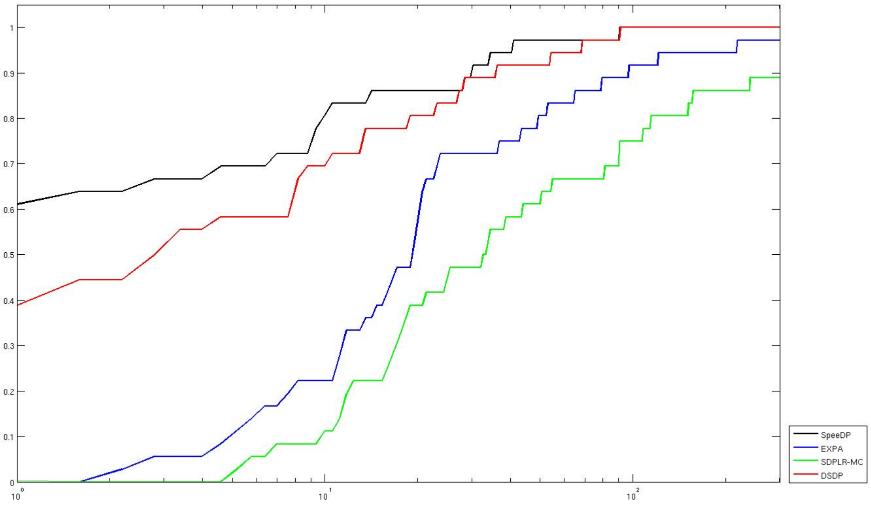

Figure 3.1: Accuracy - Relative duality gap performance profile

As regard the computational time, we report times in Table 3.2. Moreover, we also provide the performance profile for the time. We draw two profiles: one in Figure 3.2 with all the methods over test problems where DSDP succeeded. In the second profile, Figure 3.3, the three Low-Rank methods and SB time performances are reported over the entire testbed. These profiles show that SpeeDP outperforms all the other methods. More, it roughly always the fastest code (see the profile for $\tau = 0$) and with times significantly smaller than the ones of other codes.

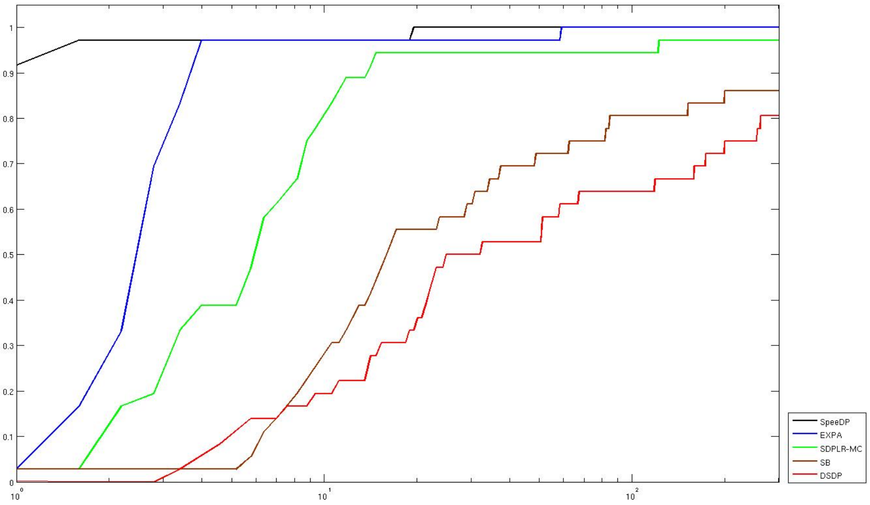

Figure 3.2: Time - All Methods CPU time performance profile

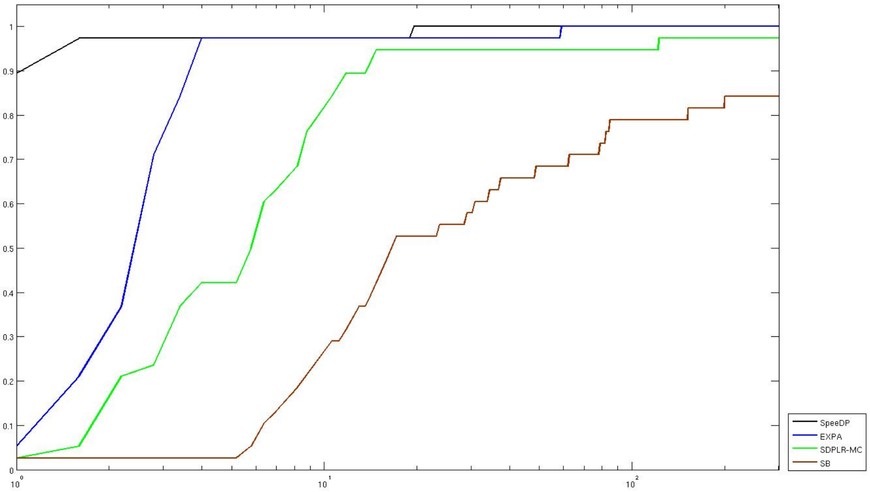

Figure 3.3: Time - NLP Methods CPU time performance profile

In conclusion for this first group of test, it emerges that `SpeeDP` is very reliable code for solving (SDP), being fast as well as accurate on the generated solution.

As second group of tests, we give an overview on the numerical results obtained by the heuristic algorithm `SpeeDP-MC` to find good solution for problem (MC). In particular, we consider large random graphs as instances of the Max-Cut.
`SpeeDP-MC` is implemented in C and uses the Fortran 90 version of `SpeeDP` as a routine. We used the graph generator *rudy* [54] to produce instances with different sizes, densities and weights.

First, we considered graphs with number of nodes n equals to $500+i\cdot 250$, for $i = 0,\ldots,8$ and with edge *density* equals to $10\%+i\cdot 10\%$, for $i = 0,\ldots,9$. For each pair (n, *density*) we generated three different graphs with positive weights ranging between 1 and 100. Details on the results can be found in [26]. We draw in Figure 3.4 the average CPU time and in Figure 3.5 the gap obtained, both as a function of the graph density.
As it emerges from the figures, the heuristic is able to produce a good cut in a small amount of time. As expected, the performance of the heuristic is better on sparse graphs in term of time, but the gap decreases when the graph density increases.

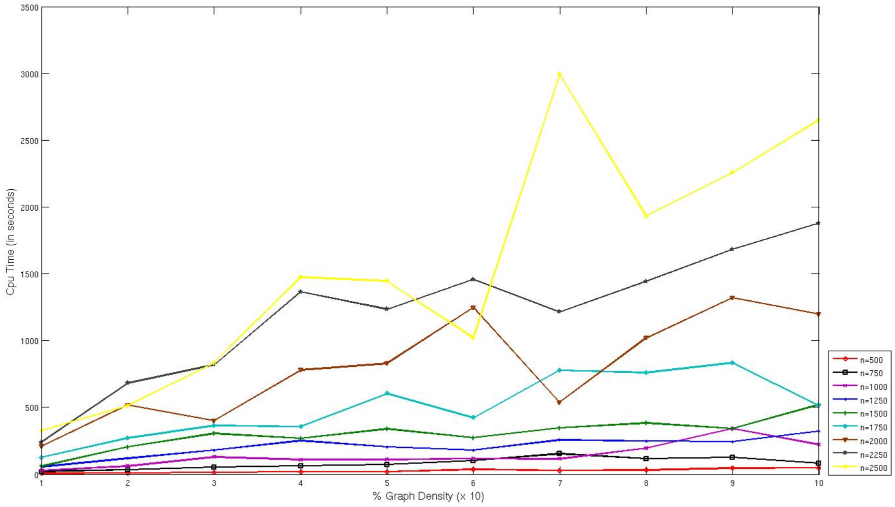

Figure 3.4: Time - Average `SpeeDP-MC` time for random graphs

In the second test, we consider huge graphs in order to verify how far we can go with the number of nodes. For this set of instances we run `SpeeDP-MC` on a machine with 6 Gb of RAM.

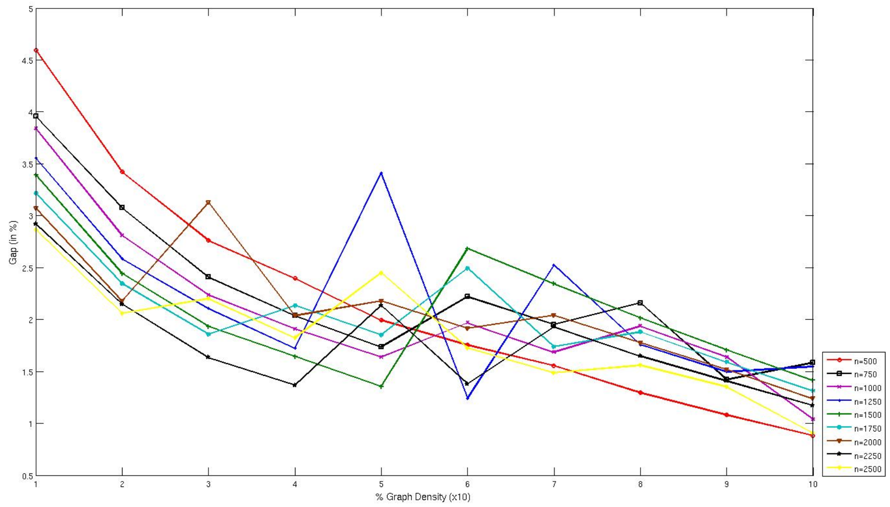

Figure 3.5: Gap - Average `SpeeDP-MC` gap for random graphs

We generate three random graphs with 100 001 nodes, 7 050 827 edges and different edge weights. The results are shown in Table 3.3, where we report the weight ranges, the total time, the bound value, the weight of the best cut obtained and the % gap.

We also generated some 6-regular graphs (3D toroidal grid graphs) with 1 030 301 nodes, 3 090 903 edges and different edge weights. The results are reported in Table 3.4.

To the best of our knowledge, no other methods can achieve this accuracy for graphs of comparable size. Actually, `SpeeDP-MC` allows to find very good cuts for graphs with million of edges.

	SpeeDP	Expa	SDPLR_MC	SB	DSDP	
Prob	primal	primal	primal	dual	primal	dual
mcp100	-226,15735	-226.15734	-226.15129	-226.15923	-226.15733	-226.15735
mcp124-1	-141.99048	-141.99041	-141.99003	-141.99370	-141.99044	-141.99048
mcp124-2	-269.88017	-269.88016	-269.88016	-269.88222	-269.88012	-269.88017
mcp124-3	-467.75011	-467.75010	-467.75009	-467.75370	-467.75004	-467.75012
mcp124-4	-864.41186	-864.41183	-864.41181	-864.41997	-864.41166	-864.41187
mcp250-1	-317.26434	-317.26421	-317.26431	-317.27079	-317.26429	-317.26435
mcp250-2	-531.93008	-531.93001	-531.92973	-531.93491	-531.92998	-531.93009
mcp250-3	-981.17257	-981.17248	-981.17239	-981.17796	-981.17239	-981.17257
mcp250-4	-1 681.9601	-1 681.9597	-1 681.9570	-1 681.9750	-1 681.9600	-1 681.9601
mcp500-1	-598.14849	-598.14798	-598.14800	-598.15877	-598.14840	-598.14852
mcp500-2	-1 070.0566	-1 070.0562	-1 045.0727	-1 070.0759	-1 070.0566	-1 070.0568
mcp500-3	-1 847.9700	-1 847.9694	-1 847.9666	-1 847.9836	-1 847.9695	-1 847.9700
mcp500-4	-3 566.7376	-3 566.7357	-3 566.7334	-3 566.7479	-3 566.7377	-3 566.7381
G01_mc	-12 083.198	-12 083.197	-12 083.042	-12 083.265	-12 083.196	-12 083.198
G11_mc	-629.16298	-629.14611	-629.15995	-629.17007	-629.16472	-629.16478
G14_mc	-3 191.5667	-3 191.5654	-3 191.5633	-3 191.5847	-3 191.5661	-3 191.5668
G22_mc	-14 135.946	-14 135.943	-14 135.867	-14 136.044	-14 135.945	-14 135.946
G32_mc	-1 567.6303	-1 567.5895	-1 567.6323	-15 67.6519	-1 567.6394	-1 567.6397
G35_mc	-8 014.7388	-8 014.7379	-8 014.7307	-8 014.8070	-8 014.7376	-8 014.7397
G36_mc	-8 005.9552	-8 005.9512	-8 005.9483	-8 006.0213	-8 005.9632	-8 005.9638
G43_mc	-7 032.2217	-7 032.2196	-7 032.2078	-7 032.2749	-7 032.2208	-7 032.2219
G48_mc	-5 999.9993	-5 999.9968	-5 999.9662	-6 000.0000	-5 999.9985	-6 000.0000
G51_mc	-4 006.2553	-4 006.2533	-4 006.2537	-4 006.2745	-4 006.2546	-4 006.2555
G52_mc	-4 009.6384	-4 009.6380	-4 009.6202	-4 009.6574	-4 009.6383	-4 009.6388
G55_mc	-11 039.460	-11 039.450	-11 039.341	-11 040.159	-11 039.449	-11 039.461
G57_mc	-3 885.4783	-3 885.3318	-3 885.4501	-3 885.5189	-3 885.4868	-3 885.4892
G58_mc	-20 135.875	-20 135.854	-20 136.032	-20 136.287	-20 136.181	-20 136.190
G60_mc	-15 222.239	-15 222.220	-15 222.138	-15 223.193	-15 222.257	-15 222.268
G62_mc	-5 430.8903	-5 430.6629	-5 430.8413	-5 430.9512	-5 430.9084	-5 430.9104
G63_mc	-28 243.308	-28 243.218	-28 243.876	-28 244.577	-28 244.406	-28 244.418
G64_mc	-10 465.836	-10 465.804	-10 465.868	-10 465.970	-10 465.898	-10 465.904
G65_mc	-6 205.4852	-6 205.2216	-6 205.4434	-6 205.5822	-6 205.5322	-6 205.5382
G66_mc	-7 077.1819	-7 077.0132	-7 077.1139	-7 077.2640	-7 077.2090	-7 077.2137
G67_mc	-7 744.3288	-7 744.2624	-7 744.3011	-7 744.4942	-7 744.4245	-7 744.4365
G70_mc	-9 861.5209	-9 861.4825	-9 861.3992	-9 861.7340	-9 861.5143	-9 861.5246
G72_mc	-7 808.3993	-7 808.1436	-7 808.4139	-7 808.5914	-7 808.5343	-7 808.5393
G77_mc	-11 045.624	-11 045.108	-11 045.448	-11 045.751	***	***
G81_mc	-15 656.082	-15 655.574	-15 655.791	-15 656.279	***	***

Table 3.1: Approximated optimal values on solving problem (SDP)

Prob	SpeeDP	Expa	SDPLR_MC	SB	DSDP
mcp100	0.016	0.032	0.050	0.240	0.065
mcp124-1	0.024	0.044	0.040	4.800	0.074
mcp124-2	0.016	0.028	0.130	0.230	0.086
mcp124-3	0.016	0.036	0.100	0.200	0.116
mcp124-4	0.020	0.052	0.120	0.160	0.096
mcp250-1	0.040	0.104	0.140	69.840	0.357
mcp250-2	0.028	0.072	0.380	0.430	0.388
mcp250-3	0.040	0.128	0.230	0.470	0.566
mcp250-4	0.064	0.152	0.240	0.420	0.710
mcp500-1	0.112	0.256	0.320	303.710	1.687
mcp500-2	0.104	0.208	51.110	15.820	2.040
mcp500-3	0.116	0.228	1.240	0.980	2.660
mcp500-4	0.164	0.456	1.480	1.000	4.003
G01_mc	0.576	1.448	4.930	3.410	18.690
G11_mc	1.060	1.380	1.010	12.180	4.166
G14 _mc	0.464	1.572	3.880	4.580	8.557
G22_mc	0.956	3.768	10.210	9.900	249.800
G32_mc	2.552	3.580	4.800	43.750	55.130
G35_mc	1.864	5.596	26.500	31.130	124.900
G36_mc	2.624	7.260	21.650	36.620	134.000
G43_mc	0.520	1.812	3.200	2.720	30.230
G48_mc	1.164	3.524	7.310	0.060	102.700
G51_mc	0.772	2.980	5.150	6.660	16.210
G52_mc	0.768	2.084	5.760	5.960	17.490
G55_mc	4.144	10.737	24.540	3370.770	1402.000
G57_mc	16.069	14.809	24.440	345.100	755.300
G58_mc	12.877	49.647	122.970	368.540	2053.000
G60_mc	7.932	18.597	25.260	6409.310	4552.000
G62_mc	17.641	21.565	36.240	535.630	2088.000
G63_mc	23.561	83.413	133.570	800.420	6019.000
G64_mc	17.905	56.884	178.380	665.280	6958.000
G65_mc	17.993	28.378	41.060	868.650	3123.000
G66_mc	22.733	33.526	46.580	1410.460	4548.000
G67_mc	16.349	38.094	54.440	1344.120	6191.000
G70_mc	12.545	35.474	71.640	15036.830	8939.000
G72_mc	17.405	34.814	55.180	1470.190	6040.000
G77_mc	57.816	54.227	71.670	4260.610	***
G81_mc	64.720	86.541	108.260	33538.190	***

Table 3.2: Time in seconds for solving problem (SDP)

Weights	Total CPU time	Upper Bound	Best Cut	gap%
1	15 043.98	4 113 227.8	3 959 852	3.87
$[1, 100]$	15 142.22	212 076 831.2	203 236 495	4.35
$[-1\,000, 1\,000]$	15 919.40	21 006 071 437.9	20 129 935 523	4.35

Table 3.3: Random sparse graphs with 100 001 nodes and 7 050 827 edges

Weights	Total CPU time	Upper Bound	Best Cut	gap%
1	4 723	3 090 133	3 060 300	0.97
$[1, 10]$	22 042	15 454 739	15 338 007	0.76
$[1, 1\,000]$	29 072	1 545 550 679	1 534 441 294	0.72
$[-100, 100]$	47 491	57 288 795	49 111 079	14.27

Table 3.4: 6-regular graphs with 1 030 301 nodes and 3 090 903 edges

4

Eigenvalue Optimization

Eigenvalues of symmetric matrices play an important role in many different areas of applied mathematics. Depending on the application, it can be required the computation of the eigenvalues of a constant matrix or of a matrix submitted to some optimization process.

In this chapter we consider both eigenvalues problems. In Section 4.1 we provide a new Low-Rank formulation for the problem of maximizing the smallest eigenvalue of symmetric matrix constrained to some subspace. As opposed, in Section 4.2 we deal with the classical problem of finding a bunch of smallest eigenvalues for a constant symmetric matrix. In particular, we provide a new nonlinear constrained formulation and for the special case of the smallest eigenvalues we give also a new unconstrained formulation, suitable for large-scale instances.

4.1 Extreme Eigenvalue Optimization

Optimization problems involving eigenvalues arise in many different application, with strong implications in Combinatorial Optimization and Structural Analysis. For a complete reference, dealing applications, theory and algorithms in Eigenvalue Optimization we advise [40].

Given $A_0, A_1, A_2, \ldots, A_m$ a set of $m+1$ symmetric matrices of order n, in eigenvalue optimization the problem is to find the linear combination of those matrices, constrained in some subspace, such that a certain convex combination of its eigenvalues is minimized. For simplicity and because its wide range of applications, we examine the problem relative to maximizing the smallest eigenvalue.

For any $x \in \mathbb{R}^m$, given the corresponding linear combination of the input matrices,

$$A_0 + \mathcal{A}^T(x) = A_0 + \sum_{i=1}^{m} x_i A_i,$$

are defined the eigenvalues (in not decreasing order)

$$\lambda_1(x) \leq \lambda_2(x) \leq \cdots \leq \lambda_n(x),$$

and the associated eigenvectors $y_1(x), y_2(x), \ldots, y_n(x)$. Without loss of generality, we consider the eigenvectors as an orthonormalized basis of $\mathbb{R}^n$. Moreover, x is constrained to the subspace defined by the intersection of t linear equalities with the nonnegative orthant, namely the set

$$\mathcal{X} = \{x \in \mathbb{R}^m : Bx = b,\, x \geq 0\},$$

where $B \in \mathbb{R}^{t \times m}$ and $b \in \mathbb{R}^t$. Overall, we get the following optimization problem

$$\begin{aligned} \max_{x} \quad & f(x) = \lambda_1(x) \\ & Bx = b \\ & x \geq 0. \end{aligned} \tag{P}$$

Without loss of generality, we assume $\mathcal{X}$ regular, namely with the columns of B linear independent. Moreover, we also suppose that there exists an optimal solution x^* for (P).

It is well known that the function $\lambda_1(x)$ defines a concave function, so that problem (P) is a convex problem. What makes this problem very difficult is that the objective function is not smooth and probably at the optimal solutions. Actually, it well known that functions depending on eigenvalue are not continuously differentiable at points where eigenvalues are not simple. What it is worse is that the optimization process leads generally to points where the eigenvalues coalescence, triggering the non-differentiability of the objective function. A theoretically explanation of this phenomenon can be found in [50], in truth for a more general eigenvalue function. More, as a function of m and n, are provided some lower and upper bounds on the multiplicity of the smallest eigenvalue at the optimal solutions.

In literature, nice theories and algorithms were developed for this non-smooth formulation [48, 49] as well as for the formulation of (P) as a convex feasibility problem [47]. Moreover, the most efficient solution approaches can be derived by means of Interior Point methods applied to the SDP formulations of (P).

As opposed, our approach tries to overcome non-smoothness by using an equivalent nonlinear continuously differentiable optimization problem. This formulation can be obtained by applying the Low-Rank approach to one SDP formulation of (P). Although this new formulation is non-convex, it is possible to show the standard necessary local optimality conditions are sufficient for global optimality. It follows that any second-order standard algorithm could be used to solve this new formulation in order to get a solution for (P).
Before to describe this new formulation and its properties, we recall the SDP formulations of (P), focusing on some characteristics which shed a light in the Low-Rank approach.

An obvious remark on notation $\mathcal{A}^T(x)$: it is used on purpose to recall the standard notation for SDP problems. Actually, it represents the adjoint operator of $\mathcal{A}(Z)$, operator for symmetric matrices, that is

$$\mathcal{A}(Z) = \begin{bmatrix} A_1 \bullet Z \\ \vdots \\ A_m \bullet Z \end{bmatrix}.$$

4.1.1 SDP formulation

First of all, we express the objective function $f(x)$ as a linear SDP problem.
Exploiting the well-known Fan Theorem [21], applied to smallest eigenvalues of $A_0 + \mathcal{A}^T(x)$, we express $f(x)$ as a nonlinear minimization problem, namely

$$f(x) = \lambda_1(x) = \min_{v \in \mathbb{R}^n} \left\{ v^T \left(A_0 + \mathcal{A}^T(x) \right) v : \|v\|^2 = 1 \right\}.$$

Thanks to the Fillmore-Williams result [22], that is

$$\text{conv} \left\{ vv^T : \ \|v\|^2 = 1 \right\} = \left\{ Z \in \mathbb{S}^n : \ I \bullet Z = 1, \ Z \succeq 0 \right\},$$

it follows the SDP formulation for $f(x)$

$$\begin{aligned} f(x) = \lambda_1(x) = \min_{Z} \quad & \left(A_0 + \mathcal{A}^T(x) \right) \bullet Z \\ & I \bullet Z = 1, \\ & Z \succeq 0. \end{aligned} \tag{4.1}$$

As proved in [49], in order to characterize the elements that gives the minimum in (4.1), we need some spectral information of $A_0 + \mathcal{A}^T(x)$, among which $r(x)$ the multiplicity of the eigenspace associated to $\lambda_1(x)$. Moreover, we need to consider the following set

$$\phi_d = \{u \in \mathbb{R}^d : \sum_i u_i = 1, \ u \geq 0\}$$

and the next lemma, whose proof is trivial and it is left to the reader.

Lemma 4.1.1 *Let $c \in \mathbb{R}^n$ and an integer $r \geq 1$ such that*

$$c_1 = \cdots = c_r < c_{r+1} \leq \cdots c_{r+1} \leq c_n,$$

then the optimal solution set of

$$\min_{\gamma \in \phi_n} c^T x$$

is given by

$$\Gamma_c^* = \{\gamma = (\gamma_a, \gamma_b) \in \mathbb{R}^n : \ \gamma_a \in \phi_r, \ \gamma_b = 0_{n-r}\}.$$

The next proposition shows that any feasible solution is optimal for (4.1) if and only its image is contained in the smallest eigenspace of $A_0 + \mathcal{A}^T(x)$. This result can be also derived from a more general one in [49].

Proposition 4.1.2 *Given $x \in \mathbb{R}^m$ and $r = r(x)$, then Z^* is an optimal solution of (4.1) if and only*

$$Z^* = \sum_{i=1}^{r} \gamma_i y_i(x) y_i(x)^T, \tag{4.2}$$

where $\gamma \in \phi_r$.

Proof Take Z^* optimal solution of (4.1). Because $\{y_1(x), \ldots, y_n(x)\}$ is an orthonormalized basis of $\mathbb{R}^n$ and by feasibility of Z^*, there exists $\gamma^* \in \phi_n$ such that

$$Z^* = \sum_{i=1}^{n} \gamma_i^* y_i(x) y_i(x)^T,$$

The objective value can be rewritten as

$$\left(A_0 + \mathcal{A}^T(x)\right) \bullet Z^* = \sum_{i=1}^{n} \gamma_i^* y_i(x)^T \left(A_0 + \mathcal{A}^T(x)\right) y_i(x) = \lambda(x)^T \gamma^*.$$

As regard of Lemma 4.1.1, $\gamma^* \in \Gamma^*_{\lambda(x)}$, so that Z^* satisfies (4.2).

On the other hand, assume Z^* satisfying (4.2) for some $\gamma \in \phi_r$. Then by construction $Z^* \succeq 0$ and

$$I \bullet Z^* = \sum_{i=1}^{r} \gamma_i I \bullet y_i(x) y_i(x)^T = \sum_{i=1}^{r} \gamma_i y_i(x)^T y_i(x) = \sum_{i=1}^{r} \gamma_i = 1,$$

so that Z^* is feasible for the SDP problem in (4.1). Moreover, it is easy to show

$$\big(A_0 + \mathcal{A}^T(x)\big) \bullet Z^* = \lambda_1(x),$$

hence that Z^* is optimal for (4.1). □

Proposition 4.1.2 ensures that any optimal solution Z^* can be rewritten as a convex combination of the rank-one matrices given by the smallest eigenvectors of $A_0+\mathcal{A}^T(x)$. It follows also that $1 \leq \operatorname{rank}(Z^*) \leq r(x)$.

In [48] this characterization of the optimal solutions in $f(x)$ was used to derive its subdifferential

$$\partial f(x) = \left\{ \mathcal{A}(Z) : \; Z = \sum_{i=1}^{r(x)} \gamma_i y_i(x) y_i(x)^T, \; \gamma \in \phi_{r(x)} \right\}.$$

From the practical point of view this form of the subdifferential is particularly useful because it does not involve any convex hull.

Exploiting the SDP dual of (4.1), we get another SDP formulation for $f(x)$

$$\begin{aligned} f(x) = \lambda_1(x) = \max_{s} \quad & s \\ & sI - A_0 - \mathcal{A}^T(x) \preceq 0, \\ & s \in \mathbb{R}. \end{aligned} \tag{4.3}$$

The equivalence derives from the fact the two SDP feasible sets in (4.1) and (4.3) are both strictly feasible, so that strong duality holds.

The two SDP formulations of the objective function allow to define equivalent SDP formulations for (P), respectively dual of each other.

The first SDP formulation comes out from the dual SDP formulation (4.3) of the objective function in (P), namely

$$\begin{aligned} \max_{s,x} \quad & s \\ & tI - \mathcal{A}^T(x) \preceq A_0, \\ & Bx = b, \\ & x \geq 0. \end{aligned} \tag{SDP}$$

Strict feasibility of the previous problem guarantees strong duality and hence equivalence of (P) also with the dual of (SDP), namely the problem

$$\begin{aligned}\min_{Z,\mu} \quad & b^T\mu + A_0 \bullet Z \\ & \mathcal{A}(Z) - B^T\mu \leq 0, \\ & I \bullet Z = 1, \\ & Z \succeq 0.\end{aligned} \tag{DSDP}$$

This SDP pair gives an efficient solution approach for (P). Actually, Interior Point methods, applied to (SDP) and (DSDP), compute an approximated optimal solution for (P) in polynomial time. Unfortunately, from the practical point of view, Interior Point methods slow down as the dimensions increase. For this reason we are interested in alternative solution approaches, less depending on dimensions n and m.

Consider an optimal solution x^* of (P), with the spectral information of $A_0 + \mathcal{A}^T(x^*)$ and in particular $r^* = r(A_0 + \mathcal{A}^T(x^*))$. In this situation we can not provide a complete characterization of the optimal solution set for (DSDP), but just a restriction.

Proposition 4.1.3 *Given x^* optimal solution of* (P) *and (Z^*, μ^*) an optimal solution of Problem* (DSDP)*, then Z^* can be rewritten as*

$$Z^* = \sum_{i=1}^{r^*} \gamma_i y_i(x^*) y_i(x^*)^T, \tag{4.4}$$

for some $\gamma \in \phi^*_r$.

Proof As in the proof of Proposition 4.1.2, there exists $\gamma^* \in \phi_n$ such that

$$Z^* = \sum_{i=1}^{n} \gamma^*_i y_i(x^*) y_i(x^*)^T.$$

By contradiction, assume that (4.4) is not satisfied, hence $\gamma^* \notin \Gamma^*_{\lambda(x^*)}$. Therefore, by Lemma 4.1.1 applied to $\lambda(x^*)$, we have that

$$A_0 \bullet Z^* + \mathcal{A}^T(x^*) \bullet Z^* = \lambda(x^*)^T \gamma^* > \lambda_1(x^*).$$

On the other hand, exploiting primal-dual feasibility, strong duality and the equivalence with (P), we have

$$\begin{aligned} A_0 \bullet Z^* + \mathcal{A}^T(x^*) \bullet Z^* &= A_0 \bullet Z^* + \mathcal{A}(Z^*)^T x^* \leq A_0 \bullet Z^* + (B\mu^*)^T x^* \\ &= A_0 \bullet Z^* + b^T\mu^* = \lambda_1(x^*), \end{aligned}$$

which is obviously a contradiction. Hence Z^* satisfies (4.4). □

Condition (4.4) is in this case only necessary for optimality. Actually, there exist convex combinations of the rank-one matrices given by the smallest eigenvectors of $A_0 + \mathcal{A}^T(x^*)$ that are not optimal for (DSDP), especially that one with small rank. Proposition 4.1.3 gives information just on the maximal rank, namely it ensures that any optimal solution (Z^*, μ^*) of (DSDP) satisfies $\mathrm{rank}(Z^*) \leq r^*$. Therefore, if we knew r^* of an optimal solution x^*, we could think to factorize $Z = VV^T$, with $V \in \mathbb{R}^{n \times r^*}$. This transformation is referable to Low-Rank approaches for standard SDP problems, as defined in Chapter 2, and its correctness is tied to a priori information about the minimal rank of an optimal solution.

Proposition 4.1.4 *Given x^* optimal solution of* (P)*, then the minimal rank satisfies*

$$r^* \leq \hat{r} = \min\left(\left\lfloor \frac{\sqrt{1 + 8(m + 1 - t)} - 1}{2} \right\rfloor, n\right). \tag{4.5}$$

Proof First of all, we rewrite (DSDP) in standard SDP notation, that is

$$\begin{aligned} \min_{Z, \mu^+, \mu^-, \theta} \quad & b^T\mu^+ - b^T\mu^- + A_0 \bullet Z \\ & \mathcal{A}(Z) - B^T\mu^+ + B^T\mu^- + \theta = 0, \\ & I \bullet Z = 1, \\ & \mu^+, \mu^-, \theta \geq 0 \\ & Z \succeq 0. \end{aligned}$$

Let $(Z^*, {\mu^+}^*, {\mu^-}^*, \theta^*)$ be the optimal solution of the above problem. In order to derive an upper bound on rank of Z^*, we apply results of Corollary 1.4.7, relative to SDP problems with blocks semidefinite variables. Actually, ${\mu^+}^*, {\mu^-}^*, \theta^*$ can be viewed as semidefinite diagonal matrices, with rank corresponding to the number of nonzero entries. Moreover, we assume that, in terms each component, μ^+ and μ^- can not be both different than zero. Overall, we have

$$\frac{r^*(r^* + 1)}{2} + t + \|\theta^*\|_0 \leq m + 1,$$

so that (4.5) follows. □

4.1.2 Low-Rank SDP formulation

In this part we apply the Low-Rank approach to (DSDP). However, instead of referring completely to the methodology described in Chapter 2, we define a stronger Low-Rank formulation. Main advantages are given from the possibility to define weaker global optimality conditions, than the ones for the general Low-Rank approach (see Propositions 2.2.2 and 2.2.3).

For given integer $r \geq 1$, we consider the transformation $Z = VV^T$ with $V \in \mathbb{R}^{n\times r}$. Beside the constraints derived from (DSDP), we we require also that V satisfies $U_r \bullet V^TV \geq 1$, in order to strength the formulation. Overall, we get the following problem

$$\begin{aligned} \min_{V,\mu} \quad & b^T\mu + A_0 \bullet VV^T \\ & \mathcal{A}(VV^T) - B^T\mu \leq 0, \\ & I_r \bullet V^TV = 1, \\ & U_r \bullet V^TV \geq 1, \\ & V \in \mathbb{R}^{n\times r}, \mu \in \mathbb{R}^t. \end{aligned} \tag{LR$_r$}$$

The first property for this formulation reminds a weak duality relation with (P) and this holds for any r.

Proposition 4.1.5 *Given $r \geq 1$, then for any $x \in \mathcal{X}$ and (V,μ) feasible for* (LR$_r$)*, the following inequality holds*

$$\lambda_1(x) \leq b^T\mu + A_0 \bullet VV^T.$$

Proof Given (V,μ) feasible for (LR$_r$), (VV^T,μ) is feasible for (DSDP). Hence, the optimal value of (LR$_r$) can not be smaller than the optimal value of (DSDP), which is equal to the optimal value of (P). Then the thesis follows for a generic feasible point in (P). □

One effect of the factorization is that solutions of (DSDP) with small rank are implicitly cut out from the feasible region of (LR$_r$). In order to preserve at least an optimal solution of (DSDP), r must be chosen not smaller than r^*.

Proposition 4.1.6 *Given $r \geq r^*$, problems* (DSDP) *and* (LR$_r$) *are equivalent in terms of global optimality.*

Proof First of all, (LR$_r$) admits an optimal solution because the objective function is linear and the feasible set is not empty and compact.
By weak duality defined in Proposition 4.1.5, the optimal value of (LR$_r$) can not be smaller than the optimal value of (P).
On the other side, consider the optimal solution x^* of (P) with associated dual optimal solution (Z^*,μ^*) of (DSDP). As a result of Proposition 4.1.3, we rewrite $Z^* = V^*V^{*T}$ with

$$V^* = \begin{bmatrix} \gamma_1 y_1(x^*) & \cdots & \gamma_{r^*} y_{r^*}(x^*) & 0_{n\times r-r^*} \end{bmatrix}$$

and $\gamma \in \phi_{r^*}$. It is not hard to see that (V^*, μ^*) is feasible for (LR_r). In particular, because $y_1(x^*), \ldots, y_{r^*}(x^*)$ are orthogonal between each other, we have

$$U_r \bullet {V^*}^T V^* = I_r \bullet {V^*}^T V^* = I \bullet V^* {V^*}^T = I \bullet Z^* = 1.$$

Then, it follows that the optimal value of (LR_r) can not be larger than the optimal value of (DSDP).

In conclusion, optimal values of (DSDP) and (LR_r) need to be equal. Moreover, a global solution of (LR_r) defines an optimal solution (DSDP) and viceversa. □

Although the equivalence expressed in the above proposition, (LR_r) can be used as alternative formulation for (DSDP) (hence for (P)) if one is able to recognize and compute global solutions. Actually, (LR_r) is a nonlinear constrained programming problem, non-convex and hence generally difficult to solve globally.

The loss of convexity could be implicitly retrieved as described in Chapter 2, where for the Low-Rank formulation of a standard primal SDP problem are provided necessary and sufficient global optimality conditions. For our specific problem, it can be shown something a bit stronger: first and second order KKT conditions are not only necessary but also sufficient for global optimality in (LR_r).

Using the Kronecker product and the vector representation for V we reformulate (LR_r) in this way

$$\begin{aligned} \min_{v,\mu} \quad & b^T\mu + v^T(I_r \otimes A_0)v \\ & v^T(I_r \otimes A_i)v - B_i^T\mu \le 0, \quad i = 1, \ldots, m \\ & 1 - v^T(U_r \otimes I_n)v \le 0, \\ & 1 - v^T(I_r \otimes I_n)v = 0, \\ & v \in \mathbb{R}^{nr}, \mu \in \mathbb{R}^t. \end{aligned} \tag{LR_r}$$

In the following, we refer with $g_1(v, \mu)$ for the first group of inequality constraints and with $g_2(v, \mu)$ for the last inequality constraint. On the other side we use $h_1(v, \mu)$ for the only equality constraint.

First of all, we observe that constraints g_2 is always active over the entire feasible set.

Proposition 4.1.7 *Given (v, μ) feasible for problem* (LR_r)*, then*

$$v^T(U_r \otimes I_n)v = 1. \tag{4.6}$$

Proof Assume by contradiction that $v^T(U_r \otimes I_n)v > 1$. As opposed, we have that

$$\begin{aligned} v^T(U_r \otimes I_n)v &= v^T(u_r \otimes I_n)(u_r \otimes I_n)^T v = \left\|(u_r \otimes I_n)^T v\right\|^2 = \left\|\sum_{i=1}^{r}(e_i \otimes I_n)^T v\right\|^2 \\ &\leq \sum_{i=1}^{r} \|(e_i \otimes I_n)^T v\|^2 = \sum_{i=1}^{r} v^T(E_{ii} \otimes I_n)v = v^T(I_r \otimes I_n)v = 1. \end{aligned}$$

□

The second property of (LR$_r$) is related to the regularity of the feasible set: actually it can be shown that a qualification condition for the constraints holds, namely a sort of *Mangasarian-Fromovitz* conditions.

Proposition 4.1.8 *Given* (v, μ) *feasible for* (LR$_r$), *then*

(i) $\nabla h_1(v, \mu)$ *and* $\nabla g_2(v, \mu)$ *are linear independent;*

(ii) *there exists* $d \in \mathbb{R}^{nr+t}$ *such that*

$$\nabla h_1(v, \mu)^T d = 0, \quad \nabla g_2(v, \mu)^T d = 0, \quad \nabla g_1(v, \mu)^T d < 0.$$

Proof By Proposition 4.1.7 constraint $g_2(v, \mu)$ is surely active. Assume that there exists a linear combination of $\nabla h_1(v, \mu)$ and $\nabla g_2(v, \mu)$, with coefficients α and γ, such that

$$\alpha \nabla h_1(v, \mu) + \gamma \nabla g_2(\mu, v) = 0.$$

In terms of variable v we have

$$2\alpha(I_r \otimes I_n)v + 2\gamma(U_r \otimes I_n)v = 0. \tag{4.7}$$

Pre-multiplying (4.7) with v^T gives

$$0 = 2\alpha v^T(I_r \otimes I_n)v + 2\gamma v^T(U_r \otimes I_n)v = 2\alpha + 2\gamma,$$

while with $v^T(U_r \otimes I_n)$ it gives

$$0 = 2\alpha v^T(U_r \otimes I_n)v + 2r\gamma v^T(U_r \otimes I_n)v = 2\alpha + 2r\gamma,$$

The last two equalities ensure (assumed $r > 1$) $\alpha, \gamma = 0$, hence that $\nabla h_1(v, \mu)$ and $\nabla g_2(v, \mu)$ are linear independent.

By assumption columns of B are linear independent, hence there not exists a not null vector x such that $Bx = 0$. In this situation, Gordan's Theorem guarantees that there exists a vector $s \in \mathbb{R}^m$ such that $B^T s > 0$. So, chosen $d = \begin{pmatrix} 0_{nr} & s \end{pmatrix}^T$, the second point in the proposition follows. □

Regularity of the feasible set makes KKT conditions being necessary to be satisfied for any local minimizer of (LR$_r$). In order to define these conditions, we introduce the Lagrangian function

$$\begin{aligned} L(\mu, v, x, \alpha, \gamma) &= b^T\mu + v^T\left(I_r \otimes A_0 + \mathcal{A}^T(x)\right)v \\ &+ \alpha\left(1 - v^T(I_r \otimes I_n)v\right) + \gamma\left(1 - v^T(U_r \otimes I_n)v\right). \end{aligned}$$

Under this notation, $x \in \mathbb{R}^m$ is the multiplier vector associated to the first block of inequality g_1, while $\alpha, \gamma \in \mathbb{R}$ are the multipliers associated respectively to h_1 and g_2.

We recall the first and second order KKT conditions for (LR$_r$): given $\hat{p} = (\hat{v}, \hat{\mu})$ a local minimum for (LR$_r$), then there exists $(\hat{x}, \hat{\alpha}, \hat{\gamma})$ such that

First-Order KKT conditions (FOC)

1. $\nabla_p L(\hat{p}, \hat{x}, \hat{\alpha}, \hat{\gamma}) = 0$;
2. $\hat{p}$ feasible;
3. $\hat{x}, \hat{\gamma} \geq 0$;
4. $\hat{x}^T g_1(\hat{p}) = 0$;
5. $\hat{\gamma} g_2(\hat{p}) = 0$;

Second-Order KKT conditions (SOC)

$$d^T\nabla^2_{pp} L(\hat{p}, \hat{x}, \hat{\alpha}, \hat{\gamma}) d \geq 0 \quad \forall d \in \mathcal{D},$$

where

$$\mathcal{D} = \left\{d \in \mathbb{R}^{nr+t} : \nabla h_1(\hat{p})^T d = 0,\ \nabla g_2(\hat{p})^T d = 0,\ \nabla g_1^{I_a}(\hat{p})^T d \leq 0\right\},$$

with $I_a = \left\{i \in \{1, \ldots, m\} : g_1^i(\hat{p}) = 0\right\}$.

Our goal is to show that conditions (FOC) and (SOC) characterize completely global solutions for (LR$_r$).

Proposition 4.1.9 *Given $(\hat{v}, \hat{\mu})$ a point satisfying (FOC) with multiplier $(\hat{x}, \hat{\alpha}, \hat{\gamma})$, then the following conditions are true:*

(i) *$\hat{x}$ is feasible for* (P)*;*

(ii) *$(\hat{\alpha}, \hat{\gamma})$ can be expressed in closed form expression as a function of $\hat{v}$ and $\hat{x}$, namely*

$$\hat{\alpha} + \hat{\gamma} = \hat{v}^T\left(I_r \otimes A_0 + \mathcal{A}^T(\hat{x})\right)\hat{v}, \tag{4.8}$$

$$\hat{\alpha} + r\hat{\gamma} = \hat{v}^T\left(U_r \otimes A_0 + \mathcal{A}^T(\hat{x})\right)\hat{v}; \tag{4.9}$$

(iii) *an upper bound of the objective value can be computed*

$$b^T\hat{\mu} + \hat{v}^T(I_n \otimes A_0)\hat{v} = \hat{\alpha} + \hat{\gamma} \leq \hat{\alpha} + r\hat{\gamma}. \tag{4.10}$$

(iv) $\hat{y}_1 = (u_r \otimes I_n)^T\hat{v}$ *is an eigenvector of* $A_0 + \mathcal{A}^T(\hat{x})$ *with eigenvalue* $\hat{\lambda}_1 = \hat{\alpha} + r\hat{\gamma}$.

Proof Condition 1 in (FOC) says that

$$0 = \nabla_\mu L = b - B\hat{x}, \tag{4.11}$$

$$0 = \nabla_v L = 2\left(I_r \otimes A_0 + \mathcal{A}^T(\hat{x})\right)\hat{v} - 2\hat{\alpha}(I_r \otimes I_n)\hat{v} - 2\hat{\gamma}(U_r \otimes I_n)\hat{v}. \tag{4.12}$$

Condition (4.11) together with non-negativity given by statement 3 in (FOC) proves that $\hat{x} \in \mathcal{X}$ and hence point (i).

As opposed, for the point (ii), we exploit (4.12) and the fact that $\hat{v}^T(U_r \otimes I_n)\hat{v} = 1$ (by Proposition 4.1.7). First, because

$$\begin{aligned} 0 = \hat{v}^T\nabla_v L &= 2\hat{v}^T\left(I_r \otimes A_0 + \mathcal{A}^T(\hat{x})\right)\hat{v} - 2\hat{\alpha}\hat{v}^T(I_r \otimes I_n)\hat{v} - 2\hat{\gamma}v^T(U_r \otimes I_n)\hat{v} \\ &= 2\hat{v}^T\left(I_r \otimes A_0 + \mathcal{A}^T(\hat{x})\right)\hat{v} - 2\hat{\alpha} - 2\hat{\gamma}, \end{aligned}$$

equality (4.8) follows. Second, equality (4.9) results from

$$\begin{aligned} 0 = \hat{v}^T(U_r \otimes I_n)\nabla_v L &= 2\hat{v}^T\left(U_r \otimes A_0 + \mathcal{A}^T(\hat{x})\right)\hat{v} - 2\hat{\alpha}\hat{v}^T(U_r \otimes I_n)\hat{v} - 2r\hat{\gamma}v^T(U_r \otimes I_n)\hat{v} \\ &= 2\hat{v}^T\left(U_r \otimes A_0 + \mathcal{A}^T(\hat{x})\right)\hat{v} - 2\hat{\alpha} - 2r\hat{\gamma}. \end{aligned}$$

Using the complementarity equation given by statement 4 in (FOC) and the closed-form expression for $\hat{\alpha}$ and $\hat{\gamma}$, we obtain

$$b^T\hat{\mu} + \hat{v}^T(I_r \otimes A_0)\hat{v} = \hat{v}^T\left(I_r \otimes A_0 + \mathcal{A}^T(\hat{x})\right)\hat{v} = \hat{\alpha} + \hat{\gamma} \leq \hat{\alpha} + r\hat{\gamma},$$

where the last inequality follows from the fact $\hat{\gamma} \geq 0$. This shows point (iii).

Consider $\hat{y}_1 = (u_r \otimes I_n)^T\hat{v}$ defined in point (iv). This vector has unit norm,

$$\|\hat{y}_1\|^2 = \hat{v}^T(U_r \otimes I_n)\hat{v} = 1.$$

Moreover, exploiting (4.12), we have that

$$\begin{aligned} 0 = (u_r^T \otimes I_n)\nabla_v L &= 2\left(u_r^T \otimes A_0 + \mathcal{A}^T(\hat{x})\right)\hat{v} - 2\hat{\alpha}(u_r^T \otimes I_n)\hat{v} - 2r\hat{\gamma}(u_r^T \otimes I_n)\hat{v} \\ &= 2\left(A_0 + \mathcal{A}^T(\hat{x})\right)\hat{y}_1 - 2(\hat{\alpha} + r\hat{\gamma})\hat{y}_1, \end{aligned}$$

which proves that $\hat{y}_1$ is an eigenvector of $A_0 + \mathcal{A}^T(\hat{x})$ with eigenvalue $\hat{\lambda}_1 = \hat{\alpha} + r\hat{\gamma}$. □

In order to guarantee that $(\hat{v}, \hat{\mu})$ is a global solution, $\hat{\lambda}_1$ must correspond to the smallest eigenvalue of $A_0 + \mathcal{A}^T(\hat{x})$. This is ensured by second-order KKT conditions.

Proposition 4.1.10 *Let $\hat{p} = (\hat{v}, \hat{\mu})$ a point satisfying (FOC) and (SOC) with multiplier $(\hat{x}, \hat{\alpha}, \hat{\gamma}.\hat{\eta})$. Then, $\hat{p}$ is a global solution for* (LR_r) *and $\hat{x}$ is a global solution for* (P). *Moreover, $\hat{\lambda}_1 = \hat{\alpha} + r\hat{\gamma}$ is the smallest eigenvalue of $A_0 + \mathcal{A}^T(\hat{x})$ with associated eigenvector $\hat{y}_1 = (u_r \otimes I_n)^T \hat{v}$.*

Proof As $\hat{p}$ satisfies (FOC), all the results in Proposition 4.1.9 hold. From now on, we use also the (SOC) conditions. The Hessian computed for $\hat{p}$ and the corresponding multipliers is given by

$$\nabla^2_{pp} L = \begin{pmatrix} 2I_r \otimes \left(A_0 + \mathcal{A}^T(\hat{x})\right) - 2\hat{\alpha} I_r \otimes I_n - 2\hat{\gamma} U_r \otimes I_n & 0_{nr\times t} \\ 0_{t\times nr} & 0_{t\times t} \end{pmatrix},$$

By definition of $\mathcal{D}$, a direction $d = (d_v, d_\mu) \in \mathcal{D}$ if and only if all those conditions are satisfied:

1. $d_v^T (I_r \otimes I_n)\hat{v} = 0$;
2. $d_v^T (U_r \otimes I_n)\hat{v} = 0$;
3. $2d_v^T (I_r \otimes A_i)\hat{v} \leq B_i^T d_\mu$, for any $i \in I_a(\hat{p})$.

Nevertheless, we can restrict to a subspace of $\mathcal{D}$, where the subvector d_v satisfies conditions $1, 2$ and $d_\mu = \sigma s$, where s such that $B^T s > 0$. Actually, in this case condition 3 is automatically satisfied with σ sufficiently large.

By Proposition 4.1.9, $\hat{y}_1$ is an eigenvector of $A_0 + \mathcal{A}^T(\hat{x})$. Assume by contradiction that $\hat{y}_1$ is not the smallest eigenvector of $\mathcal{A}^T(\hat{x})$, so that

$$\hat{\lambda}_1 > \lambda_1(\hat{x}), \tag{4.13}$$

and $y_1(\hat{x})^T \hat{y}_1 = 0$. Therefore, we can define $d = \begin{pmatrix} u_r \otimes y_1(\hat{x}) & \sigma s \end{pmatrix}$, for which conditions 1 and 2 are satisfied

$$\begin{aligned} d_v^T (I_r \otimes I_n)\hat{v} &= \left(u_r^T \otimes y_1(\hat{x})^T\right)(I_r \otimes I_n)\hat{v} = y_1(\hat{x})^T (u_r \otimes I_n)^T \hat{v} = y_1(\hat{x})^T \hat{y}_1 = 0, \\ d_v^T (U_r \otimes I_n)\hat{v} &= \left(u_r^T \otimes y_1(\hat{x})^T\right)(U_r \otimes I_n)\hat{v} = r y_1(\hat{x})^T (u_r \otimes I_n)^T \hat{v} = r y_1(\hat{x})^T \hat{y}_k = 0. \end{aligned}$$

It follows that $d \in \mathcal{D}$ and that second-order conditions can be used

$$\begin{aligned} 0 \leq d^T \nabla^2_{pp} L d &= 2\left(u_r^T \otimes y_1(\hat{x})^T\right)\left((I_r \otimes A_0 + \mathcal{A}^T(\hat{x})) - \hat{\alpha}(I_r \otimes I_n) - \hat{\gamma}(U_r \otimes I_n)\right)(u_r \otimes y_1(\hat{x})) \\ &= 2r y_1(\hat{x})^T \left(A_0 + \mathcal{A}^T(\hat{x})\right) y_1(\hat{x}) - 2r\hat{\alpha} - 2r^2\hat{\gamma}. \end{aligned}$$

Rearranging the above terms, we have

$$\lambda_1(\hat{x}) = y_1(\hat{x})^T \left(A_0 + \mathcal{A}^T(\hat{x})\right) y_1(\hat{x}) \geq \hat{\eta} + r\hat{\gamma} = \hat{\lambda}_1,$$

which is conflict with (4.13). It follows that $\hat{\lambda}_1 = \lambda_1(\hat{x})$ and $\hat{y}_1$ is the associated eigenvector.

By equation (4.10) and weak duality between (P) and (LR$_r$). we have that

$$b^T\hat{\mu} + \hat{v}^T(I_n \otimes A_0)\hat{v} \leq \hat{\alpha} + r\hat{\gamma} = \hat{\lambda}_1 = \lambda_1(\hat{x}) \leq b^T\hat{\mu} + \hat{v}^T(I_n \otimes A_0)\hat{v},$$

so that equality holds throughout. Strong duality proves optimality of $\hat{p}$ for (LR$_r$) and $\hat{x}$ for (P). □

In summary, we have defined a new Low-Rank formulation for (P). In particular, (LR$_r$) is a well-posed problem: all the functions are continuously differentiable and the feasible set is compact and non-empty. Moreover, for $r \geq r^*$, problems (LR$_r$) and the SDP formulation (DSDP) of (P) are equivalent in terms of global solutions. In addition, optimal solutions for (LR$_r$) are fully characterized by the first and second-order KKT conditions, with associated optimal multipliers defining the optimal solutions for (P).

Therefore, assumed $r \geq r^*$, a second-order method applied to (LR$_r$) can be viewed as a solution algorithm for (P). Generally speaking, a second-order algorithm, based in first and second-order derivatives, guarantees global convergence at least to second-order KKT point. An example of second-order algorithm is the one defined in [4].

The only difficulty that is left is the choice of a suitable r. On the one side, r must be chosen sufficiently large, in order to have $r \geq r^*$. On the other side, choosing r too large implies a too huge problem that needs to be solved.
In the spirit of Low-Rank approach, this uncertainty is solved sequentially: starting from r sufficiently small, a second-order stationary point for (LR$_r$) is computed. If the point is not optimal, namely strong duality does not hold, then r is increased. Because of Proposition 4.1.4, There is no sense to choose r larger than $\hat{r}$, the upper bound on r^*. Overall, we think at the following scheme.

Low-Rank algorithm (LoREig) for problem (P)

```
Parameter. $r^0 \in [2, \hat{r}]$, $\beta > 1$, $\rho > 0$

Initialization. $r = r^0$.

While $(r \leq \hat{r})$

    1. compute $\hat{p} = (\hat{\mu}, \hat{v})$, a second-order stationary point for $(\mathrm{LR}_r)$, with
       multiplier $(\hat{x}, \hat{\alpha}, \hat{\gamma})$.
    2. compute $\hat{f}_d = b^T \hat{\mu} + \hat{v}^T (I_n \otimes A_0) \hat{v}$.
    3. compute $\hat{f}_p = \lambda_1(\hat{x})$ smallest eigenvalue for $A_0 + \mathcal{A}^T(\hat{x})$.
    4. if $\left| \hat{f}_p - \hat{f}_d \right| \leq \rho$ then exit.
    5. otherwise set $r = \lceil \beta r \rceil$.

End While

Return $\hat{x}$, $\hat{f}_p$, $(\hat{v}, \hat{\mu})$, $\hat{f}_d$.
```

Algorithm LoREig is proved to be globally convergent to an optimal solution of (P) and to an optimal solution of (LR_r), within a fixed tolerance ρ. Actually, because of (4.5), in a finite number of increasing steps the condition $r \geq r^*$ will be satisfied. Therefore, eventually, the second-order point will results in an optimal solution of (LR_r), defining through the associated multiplier the optimal solution for (P). The tolerance ρ is needed because of the corresponding tolerances on computing the second-order point and the smallest eigenvalue.

In the end, we give some practical details in order to improve the performance of the algorithm. In the first step we use a cheaper first-order algorithm, that guarantees at least convergence to first-order KKT points. If r is getting too large, without getting strong duality, we switch to the second-order algorithm. This threshold value can be chosen as a fraction of the upper bound $\hat{r}$. Moreover, a smaller fraction of $\hat{r}$ can be also chosen as the initial r^0.

4.2 Smallest Eigenvalues of symmetric matrices

Computing few smallest eigenvalues, with the corresponding eigenvectors, of a matrix is a very important task because it arises in several applications. In particular, we think of the case relative to large and sparse symmetric matrices. The survey [34] gives a complete list of the eigensolver software available in literature, with also a brief description of the implemented methods.

In this section, we reformulate this problem as a nonlinear constrained optimization problem, a non-convex problem with first and second order conditions as necessary and sufficient for global optimality. For the special case of the extreme eigenvalues we give also a nonlinear unconstrained formulation.

Given $A \in \mathbb{S}^n$, chosen an integer $k \ll n$, the goal is to find the k smallest eigenvalues and the associated eigenvectors. In particular, we consider the eigenvalues of A labeled in not decreasing order,

$$\lambda_1 \leq \lambda_2 \leq \cdots \leq \lambda_n,$$

and the corresponding eigenvectors $y_1, y_2, \ldots, y_n$. Under this notation, our problem consists in finding the k eigenpairs $(\lambda_1, y_1), \ldots, (\lambda_k, y_k)$.

4.2.1 Nonlinear constrained formulation

In this subsection, we describe a new nonlinear constrained formulation for finding k smallest eigenvalues of A.
For this purpose, we define a vector $\beta \in \mathbb{R}^k$ such that

$$\beta_1 > \beta_2 > \cdots > \beta_k > 0, \tag{4.14}$$

and we recall a generalization of the Hoffman-Wielandt theorem (see [33]), which characterizes the solution for the non-convex quadratic problem over the set of orthogonal matrices.

Theorem 4.2.1 *Let $A \in \mathbb{S}^n$ and $B \in \mathbb{S}^k$, where $k \leq n$, with given spectral decompositions $A = Y\mathrm{Diag}(\lambda)Y^T$ and $B = Q\mathrm{Diag}(\beta)Q^T$. Then*

$$\min_{V \in \mathbb{R}^{n \times k}} \left\{ A \bullet VBV^T : V^TV = I_k \right\} = \min_{\phi:\{1,\ldots,k\} \to \{1\ldots,n\}} \left\{ \sum_i^k \beta_i \lambda_{\phi(i)} : \phi \textit{ injective} \right\}, \tag{4.15}$$

with the optimal solution given by

$$V^* = \begin{bmatrix} y_{\phi^*(1)} & \cdots & y_{\phi^*(k)} \end{bmatrix} \begin{bmatrix} q_1 & \cdots & q_k \end{bmatrix}^T.$$

The nonlinear formulation that we propose is

$$\begin{aligned} \min_{V} \quad & A \bullet VBV^T \\ & V^TV = I_k \\ & V \in \mathbb{R}^{n\times k}, \end{aligned} \tag{CP$_k$}$$

where $B = \mathrm{Diag}(\beta)$. The equivalence with the eigenvalue problem is proved by means of Theorem 4.2.1. First, because B is already in the diagonal form, the matrix Q in the Theorem is just the identity matrix. Then, because the eigenvalues of A are in not decreasing order and β satisfies (4.14), the equivalence in (4.15) implies that the optimal value of (CP$_k$) is given by

$$\sum_{i=1}^{k} \beta_i \lambda_i,$$

with optimal solution

$$V^* = [y_1, \dots, y_k],$$

namely with columns the k smallest eigenvectors of A. The problem of finding the k smallest eigenvalues of a matrix can be recast as nonlinear programming problem with orthogonality constraints.

To keep description more compact, we define the matrix operator $\bar{\mathcal{E}}$ as

$$\bar{\mathcal{E}}(M) = [\bar{E}_{ij} \bullet M]_{ij} \quad i = 1, \dots, k, \quad j = i, \dots, k,$$

where $\bar{E}_{ii} = e_i e_i^T$ and $\bar{E}_{ij} = e_i e_j^T + e_j e_i^T$ are matrices in $\mathbb{S}^k$. In addition, we take the vector $\delta \in \mathbb{R}^{\frac{(k+1)k}{2}}$ with each entry defined as

$$\delta_{ij} = \begin{cases} 1 & \text{if } i = j, \\ 0 & \text{otherwise} \end{cases} \qquad i = 1, \dots, k,\ j = i, \dots, k.$$

Under this notation

$$V^TV = I_k \quad \Leftrightarrow \quad \bar{\mathcal{E}}(V^TV) = \delta.$$

Using the Kronecker product and the vector representation, we reformulate (CP$_k$) in this way

$$\begin{aligned} \min_{v} \quad & v^T(B \otimes A)v \\ & v^T(\bar{E}_{ij} \otimes I_n)v = \delta_{ij} \quad i = 1, \dots, k, \quad j = i, \dots, k, \\ & v \in \mathbb{R}^{nk}. \end{aligned} \tag{CP$_k$}$$

In the following we refer the objective function with $f(v)$ and the vector of constraints with $h(v) \in \mathbb{R}^{\frac{(k+1)k}{2}}$, where each component is defined as

$$h(v)_{ij} = v^T(\bar{E}_{ij} \otimes I_n)v - \delta_{ij} \quad i = 1, \ldots, k, \quad j = i, \ldots, k,$$

For these function we have the following gradients

$$\begin{aligned}
\nabla f(v) &= 2(B \otimes A)v, \\
\nabla h_{ij} &= 2(\bar{E}^{ij} \otimes I_n)v \quad i = 1, \ldots, k, \quad j = i, \ldots, k,
\end{aligned}$$

Respect to the gradient of the constraints, for a fixed $v \in \mathbb{R}^{nk}$ and coefficient vector $\alpha \in \mathbb{R}^{\frac{(k+1)k}{2}}$, the following properties hold for any $i = 1, \ldots, k$ and $j = i+1, \ldots, k$

$$\nabla h_{ii}(v)^T \sum_{l=1}^{k} \sum_{r=l}^{k} \alpha_{lr} \nabla h_{lr}(v) = 4\alpha_{ii}\left(h_{ii}(v) + 1\right) + 2 \sum_{\substack{l=1 \\ l \neq i}}^{k} \alpha_{il} h_{il}(v), \tag{4.16}$$

$$\begin{aligned}
\nabla h_{ij}(v)^T \sum_{l=1}^{k} \sum_{r=l}^{k} \alpha_{lr} \nabla h_{lr}(v) &= 4\alpha_{ij}\left(h_{ii}(v) + h_{jj}(v) + 2\right) + 2\left(\alpha_{ii} + \alpha_{jj}\right) h_{ij}(v), \\
&+ 2 \sum_{\substack{l=1 \\ l \neq i \\ l \neq j}}^{k} \left(\alpha_{il} h_{il}(v) + \alpha_{jl} h_{jl}(v)\right). \qquad (4.17)
\end{aligned}$$

If v is feasible for (CP_k) then the above equations reduce simply to

$$\nabla h_{ii}(v)^T \sum_{l=1}^{k} \sum_{r=l}^{k} \alpha_{lr} \nabla h_{lr}(v) = 4\alpha_{ii}, \tag{4.18}$$

$$\nabla h_{ij}(v)^T \sum_{l=1}^{k} \sum_{r=l}^{k} \alpha_{lr} \nabla h_{lr}(v) = 8\alpha_{ij}. \tag{4.19}$$

The feasible set of (CP_k) is not empty and compact. Moreover, it is also regular.

Proposition 4.2.2 *Given v feasible for* (CP_k)*, then the columns of* $\nabla h(v)$ *are linear independent.*

Proof Assume that there exists a linear combination of the columns of $\nabla h(v)$, with coefficients α, such that

$$\sum_{r=1}^{k} \sum_{r=l}^{k} \alpha_{lr} \nabla h_{rl}(v) = 0.$$

Then, for any $i = 1, \dots, k$ and $j = i+1, \dots,$ as result of (4.18) and (4.19), we have

$$\begin{aligned} 0 =& \nabla h_{ii}(v)^T \sum_{l=1}^{k} \sum_{r=l}^{k} \alpha_{lr} \nabla h_{lr}(v) = 4\alpha_{ii}, \\ 0 =& \nabla h_{ij}(v)^T \sum_{l=1}^{k} \sum_{r=l}^{k} \alpha_{lr} \nabla h_{lr}(v) = 8\alpha_{ij}. \end{aligned}$$

Because all the coefficients must be zero, the columns of $\nabla h(v)$ are linear independent.
□

For the regularity first and second-order KKT conditions can be used as necessary optimality conditions. As we already said, for this particular problem these conditions are also sufficient for global optimality.
We start by introducing the Lagrangian function

$$\begin{aligned} L(v, \alpha) &= f(v) + \sum_{l=1}^{k} \sum_{r=l}^{k} \alpha_{lr} h_{lr}(v) \\ &= v^T(B \otimes A)v + v^T(\bar{\mathcal{E}}^T(\alpha) \otimes I_n)v - \alpha^T \delta. \end{aligned}$$

We recall the first and second order KKT conditions for (CP$_k$): if $\hat{v}$ is local minimum for (CP$_k$) then there exists $\hat{\alpha} \in \mathbb{R}^{\frac{(k+1)k}{2}}$ such that

First-Order KKT conditions (FOC)

1. $\nabla_v L(\hat{v}, \hat{\alpha}) = 0$;
2. $h(\hat{v}) = 0$;

Second-Order KKT conditions (SOC)

$$d^T \nabla^2_{vv} L(\hat{v}, \hat{\alpha}) d \geq 0 \quad \forall d \in \mathcal{D} = \left\{ d : \nabla h(\hat{v})^T d = 0 \right\}.$$

In the next two propositions we show that conditions (FOC) and (SOC) are necessary and sufficient for global optimality for (CP$_k$). To make it simple this is proven by blocks.

Proposition 4.2.3 *Given $\hat{v}$ a point satisfying (FOC) with multiplier $\hat{\alpha}$, then the following conditions are true:*

(i) *Multipliers $\hat{\alpha}$ can be expressed in closed form expression as a function of $\hat{v}$*

$$\hat{\alpha}_{ii} = -\hat{v}^T(\bar{E}_{ii} B \otimes A)\hat{v} \quad i = 1, \dots, k, \tag{4.20}$$

$$\hat{\alpha}_{ij} = -\frac{\hat{v}^T(\bar{E}_{ij} B \otimes A)\hat{v}}{2} = 0 \quad i = 1, \dots, k, j = i+1, \dots, k. \tag{4.21}$$

(ii) $\hat{y}_1 = (e_1 \otimes I_n)^T \hat{v}, \dots, \hat{y}_k = (e_k \otimes I_n)^T \hat{v}$ *are orthonormalized eigenvectors of A with eigenvalues*

$$\hat{\lambda}_1 = -\frac{\hat{\alpha}_{11}}{\beta_1}, \dots, \hat{\lambda}_k = -\frac{\hat{\alpha}_{kk}}{\beta_k}.$$

Proof Condition 1 in (FOC) says that

$$0 = \nabla_v L(v, \alpha) = \nabla f(v) + \sum_{l=1}^{k} \sum_{r=l}^{k} \alpha_{lr} \nabla h_{lr}(v).$$

Then, for any $i = 1, \dots, k$ and $j = i+1, \dots, k$, as result of (4.18) and (4.19), we have

$$\begin{aligned} 0 =& \nabla h_{ii}(v)^T \nabla_v L(\hat{v}, \hat{\alpha}) = 4\hat{v}^T (\bar{E}_{ii} B \otimes A)\hat{v} + 4\hat{\alpha}_{ii}, \\ 0 =& \nabla h_{ij}(v)^T \nabla_v L(\hat{v}, \hat{\alpha}) = 4\hat{v}^T (\bar{E}_{ij} B \otimes A)\hat{v} + 8\hat{\alpha}_{ij}, \end{aligned}$$

so that the closed-form expression for α are true. The fact that $\alpha_{ij} = 0$ follows from

$$\begin{aligned} 0 =& \hat{v}^T (E_{ij} \otimes I_n) \nabla_v L(\hat{v}, \hat{\alpha}) = 2\hat{v}^T (E_{ij} B \otimes A)\hat{v} + 2\hat{\alpha}_{ij} = 2\beta_j \hat{v}^T (E_{ij} \otimes A)\hat{v}v + 2\hat{\alpha}_{ij}, \\ 0 =& \hat{v}^T (E_{ji} \otimes I_n) \nabla_v L(\hat{v}, \hat{\alpha}) = 2\hat{v}^T (E_{ji} B \otimes A)\hat{v} + 2\hat{\alpha}_{ij} = 2\beta_i \hat{v}^T (E_{ji} \otimes A)\hat{v} + 2\hat{\alpha}_{ij}. \end{aligned}$$

Because $\hat{v}^T (E_{ji} \otimes A)\hat{v} = \hat{v}^T (E_{ij} \otimes A)\hat{v}$ and $\beta_i > \beta_j$, then $\hat{\alpha}_{ij} = 0$.

In point (ii) we need to show that $(\hat{\lambda}_1, \hat{y}_1), \dots, (\hat{\lambda}_k, \hat{y}_k)$ are k distinct eigenpairs of A. By definition of these vectors and by feasibility of $\hat{v}$, for any $i = 1, \dots, k$ and $j = i+1, \dots, k$, we have

$$\begin{aligned} \|\hat{y}_i\|^2 &= \hat{v}(E_{ii} \otimes I_n)\hat{v} = \hat{v}(\bar{E}_{ii} \otimes I_n)\hat{v} = 1, \\ \hat{y}_i^T \hat{y}_j &= \hat{v}^T (E_{ij} \otimes I_n)\hat{v} = \hat{v}^T (\bar{E}_{ij} \otimes I_n)\hat{v}/2 = 0, \end{aligned}$$

so that $\hat{y}_1, \dots, \hat{y}_k$ are orthonormalized vectors. Moreover, for any $i = 1, \dots, k$, we have

$$\begin{aligned} 0 = (e_i^T \otimes I_n) \nabla_v L(\hat{v}, \hat{\alpha}) &= 2(e_i^T B \otimes A)\hat{v} + 2\sum_{l=1}^{k} \sum_{r=l}^{k} \hat{\alpha}_{lr} (e_l^T \bar{E}_{lr} \otimes I_n)\hat{v} \\ &= 2\beta_i (e_i^T \otimes A)\hat{v} + 2\alpha_{ii} (e_i^T \otimes I_n)\hat{v} + 2\sum_{\substack{l=1 \\ l \neq i}}^{k} \alpha_{il} (e_i^T \otimes I_n)\hat{v} \\ &= 2\beta_i A\hat{y}_i + 2\hat{\alpha}_{ii}\hat{y}_i, \end{aligned}$$

which proves that each $\hat{y}_i$ is an eigenvector of A with eigenvalue $\hat{\lambda}_i = -\frac{\hat{\alpha}_{ii}}{\beta_i}$. □

Therefore, any stationary point of (CP$_k$) provides exactly k distinct eigenpairs of A. To ensure that such eigenpairs are the smallest ones we need additional second order assumptions on the stationary point.

Proposition 4.2.4 *Given $\hat{v}$ a point satisfying (FOC) and (SOC) with multiplier $\hat{\alpha}$, then*

(i) *$\hat{v}$ is a global solution for* (CP_k).

(ii) $(\hat{\lambda}_1, \hat{y}_1), \ldots, (\hat{\lambda}_k, \hat{y}_k)$, *as defined in Proposition* (4.2.3) *are the k smallest eigenpairs of A (in the right order).*

Proof As $\hat{v}$ satisfies (FOC), all the results in Proposition 4.2.3 hold. From now on, we will start using the second order information. The Hessian computed for $\hat{v}$ and the corresponding multiplier $\hat{\alpha}$ gives

$$\nabla^2_{vv} L = 2(B \otimes A) + 2(\bar{\mathcal{E}}^T(\hat{\alpha}) \otimes I_n).$$

By definition of $\mathcal{D}$, $d \in \mathcal{D}$ if and only if

$$d^T(\bar{E}_{lr} \otimes I_n)\hat{v} = 0, \quad l = 1, \ldots, k, \; r = l, \ldots, k.$$

Let us show that $\hat{\lambda}_1, \ldots, \hat{\lambda}_k$ are ordered such that

$$\hat{\lambda}_1 \leq \hat{\lambda}_2 \leq \cdots \leq \hat{\lambda}_k. \tag{4.22}$$

Given i and $j = i + 1$, consider the following direction

$$d_{ij} = (E_{ij} \otimes I_n)\hat{v} - (E_{ji} \otimes I_n)\hat{v},$$

which satisfies

$$\begin{aligned}
d_{ij}^T(\bar{E}_{ij} \otimes I_n)\hat{v} &= h_{jj}(v) - h_{ii}(v) = 0, \\
d_{ij}^T(\bar{E}_{ii} \otimes I_n)\hat{v} &= h_{ij}(v)/2 = 0, \\
d_{ij}^T(\bar{E}_{jj} \otimes I_n)\hat{v} &= -h_{ij}(v)/2 = 0, \\
d_{ij}^T(\bar{E}_{ll} \otimes I_n)\hat{v} &= h_{jl}(v)/2 - h_{il}(v)/2 = 0, \\
d_{ij}^T(\bar{E}_{lr} \otimes I_n)\hat{v} &= 0,
\end{aligned}$$

where r and l are indexes different than i and j. It follows that $d_{ij} \in \mathcal{D}$ and so second-order condition gives

$$\begin{aligned}
0 \leq d_{ij}^T \nabla^2_{vv} L d_{ij} &= \ldots \\
&= \beta_i \hat{y}_j^T A \hat{y}_j + \beta_j \hat{y}_i^T A \hat{y}_i + \hat{\alpha}_{ii} + \hat{\alpha}_{jj} \\
&= \beta_i \hat{\lambda}_j + \beta_j \hat{\lambda}_i - \beta_i \hat{\lambda}_i - \beta_j \hat{\lambda}_j \\
&= \beta_i (\hat{\lambda}_j - \hat{\lambda}_i) - \beta_j (\hat{\lambda}_j - \hat{\lambda}_i) \\
&= (\beta_i - \beta_j)(\hat{\lambda}_j - \hat{\lambda}_i),
\end{aligned}$$

which implies $\hat{\lambda}_i \leq \hat{\lambda}_j$ because $\beta_i > \beta_j$. Hence (4.22) holds.

It remains only to show that $\hat{\lambda}_1, \ldots, \hat{\lambda}_k$ are the k smallest eigenvalues of A, that is

$$\hat{\lambda}_1 = \lambda_1, \cdots \hat{\lambda}_k = \lambda_k.$$

Let $y \in \operatorname{span}\{y_1, \ldots, y_k\}^\perp$ another eigenvector of A with eigenvalue λ. Consider the following direction

$$d_k = (E_{kk} \otimes y),$$

which satisfies

$$\begin{aligned} d_k^T(\bar{E}_{kk} \otimes I_n)\hat{v} &= y^T \hat{y}_k = 0, \\ d_k^T(\bar{E}_{kl} \otimes I_n)\hat{v} &= y^T \hat{y}_l = 0, \\ d_k^T(\bar{E}_{lr} \otimes I_n)\hat{v} &= 0, \end{aligned}$$

with l and r are indexes different than k. It follows that $d_k \in \mathcal{D}$ and hence second-order condition gives

$$\begin{aligned} 0 \leq d_k^T \nabla_{vv}^2 L d_k &= \ldots \\ &= \beta_k \hat{y}^T A y + \hat{\alpha}_{kk} \\ &= \beta_k \lambda - \beta_k \hat{\lambda}_k \\ &= \beta_k(\lambda - \hat{\lambda}_k). \end{aligned}$$

Therefore, given a different eigenvector y with eigenvalue λ it follows

$$\lambda \geq \hat{\lambda}_k \geq \ldots \geq \hat{\lambda}_1,$$

which proves finally that $(\hat{\lambda}_1, \hat{y}_1), \ldots, (\hat{\lambda}_k, \hat{y}_k)$ are the k smallest eigenpairs of A. By definition of (CP$_k$), $\hat{v}$ is the optimal solution of (CP$_k$). □

Corollary 4.2.5 *First and second-order KKT conditions are necessary and sufficient conditions for global optimality for* (CP$_k$)*.*

Corollary 4.2.6 *For problem* (CP$_k$) *there are not exists local minimizers which are not global.*

In summary, we reformulate the problem of finding the k smallest eigenpair of a symmetric matrix as a nonlinear programming problem, with defined global optimality conditions.

4.2.2 Nonlinear unconstrained formulation for the Smallest Eigenvalue

Propositions 4.2.3 and 4.2.4, beyond the characterization of the global solutions of (CP_k) as stationary points, provide a closed-form expression for the multiplier α associated to KKT points. This information should help on defining an unconstrained formulation for (CP_k).

In this section we focus on (CP_1), namely the problem of the smallest eigenvalue of A. For this case we provide an equivalent unconstrained formulation with nice algorithmic properties. In particular, it can be shown that any gradient-type method converges directly to the global minimizer of (CP_1).
For the general problem (CP_k), with $k > 1$, the unconstrained formulation for (CP_1) could be also extended. Nevertheless, so far we were not able to prove the complete equivalence.

In the following we restrict our attention to (CP_1), so with $k = 1$ and $\beta_1 = 1$. It should be not hard to specialize all the results in Subsection 4.2.1 to this easier case. For this problem there is only one equality constraint. We denote the feasible set of (CP_1) with $\mathcal{F}$, that is

$$\mathcal{F} = \{v \in \mathbb{R}^n : \ \|v\|^2 = 1\}.$$

From Proposition 4.2.3, specialized to (CP_1), we can derive the following multiplier function

$$\alpha(v) = -v^T Av.$$

Up to the sign, for feasible points this function is equivalent to

$$\lambda(v) = \frac{v^T Av}{\|v\|^2}. \tag{4.23}$$

Function (4.23) is called Rayleigh Quotient and it gives the best estimate of an eigenvalue associated to the vector v. Actually, if v is an eigenvector of A, then $\lambda(v)$ gives exactly the associated eigenvalue, otherwise $\lambda(v)$ represents the scalar that minimizes the error

$$\|Av - \lambda v\|.$$

For given constant $\delta \in (0, 1)$ and parameter $\epsilon > 0$, over the open set

$$\mathcal{S}_\delta = \{v \in \mathbb{R}^n : \|v\|^2 > 1 - \delta\},$$

we define the following merit function

$$r_\epsilon(v) = \lambda(v) + \frac{1}{\epsilon}\frac{\left(\|v\|^2 - 1\right)^2}{d(v)}, \tag{4.24}$$

where the term

$$d(v) = \delta^2 - \left(1 - \|v\|^2\right)_+^2,$$

acts as a shifted barrier on the open set $\mathcal{S}_\delta$. Function (4.24) can be interpreted either as a regularization of the Rayleigh Quotient function or as an Exact Penalty function, derived from the corresponding Augmented Lagrangian for (CP$_1$), using $-\lambda(v)$ as a multiplier function. Regularization of the Rayleigh Quotient is similar to the one applied to Quotient function used for the relaxation of the Max-Cut (see Subsection 3.3.2).

The idea is to solve the following problem

$$\min_{v \in \mathcal{S}_\delta} \quad r_\epsilon(v). \tag{RQ$_1$}$$

The first important property is the compactness of the level sets of function $r_\epsilon(v)$, among which it guarantees the existence of a solution of problem (RQ$_1$).

Proposition 4.2.7 *For every given $\epsilon > 0$ and for every given $v^0 \in \mathcal{F}$, the level set $\mathcal{L}_\epsilon(v^0) = \{v \in \mathcal{S}_\delta : \; r_\epsilon(v) \le r_\epsilon(v^0)\}$ is compact and*

$$\mathcal{L}_\epsilon(v^0) \subseteq \left\{v \in \mathbb{R}^n : \|v\|^2 \le \left(2\epsilon\delta^2\|A\|\right)^{\frac{1}{2}} + 1\right\}. \tag{4.25}$$

Proof First, for every $v \in \mathcal{S}_\delta$, we have that

$$\lambda(v) = \frac{v^T A v}{\|v\|^2} \ge -\frac{\|A\|\|v\|^2}{\|v\|^2} = -\|A\|.$$

Hence we get

$$r_\epsilon(v) \ge -\|A\| + \frac{1}{\epsilon}\frac{\left(\|v\|^2 - 1\right)^2}{d(v)} \ge -\|A\| + \frac{1}{\epsilon}\frac{(\|v\|^2 - 1)^2}{\delta^2}. \tag{4.26}$$

For every $v \in \mathcal{L}_\epsilon(v^0)$, as $v^0 \in \mathcal{F}$, we can write

$$r_\epsilon(v) \le r_\epsilon(v^0) = f(v^0) \le C,$$

so that using (4.26) we get

$$\|v\|^2 \le \left(2\epsilon\delta^2\|A\|\right)^{\frac{1}{2}} + 1.$$

This implies that (4.25) holds and hence that $\mathcal{L}_\epsilon(v^0)$ is bounded.

On the other hand, any limit point $\hat{v}$ of a sequence of points $\{v^k\}$ in $\mathcal{L}_\epsilon(v^0)$ cannot belong to the boundary of $\mathcal{S}_\delta$. Indeed, if $\|\hat{v}\|^2 = 1 - \delta$ then $d(\hat{v}) = 0$ and hence $\lim_{k\to\infty} r_\epsilon(v^k) = \infty$. This fact contradicts $v^k \in \mathcal{L}_\epsilon(v^0)$ for k sufficiently large. Therefore, the level set $\mathcal{L}_\epsilon(v^0)$ is also closed and the claim follows. □

Function (4.24) is continuously differentiable over $\mathcal{S}_\delta$, with gradient defined as

$$\nabla r_\epsilon(v) = \nabla\lambda(v) + \frac{4}{\epsilon}\frac{(\|v\|^2 - 1)}{d(v)}\left(1 - \frac{(\|v\|^2-1)(1-\|v_i\|^2)_+}{d(v)}\right),$$

where

$$\nabla\lambda(v) = \frac{2}{\|v\|^2}\left(Av - \lambda(v)v\right).$$

Observe that, given any $v \in \mathcal{S}_\delta$, the following ortogonality condition holds

$$v^T\nabla\lambda(v) = \frac{2}{\|v\|^2}\left(v^TAv - \lambda(v)\|v\|^2\right) = 0. \tag{4.27}$$

Now, we are ready to show the most significant property for problem (RQ$_1$), the exactness respect to (CP$_1$).

Theorem 4.2.8 *For any $\epsilon > 0$ the following correspondences hold:*

(i) *a point $\hat{v}$ is a stationary point of Problem* (RQ$_1$) *if and only if it is a stationary point of problem* (CP$_1$).

(ii) *a point $\hat{v}$ is a global minimizer of problem* (RQ$_1$) *if and only if it is a global minimizer of problem* (CP$_1$).

(iii) *a point $\hat{v}$ is a local minimizer of problem* (RQ$_1$) *if and only if it is a local minimizer of problem* (CP$_1$).

Proof Exploiting property (4.27), for any $v \in \mathcal{S}_\delta$, we have

$$v^T\nabla r_\epsilon(v) = \frac{4}{\epsilon}\frac{(\|v\|^2-1)\|v\|^2}{d(v)}\left(1 - \frac{(\|v\|^2-1)(1-\|v_i\|^2)_+}{d(v)}\right),$$

so that if $\|v\|^2 \geq 1$ we get

$$v^T\nabla r_\epsilon(v) = \frac{4}{\epsilon}\frac{(\|v\|^2-1)\|v\|^2}{\delta^2}, \tag{4.28}$$

otherwise

$$v^T\nabla r_\epsilon(v) = \frac{4}{\epsilon}\frac{(\|v\|^2-1)\|v\|^2}{d(v)}\left(1 + \frac{(\|v\|^2-1)^2}{d(v)}\right). \tag{4.29}$$

For any point $v \in \mathcal{F}$

$$\begin{aligned} r_\epsilon(v) &= f(v), \qquad (4.30) \\ \nabla r_\epsilon(v) &= \nabla\lambda(v) = \nabla_v L(v, -\lambda(v)). \qquad (4.31) \end{aligned}$$

Now we prove the correspondences stated in our claims.

(*Correspondence of stationary points*).

Necessity. if $\hat{v} \in \mathcal{S}_\delta$ is a stationary point of r_ϵ, namely with $\nabla r_\epsilon(\hat{v}) = 0$, then, by equation (4.28) or (4.29), we have $\hat{v} \in \mathcal{F}$. Therefore (4.31) proves that $\hat{v}$ is a stationary point also for problem (CP$_1$).

Sufficiency. Let $\hat{v}$ be a stationary point for problem (CP$_1$). Then $\hat{v} \in \mathcal{F}$ is stationary point of (CP$_1$) by (4.31).

(*Correspondence of global minimizers*).

Necessity. By Proposition 4.2.7, the function r_ϵ admits a global minimizer $\hat{v}$, so that it is a stationary point of r_ϵ. It follows $\hat{v} \in \mathcal{F}$, and hence $r_\epsilon(\hat{v}) = f(\hat{v})$. We proceed by contradiction. Assume that a global minimizer $\hat{v}$ of r_ϵ is not a global minimizer of problem (CP$_1$). Then there exists a point v^* global minimizer of problem (CP$_1$) such that

$$r_\epsilon(\hat{v}) = f(\hat{v}) > f(v^*) = r_\epsilon(v^*),$$

but this contradicts the assumption that $\hat{v}$ is a global minimizer of r_ϵ.

Sufficiency. The claim is true by similar arguments.

(*Correspondence of local minimizers*).

Necessity. Since $\hat{v}$ is a local minimizer of r_ϵ over $\mathcal{S}_\delta$, it is also stationary point, so that $\hat{v} \in \mathcal{F}$ and $r_\epsilon(\hat{v}) = f(\hat{v})$. Furthermore, there exists a $\rho > 0$ such that

$$f(\hat{v}) = r_\epsilon(\hat{v}) \le r_\epsilon(v) \quad \forall\, v \in \mathcal{S}_\delta \cap \mathcal{B}_\rho(\hat{v}).$$

Therefore by using (4.30)

$$f(\hat{v}) \le r_\epsilon(v) = f(v) \quad \forall\, v \in \mathcal{F} \cap \mathcal{B}_\rho(\hat{v}),$$

hence $\hat{v}$ is a local minimizer for problem (CP$_1$).

Sufficiency. Since $\hat{v}$ is a local minimizer of (CP$_1$), then it is proved to be global by Corollary 4.2.6. Therefore $\hat{v}$ is global and hence also local minimizer for (RQ$_1$).

□

Theorem (4.2.8) states an exact relation between Problem (CP$_1$) and (RQ$_1$). Also, the equivalence between local minimizers ensures that also for problem (RQ$_1$) there not exists local minimizer not global.

Moreover, Proposition 4.2.7, together with the differentiability of r_ϵ, makes any standard optimization algorithm globally convergent at least to a stationary point of problem (RQ$_1$), hence of (CP$_1$). In particular, standard algorithms applied to function r_ϵ produce a sequence $\{v^k\}$ such that

- the sequence $\{v^k\}$ is contained in the compact $\mathcal{L}_\epsilon(v^0)$;
- the sequence $\{v^k\}$ admits accumulation points;
- every accumulation point $\hat{v}$ of the sequence $\{v^k\}$ is a stationary point for (RQ$_1$), namely
$$\nabla r_\epsilon(\hat{v}) = \nabla\lambda(\hat{v}) = 2(A\hat{v} - \lambda(\hat{v})\hat{v}) = 0.$$

Therefore, any standard algorithm guarantees at least convergence to an eigenvector $\hat{v}$ of A with eigenvalue $\lambda(\hat{v})$. The unconstrained algorithm should fall in the gradient-type class. This choice is motivated by two reasons.

The barrier term plays a key role to make standard optimization methods be globally convergent for problem (RQ$_1$). Nevertheless, generally speaking, a barrier term affects negatively the performance behavior of any optimization method, especially when the produced sequence gets closer to the boundary of $\mathcal{S}_\delta$. Favorably, the use of a gradient-like method makes the barrier-penalty term behaves just as a penalty term on the feasibility of problem (CP$_1$).

First of all, we give a description for a general gradient method: starting with $v^0 \in \mathcal{F}$, we define an iteration of the form

$$v^{k+1} = v^k - \alpha^k \nabla r_\epsilon(v^k), \tag{4.32}$$

where $\alpha^k > 0$ is obtained by a suitable linesearch procedure satisfying at least

$$r_\epsilon(v^{k+1}) \leq r_\epsilon(x^0). \tag{4.33}$$

We prove that for ϵ sufficiently large the produced sequence stays in the set

$$\{v \in \mathbb{R}^n \, : \, \|v\|^2 \geq 1\},$$

hence the penalty-barrier term reduces simply to a penalty term. In particular, the following proposition holds.

Proposition 4.2.9 *Let $v^0 \in \mathcal{F}$ and let $\{v^k\}$ be the sequence generated with the iterative scheme* (4.32), *where each α^k satisfies* (4.33) *and $\alpha^k \leq \alpha_M$. There exists $\bar{\epsilon} > 0$ such that if $\epsilon \geq \bar{\epsilon}$ then*

$$\|v^k\| \geq 1 \quad k = 1, 2, \ldots.$$

Proof By (4.33), for a fixed value $\epsilon > 0$, the sequence $\{v^k\}$ stays in the compact level set $\mathcal{L}_\epsilon(v^0)$. The proof is by induction. Assume that there exists $\bar{\epsilon} > 0$ such that, for any $\epsilon \geq \bar{\epsilon}$, it is true that $\|v^k\|^2 \geq 1$. We show that is true also for $k+1$. By property (4.27) we get

$$\begin{aligned} \|v^{k+1}\|^2 &= \|v^k\|^2 + (\alpha^k)^2 \|\nabla_v r_\epsilon(v^k)\|^2 - 2\alpha^k (v^k)^T \nabla r_\epsilon(v^k) \\ &= \|v^k\|^2 + (\alpha^k)^2 \|\nabla r_\epsilon(v^k)\|^2 - \frac{8\alpha^k}{\epsilon} \frac{(\|v^k\|^2 - 1)\|v^k\|^2}{\delta^2} \\ &\geq \|v^k\|^2 - \frac{8\alpha_M}{\epsilon\delta^2} (\|v^k\|^2 - 1)\|v^k\|^2, \end{aligned}$$

Assume that $\|v^k\| \geq 1$. If $\|v^k\| = 1$ then $\|v^{k+1}\|^2 \geq 1$. Otherwise we need to verify that a value of $\bar{\epsilon}$ exists such that for all $\epsilon \geq \bar{\epsilon}$

$$(\|v^k\|^2 - 1) - \frac{8\alpha_M}{\epsilon\delta^2} (\|v^k\|^2 - 1)\|v^k\|^2 \geq 0,$$

namely

$$1 - \frac{8\alpha_M}{\epsilon\delta^2} \|v^k\|^2 \geq 0. \tag{4.34}$$

By Proposition 4.25 we have that for all iterations k

$$\|v^k\|^2 \leq (\epsilon\delta^2 \|A\|)^{\frac{1}{2}} + 1. \tag{4.35}$$

Therefore (4.35) combined with (4.34) implies

$$\epsilon - 8\frac{\alpha_M}{\delta^2} \left((2\delta\epsilon\|A\|)^{\frac{1}{2}} + 1 \right) \geq 0$$

which is satisfied for some $\epsilon \geq \bar{\epsilon}$. □

Anyway, the most important advantage in using a gradient-based method, instead of other unconstrained minimization methods, is that it is guaranteed convergence to an eigenvector associated to smallest eigenvalue, global solution of (CP_1) and (RQ_1).

The eigenvectors $y_1, \ldots, y_n$ of A represent an orthonormalized basis of $\mathbb{R}^n$, so that any starting point v^0 can be expressed through this basis, namely

$$v^0 = \sum_{i=1}^{n} c_i^0 y_i$$

where $c_i^0 = y_i^T v^0$ with $i = 1, \ldots, n$. It follows that the sum can be restricted to the subset of indexes defined by

$$I(v^0) = \{i \in \{1, \ldots, n\} : \; y_i^T v^0 \neq 0\}.$$

Likewise, we can express every point of the sequence $\{v^k\}$ generated by iteration like (4.32): for example v^{k+1} can be expressed by

$$v^{k+1} = \sum_{i=1}^{n} c_i^{k+1} y_i,$$

where each c_i^{k+1} can be evaluated through the previous c_i^k, namely by

$$\begin{aligned} c_i^{k+1} &= y_i^T v^{k+1} \\ &= y_i^T (v^k - \alpha^k \nabla r_\epsilon(v^k)) \\ &= \left(1 + \frac{2\alpha^k}{\|v^k\|^2}(\lambda(v^k) - \lambda_i) - \frac{4\alpha^k}{\epsilon}\frac{(\|v^k\|^2 - 1)}{b(v^k)}\right) c_i^k. \end{aligned}$$

Therefore, we have that $I(v^{k+1}) \subseteq I(v^k) \subseteq \ldots \subseteq I(v^0)$.

In summary, in order to have convergence to a smallest eigenvalue, the minimal condition is that v^0 is not orthogonal to the eigenspace associated to λ_1, which we denote with $\mathcal{Y}_1$. Luckily this is the only condition needed.

Proposition 4.2.10 *Given $v^0 \in \mathcal{F}$ such that $v^0 \notin \mathcal{Y}_1^\perp$, let $\{v^k\}$ be the sequence generated by a globally convergent gradient-type method. There exists $\bar{\epsilon} > 0$ such that if $\epsilon \geq \bar{\epsilon}$ then every accumulation point of $\{v^k\}$ belongs to $\mathcal{Y}_1$, namely a global minimizer of* (CP_1).

Proof Because of the assumption on standard gradient algorithms, we have that $\{v^k\}$ admits accumulation points and that every accumulation point is an eigenvector of A.

Reasoning by contradiction, let us assume that there exists a subsequence K, redefined as the entire sequence, such that

$$\lim_{k\to\infty} v^k = \hat{v} \text{ with } \|\hat{v}\| = 1, \tag{4.36}$$

where $\hat{v}$ is an eigenvector of A with eigenvalue $\lambda(\hat{v}) > \lambda_1$.
Using the basis $y_1, \dots, y_n$ the eigenvector $\hat{v}$ can be expressed as

$$\hat{v} = \sum_{i=1}^{n} c_i y_i,$$

where each $c_i = y_i^T \hat{v}$. Because $\hat{v}$ is an eigenvector, but not the smallest, there exists an index $s > 1$ such that

$$c_s = y_s^T \hat{v} \neq 0, \quad \lambda(\hat{v}) = \lambda_s.$$

Without loss of generality we can assume $y_s^T \hat{v} > 0$.
As opposed, by hypothesis on v^0, there exists $y_m \in \mathcal{Y}_1$ such that $c_m^0 = y_m^T v^0 \neq 0$. Without loss of generality, we assume $y_m = y_1$ and $c_1^0 = y_1^T v^0 > 0$. It is possible to prove by induction that $\{c_1^k\} > 0$. So let assume that $c_1^k > 0$ and show that also $c_1^{k+1} > 0$. As before we express

$$c_1^{k+1} = \left[\left(\frac{2\alpha^k}{\|v^k\|^2}(\lambda(v^k) - \lambda_1)\right) + \left(1 - \frac{4\alpha^k}{\epsilon\delta^2}(\|v^k\|^2 - 1)\right)\right] c_1^k.$$

The first term is surely nonnegative, cause $\lambda(v^k) > \lambda_1$, while the second term can be made positive choosing $\epsilon \geq \bar{\epsilon}$ as it is done in the last part of proof of Proposition 4.2.9. Hence we have $c_1^{k+1} > 0$.
Moreover, because of (4.36), we have that

$$\lim_{k\to\infty} c_s^k = c_s > 0, \tag{4.37}$$

so that, as results of $I(v^{k+1}) \subseteq I(v^k)$, we have $\{c_s^k\} > 0$.
As opposed, because $\hat{v}$ and y_1 are eigenvectors associated to different eigenvalues, they are orthogonal between each other, so that $c_1 = y_1^T \hat{v} = 0$. Again, because of (4.36), we have that

$$\lim_{k\to\infty} c_1^k = c_1 = 0. \tag{4.38}$$

By definition, we have for index 1 and s

$$\frac{c_s^{k+1}}{c_1^{k+1}} = \frac{\left[\left(\frac{2\alpha^k}{\|v^k\|^2}(\lambda(v^k) - \lambda_s)\right) + \left(1 - \frac{4\alpha^k}{\epsilon\delta^2}(\|v^k\|^2 - 1)\right)\right]}{\left[\left(\frac{2\alpha^k}{\|v^k\|^2}(\lambda(v^k) - \lambda_1)\right) + \left(1 - \frac{4\alpha^k}{\epsilon}(\|v^k\|^2 - 1)\right)\right]} \frac{c_s^k}{c_1^k} < \frac{c_s^k}{c_1^k},$$

which shows that $\left\{\frac{c_s^k}{c_1^k}\right\}$ is decreasing. This fact is in contradiction with (4.37) and (4.38). It follows that $\hat{v}$ belongs to the eigenspace $\mathcal{Y}_1$ and hence is a global minimizer of problem (CP_1) and (RQ_1). □

In conclusion, a general gradient-like method applied to the exact penalty function r_ϵ can be viewed as an iterative method to find the extreme eigenvalue-eigenvector pair of a symmetric matrix.

In order to extend this unconstrained formulation for (CP_k), for $k > 1$, the idea is to consider the following merit function

$$rk_\epsilon(v) = \sum_{i=1}^{k} \beta_i \frac{v_i^T A v_i}{\|v_i\|^2} + \frac{1}{\epsilon} \sum_{i=1}^{k} \left[\frac{(\|v_i\|^2 - 1)^2}{d(v_i)} + \sum_{j=i+1}^{k} \left(\frac{v_i^T v_j}{\|v_i\| \|v_j\|} \right)^2 \right].$$

From the theoretical point of view, respect to the space where $rk_\epsilon(v)$ is defined, what we can show is that level sets of $rk_\epsilon(v)$ are compact and that stationary points have single block components (v_i) with unit norm. Unfortunately, we cannot force stationary points to satisfies exactly the orthogonality conditions, even forcing to zero the parameter ϵ. Nevertheless, we have the feeling that global solutions of $rk_\epsilon(v)$ (hence also stationary points) cannot be unfeasible for (CP_k), at least for ϵ sufficiently small. This conjecture is motivated by the fact the optimal solution of (CP_k) and the optimal value are both known. Moreover, also some preliminary tests confirm this sensation.

4.3 Numerical results

In this section, we describe our computational preliminary tests both with algorithm `LoREig` for solving problem (P), the extreme eigenvalue problem, and algorithm `RRQ` based on the minimization of (RQ_1) for finding the smallest eigenvalue of a symmetric matrix.

All the experiments have been run on the same PC with 4 Gb of RAM and 3.16 Ghz, with all algorithms implemented in Fortran.

The first group of experiments regards `LoREig` and takes as a test problem

$$\begin{aligned} \min_x \quad & \lambda_n \left(A_0 + \sum_{i=1}^{m} x_i A_i \right) \\ & u^T x = m \\ & x \geq 0, \end{aligned} \tag{4.39}$$

namely the minimization of the largest eigenvalue of a symmetric matrix, with the coefficients x constrained to the simplex set of length m. This problem is equivalent up to the sign to a particular (P) and it is completely defined by the dimensions m, n and the matrices $A_0, A_1, \ldots, A_m \in \mathbb{S}^n$.
In order of have large number of instances for our test, we use a random generator for sparse symmetric matrix. In particular, chosen m (number of variables) and n (dimension of the matrices), we use this random generator to create $m+1$ sparse matrix with density fixed to $d = 5\%$.
Therefore, our test bed is defined by 16 instances, given by combination of $m \in \{25, 59, 75, 100\}$ and $n \in \{250, 500, 750, 1000\}$.

Technical details for algorithm LoREig concern choices for r^0, for the optimization algorithm used to find KKT points for (LR$_r$) and for the routine to compute smallest eigenvalues.
As initial rank we set $r^0 = \max(3, \lfloor 0.4\hat{r}(m,n) \rfloor)$, where

$$\hat{r}(m,n) = \min\left(\left\lfloor \frac{\sqrt{1+8m}-1}{2} \right\rfloor, n \right)$$

is the upper bound on r^* for problem (4.39). Moreover, threshold value for switching from first to second-order algorithm is set directly to $\hat{r}(m,n)$. The updating rule for the rank is simply $r = \min\{\lfloor r \cdot 1.5 \rfloor, \hat{r}\}$.
As constrained optimization algorithm, we used the Fortran code AlGENCAN, which can be downloaded from the web page[1]. This code implements a special Augmented Lagrangian algorithm, which uses first and second-order derivatives and whose references can be found in [3, 4].
Finally, we use subroutines dsaupd and dseupd of the ARPACK library to compute the smallest eigenvalue of a sparse symmetric matrix.
In particular, as tolerances, we use for the inner minimization 10^{-8} and for the strong duality error just 10^{-7}.

As term of comparison with LoREig, we consider two very efficient Interior Point codes in Semidefinite Programming, SDPA and CSDP. References for SDPA can be found in [61, 62] and it can be downloaded in the version 7.3.5 from the web page[2], while CSDP refers to [10] and it can be downloaded in the version 6.1.0 from the web page[3]. Obviously, these two algorithms are applied to the SDP formulations for (4.39), namely

[1] http://www.ime.usp.br/~egbirgin/tango/codes

[2] http://sdpa.sourceforge.net/download.html

[3] https://projects.coin-or.org/Csdp

$$
\begin{array}{llcll}
\min\limits_{s,x} & s & \qquad & \max\limits_{Z,\mu} & m\,\mu + A_0 \bullet Z \\
& sI - \sum\limits_{i=1}^{m} x_i A_i \succeq A_0, & & & A_i \bullet Z - \mu \geq 0, \quad i = 1, \ldots, m \\
& u^T x = m, & & & I \bullet Z = 1, \\
& x \geq 0, & & & Z \succeq 0.
\end{array}
\tag{4.40}
$$

Moreover, we compare also with the more general Low-Rank method [14, 16, 17] described in Subsection 2.3.1 and implemented in the code `SDPLR` (downloadable from the web page[1]). The algorithm is applied to the second SDP problem in (4.40) written in primal standard form, namely

$$
\begin{array}{ll}
\max\limits_{Z,\mu^+,\mu^-,\theta} & m\,\mu^+ - m\,\mu^- + A_0 \bullet Z \\
& A_i \bullet Z - \mu^+ + \mu^- - \theta = 0, \quad i = 1, \ldots, m \\
& I \bullet Z = 1, \\
& \mu^+, \mu^-, \theta \geq 0, \\
& Z \succeq 0.
\end{array}
$$

SDP blocks are explicitly considered because `SDPLR` takes advantage of this structure.

In all test, for any instance and for any algorithm, we set a time limit of 10 hours to get a solution within the required accuracy, otherwise a failure is declared. Results of our tests are summarized in Table 4.1. For any instance, we report dimensions details and performances of each algorithm. In particular, for any algorithm, beyond the computational time, we also consider (respect to the primal-dual solution provided) the duality gap and the overall error on primal-dual feasibility. These measures allow to understand the accuracy of each method on solving (4.39).

`LoREig`, `CSDP` and `SDPA` solve all the test problems with times far enough from the time limit, whereas `SDPLR` is not able to converge for two instances and needs a long time for other three.

From the time point of view, among the SDP algorithms `SDPA` results slightly faster for small instances, while `CSDP` seems to be better for the larger ones. Anyway both of them are outperformed by `LoREig` as the latter is on average 4-5 times more rapid on solving the instances, small and large. It seems that `LoREig` is less influenced by the dimensions (particularly by n) comparing with the SDP approaches. Among the Low-Rank approaches, `LoREig` results sensibly faster than `SDPLR` for 13 out 16 instances and

[1] `http://dollar.biz.uiowa.edu/∼sburer/software/SDPLR`

dim		LoREig			SDPLR			CSDP			SDPA		
m	n	gap	feasible	time	gap	feasible	time	gap	feasible	time	gap	feasible	time
25	250	2.8e-08	6.9e-09	2	5.3e-06	3.8e-08	2	1.8e-08	9.9e-10	7	9.5e-08	7.6e-08	4
	500	2.3e-08	1.2e-09	12	1.6e-05	3.9e-08	52	2.0e-07	6.8e-09	57	2.7e-06	1.0e-08	64
	750	5.1e-09	8.5e-11	29	9.6e-05	3.9e-08	231	6.8e-08	1.8e-09	204	7.2e-07	4.7e-08	214
	1000	1.2e-08	1.3e-08	151	1.8e-05	4.5e-08	210	7.9e-08	1.8e-09	497	3.1e-06	7.8e-08	637
50	250	1.2e-09	4.2e-08	6	2.4e-06	4.7e-08	56	3.4e-07	9.5e-09	12	6.1e-07	2.4e-08	8
	500	5.4e-08	4.3e-08	26	2.1e-06	2.3e-08	100	2.6e-08	8.1e-10	104	2.3e-06	6.7e-08	101
	750	1.2e-07	1.7e-08	86	1.3e-04	3.6e-08	172	3.5e-08	9.0e-10	363	3.2e-06	7.6e-08	401
50	1000	9.6e-09	1.1e-08	126	1.6e-05	2.3e-08	181	4.8e-08	1.4e-09	834	5.3e-06	1.2e-08	943
75	250	2.7e-08	3.9e-08	9	3.2e-04	3.8e-08	6478	1.5e-08	4.7e-10	18	7.7e-07	1.8e-08	9
	500	1.9e-08	4.1e-09	55	1.0e-04	4.8e-08	30	3.5e-08	1.0e-09	152	3.3e-06	4.7e-08	96
	750	5.9e-09	2.1e-08	142	***	***	***	3.7e-07	4.9e-09	495	3.6e-06	2.8e-08	460
	1000	1.3e-07	6.8e-08	287	***	***	***	5.5e-08	1.1e-09	1192	4.4e-06	3.4e-08	1173
100	250	1.6e-08	6.2e-08	9	1.7e-05	4.9e-08	191	2.8e-08	7.1e-10	24	2.1e-06	1.6e-08	12
	500	5.0e-08	6.2e-09	91	5.1e-05	4.9e-08	63	1.3e-08	1.3e-09	195	8.7e-07	3.9e-08	126
	750	1.4e-07	1.3e-08	178	2.1e-05	4.9e-08	13005	5.8e-08	1.4e-09	654	3.5e-06	1.3e-08	559
	1000	6.7e-08	5.8e-08	461	3.1e-05	4.7e-08	14695	6.9e-08	1.3e-09	1624	5.2e-06	6.2e-08	1478

Table 4.1: First comparison over Extreme Eigenvalue Optimization

generally much more stable in terms of expected time. Moreover, quite often SDPLR is even slower than the two SDP algorithms.
From the accuracy point of view, LoREig compares favorably with both of SDP algorithms. Actually, it provides a primal-dual pair with an error on feasibility and on the strong duality of the same order of SDPA and slightly worse than CSDP. As opposed, SDPLR provides worse approximated optimal solution, as the error on strong duality is usually 2-3 order higher respect to the other algorithms.

In order to highlight limits of SDP approaches on solving (4.39), we provide additional results on another set of test problems, larger in terms of n: the random matrix generator is used to create 20 instances varying $m \in \{10, 20, 30, 40, 50\}$ and $n \in \{2000, 3000, 4000, 5000\}$. In particular, sparse matrices are generated with density fixed to $d = 1\%$.
Results of this second group of test are reported in Table 4.2. In this situation, only LoREig is able to solve all the problems within the time limit and never with more than one hour. As opposed, SDPLR fails for 2 instances, while CSDP and SDPA fail for a third of the problems. Morevoer, respect to LoREig, both SDP algorithms are on average 100 times slower, while SDPLR compares favorably only for few instances and is outperformed for the rest of problems. Considerations on accuracy are the same as before: LoREig, CSDP and SDPA have the same order of errors on strong duality and on feasibility, while SDPLR is not comparable.

Overall, among the first and second group of test, LoREig seems to very suitable to solve Eigenvalue Optimizations problems, combining speed with a satisfactory accuracy. Actually, it is extremely more fast and accurate respect to the SDP approaches and the general Low-Rank algorithm SDPLR. Respect to the latter our approach seems to be also more reliable and stable.

The second part of the experiments deals with the computation of the smallest eigenvalue of constant symmetric matrix for large-scale instances. In particular, we want to evaluate the computational behavior of the algorithm RRQ. We consider our approach with $\delta = 0.25$ and $\epsilon = 10^3 \cdot \delta^{-1}$ (parameters in (RQ$_1$)) and with the algorithm defined in [25] as a gradient-type unconstrained algorithm used to minimize (RQ$_1$) . This method satisfies (4.32),(4.33) and the stopping criterion is defined by

$$\|Av - \lambda(v)v\| \leq \rho(1 + |\lambda(v^0)|),$$

where v^0 is the starting point and ρ is the tolerance fixed to 10^{-8}.

dim		LoREig			SDPLR			CSDP			SDPA		
m	n	gap	feasible	time	gap	feasible	time	gap	feasible	time	gap	feasible	time
10	2000	1.2e-08	3.8e-09	93	1.5e-04	2.2e-08	28	2.7e-08	1.8e-09	2037	5.7e-07	2.3e-08	1773
	3000	1.1e-08	9.8e-09	53	4.2e-05	3.7e-08	60	1.2e-07	7.9e-09	6555	1.8e-07	7.0e-08	6075
	4000	1.3e-08	6.7e-09	166	1.5e-04	4.3e-08	263	6.2e-08	1.8e-09	16623	1.4e-06	6.4e-08	16699
	5000	2.3e-08	9.4e-09	327	2.3e-05	3.6e-08	537	4.7e-08	1.8e-09	31453	7.8e-07	6.9e-08	35105
20	2000	5.7e-09	1.4e-08	188	2.2e-05	4.8e-08	81	2.2e-07	7.8e-09	3239	8.7e-07	5.9e-08	4999
	3000	1.3e-09	7.3e-10	149	1.7e-05	4.9e-08	17600	1.5e-07	4.1e-09	11093	2.6e-07	7.9e-08	17225
	4000	6.2e-09	4.4e-09	497	5.3e-06	3.7e-08	7758	2.0e-07	5.4e-09	27463	***	***	***
	5000	1.6e-08	1.1e-08	413	1.6e-04	3.1e-08	739	***	***	***	***	***	***
30	2000	9.6e-09	1.2e-08	88	1.3e-04	2.4e-08	2267	2.1e-08	1.0e-09	4520	1.6e-06	2.7e-08	9333
	3000	3.1e-08	3.0e-08	350	4.0e-05	4.7e-08	1073	1.8e-08	5.5e-10	15027	5.1e-07	5.7e-08	32680
	4000	3.0e-08	2.3e-08	646	4.3e-05	4.6e-08	674	***	***	***	***	***	***
	5000	4.3e-08	2.8e-08	2061	7.1e-06	3.8e-08	7535	***	***	***	***	***	***
40	2000	2.7e-08	2.4e-08	188	3.2e-05	4.8e-08	211	6.0e-08	1.8e-09	5444	4.1e-07	2.9e-08	23887
	3000	1.4e-08	1.6e-09	831	3.6e-05	3.0e-08	2208	2.2e-07	3.6e-09	18782	***	***	***
	4000	1.9e-07	2.2e-08	2569	***	***	***	***	***	***	***	***	***
	5000	9.9e-08	1.2e-08	3337	6.3e-06	2.8e-08	2336	***	***	***	***	***	***
50	2000	4.8e-08	4.7e-08	358	5.5e-05	5.0e-08	2345	3.3e-08	1.3e-09	6526	5.6e-07	2.6e-08	24858
	3000	7.8e-08	1.8e-08	1143	8.2e-06	4.3e-08	17284	1.9e-07	2.4e-09	22980	***	***	***
	4000	2.4e-08	2.8e-08	1901	***	***	***	***	***	***	***	***	***
	5000	3.5e-08	3.1e-08	2590	1.5e-06	4.9e-08	861	***	***	***	***	***	***

Table 4.2: Second comparison over Extreme Eigenvalue Optimization

Instance		RRQ			Arpack		
n	val	residual	num Av	time	residual	num Av	time
10000	[-10,10]	3.10E-07	343	1.74	1.16E-06	978	3.08
	[-10,10]	5.54E-07	419	2.14	1.14E-06	990	3.12
	[-10,10]	5.76E-07	357	1.83	1.12E-06	894	2.81
	[-1000,1000]	3.19E-05	445	2.25	1.16E-04	978	3.08
	[-1000,1000]	4.31E-05	398	2.01	1.14E-04	990	3.14
	[-1000.1000]	5.69E-05	383	1.92	1.12E-04	894	2.84
50000	[-10,10]	9.83E-07	722	85.83	2.58E-06	3048	230.67
	[-10,10]	1.28E-06	1188	141.66	2.53E-06	3702	279.93
	[-10,10]	1.02E-06	685	81.24	2.56E-06	2628	196.39
	[-1000,1000]	4.35E-05	747	88.72	2.58E-04	3048	231.57
	[-1000,1000]	1.09E-04	1474	175.32	2.53E-04	3702	281.99
	[-1000,1000]	1.29E-04	573	68.19	2.56E-04	2628	198.49
100000	[-10,10]	1.15E-06	1197	599.06	3.64E-06	5478	1779.67
	[-10,10]	1.23E-06	1015	506.57	3.65E-06	3756	1211.28
	[-10,10]	1.80E-06	1323	659.28	3.64E-06	3354	1066.76
	[-1000,1000]	1.82E-04	1254	625.79	3.64E-04	5478	1751.88
	[-1000,1000]	1.64E-04	1096	543.99	3.65E-04	3756	1203.30
	[-1000,1000]	9.96E-05	953	474.67	3.64E-04	3354	1067.15

Table 4.3: Comparison over the Smallest Eigenvalue problem

As term of comparison, we consider the routines of Arpack used before. In particular, this software is based upon an algorithmic variant of the Arnoldi process called the Implicitly Restarted Arnoldi Method. Because the test matrices are symmetric it reduces to a variant of the Lanczos process called the Implicitly Restarted Lanczos Method. These variants may be viewed as a synthesis of the Arnoldi/Lanczos process with the Implicitly Shifted QR technique that is suitable for large scale problems.

To create our instances we use the same random generator for sparse symmetric matrix described before, but in this case with density $d = 1\%$. For any dimension $n \in \{10000, 50000, 100000\}$, we create 6 instances, the first three with entries randomly in $[-10, 10]$ and the remaining with entries in $[-1000, 1000]$.

In Table 4.3 are reported for any instance matrix details and performances of the two algorithms. In particular, the residual column refers to the error on the eigenvalue-eigenvector equation, namely

$$\|A\hat{v} - \hat{\lambda}\hat{v}\|,$$

where $(\hat{\lambda}, \hat{v})$ is the approximated smallest eigenpair provided. Moreover, we give also the total time and the number of calls of the routine for the sparse matrix-vector product.

The analysis of the tests is quite simple. On the one side, `RRQ` provides slightly more accurate solutions than `Arpack` and for both residuals get worse as the dimension and ill-conditioning increase. On the other side, our method results extremely faster than the other and this is confirmed by the lower number of sparse matrix-vector products.

In conclusion, according to this preliminary tests, we have that `RRQ` algorithm is suitable for computing smallest eigenvalues for large-scale matrices and that it outperforms an efficient code as the one implemented in the well-known `Arpack` library.

5

Integer Quadratic Programming

Motivated by the progress made in recent decades both in nonlinear optimization and in integer programming, the focus of research in optimization has recently moved to the study of mixed-integer nonlinear optimization problems. These problems are usually hard to be solved (in theory and in practice) by the presence of two types of nonconvexity: first, the objective function or constraints can be non-convex, and second, in the case of integer variables, the domains of the variables are non-convex.

We consider a general non-convex quadratic integer optimization problem of the form

$$\begin{aligned} \min_{x} \quad & q(x) = x^T Q x + L^T x \\ & l_i \le x_i \le u_i \quad i = 1, \dots, n \\ & x \in \mathbb{Z}^n, \end{aligned} \tag{IQP}$$

where $l, u \in \mathbb{R}^n$ define the box where variables are constrained.
Almost all the algorithms for quadratic mixed-integer optimization can handle exactly just the case where the objective function is convex, see e.g. [9, 13, 38]. In case this condition is not satisfied, such algorithms can be somehow used, but the provided solution is usually suboptimal.
As far we know, among exact algorithms that cover at least non-convex quadratic mixed-integer programming, there are two based on the idea of convex estimators combined with branching and bound reduction techniques [6, 55], and one more recent approach based on a tight SDP relaxation embedded in a branch-and-bound scheme [18].

Recently, a fast branch-and-bound algorithm for convex quadratic integer minimization has been proposed in [13]. Its main features are a fast incremental computation of lower bounds given by unconstrained continuous minimizers and an improvement of these bounds by considering lattice-free ellipsoids containing the continuous minimizers. More precisely, the improved lower bound is given as the minimum of the objective function over the boundary of such ellipsoid, which can be computed efficiently.

For a non-convex quadratic objective function, this approach is not feasible any more; the unconstrained continuous minimizer does not even exist in this case.
In this chapter, we present a different approach for computing lower bounds in the non-convex case. Main ingredients in the algorithm are the definition and the solution of an appropriate continuous relaxation of problem (IQP). To this end, we choose an ellipsoid E enclosing entirely the feasible region of problem (IQP). Then we define a relaxation of problem (IQP) as

$$\begin{aligned} \min_{x} \quad & x^T Q x + L^T x \\ & x \in E. \end{aligned} \tag{R}$$

Taking advantage of dimension restriction for (IQP), the relaxation can be solved from the dual point of view in a very small amount of time. Actually, once a spectral decomposition of Q is available, the dual of (R) can be solved by a (univariate) Newton method, with cost per iteration proportional to n.
Moreover, embedding the relaxation (R) in a branch-and-bound scheme with a priori order on fixing variables, the spectral decomposition of a sub-matrix for a certain subproblem is in common with many other nodes. This implies that the number of spectral decompositions that needs to be performed is at maximum n and all these tasks can be moved in the preprocessing phase.

This chapter is organized as follows. In Section 5.1, we describe our approach for solving the continuous relaxation (R) as well as two ways to choose good relaxations. In Section 5.2, we explain how the solution of this relaxation can be embedded into a branch-and-bound scheme in an intelligent way. In Section 5.3, we report computational results of our algorithm on an extensive test problem set for the ternary case.

5.1 Continuous relaxations

We are interested in finding strong lower bounds for problems of the same type of (IQP) that are efficiently computable. The method we apply depends on whether the objective function $q(x)$ is convex or non-convex, i.e., whether Q is positive semidefinite or not. We describe these cases separately in the following.

5.1.1 Non-convex case

In this part we focus on an integer quadratic problem

$$\begin{aligned} \min_x \quad & q(x) = x^T Q x + L^T x \\ & x \in \mathbb{Z}^n \cap [l, u], \end{aligned} \tag{INC}$$

with $Q \not\succeq 0$. In this non-convex case, we aim to solve the continuous relaxation given by relaxing integrality and box constraints. In particular, we make use of an ellipsoid that contains the entire feasible region of (INC).

Since integrality is not required here, we simplify this problem in several ways. First, we may assume $l = -1$ and $u = 1$, by properly scaling Q and L as explained in Subsection 5.1.1.3. For simplicity, we assume Q and L already scaled. Moreover, we restrict ourselves to ellipsoids that are centered in the origin, hence defined as

$$E(H) = \{x \in \mathbb{R}^n : \ x^T H x \leq n\},$$

by an appropriate $H \succ 0$. In particular, H must satisfy the following assumption

$$[-1, 1]^n \subseteq E(H),$$

in order to have a valid relaxation of (INC) (of course already scaled). The addressed problem becomes

$$\begin{aligned} \min_x \quad & x^T Q x + L^T x \\ & x \in E(H). \end{aligned}$$

By applying a linear transformation with $H^{-\frac{1}{2}}$ to the variable space, we can further assume $H = I$, replacing Q by $H^{-\frac{1}{2}} Q H^{-\frac{1}{2}}$ and L by $H^{-\frac{1}{2}} L$. So, without loss of generality, we focus on the relaxation defined on the sphere, namely

$$\begin{aligned} \min_x \quad & x^T Q x + L^T x \\ & \|x\|^2 \leq n. \end{aligned}$$

By assumption on $q(x)$ ($Q \not\succeq 0$), global minimizers x^* can be found on the boundary of the feasible set. Actually, assume by contradiction that x^* lies in the interior of the feasible region, then x^* would satisfy the first and second-order local optimality conditions for unconstrained optimization. The latter would be in conflict with $Q \not\succeq 0$. Therefore, the relaxation reduces simply to problem

$$\begin{aligned} p^* = \min_x \quad & x^T Q x + L^T x \\ & \|x\|^2 = n. \end{aligned} \tag{P}$$

Despite its non convexity, it is well known that optimal solution set of problem (P) can be fully characterized by some optimality conditions (see, e.g.,[60]): a point x^* is a global minimizer for (P) if and only if there exists $\mu^* \in \mathbb{R}$ such that

$$\begin{cases} 2(Q - \mu^* I)x^* = -L, \\ \|x^*\|^2 = n, \\ Q - \mu^* I \succeq 0. \end{cases} \tag{5.1}$$

First two conditions correspond to the standard KKT condition for problem (P). Because of the regularity of the feasible set, the optimal multiplier μ^* is unique respect to x^*. Further if $Q - \mu^* I \succ 0$ then problem (P) has a unique global solution.
Most important, it has also been proved that an approximation to the global solution can be computed in polynomial time (see, for example, [60]). Therefore, (P) can be considered an easy problem from a theoretical point of view. These peculiarities led to the development of ad hoc algorithms for finding a global solution for (P).

In truth, most of the algorithms proposed in literature [45, 52, 58] are based on duality results: the Lagrangian dual for problem (P), as defined in [58], is given by

$$\begin{aligned} d^* = \max_{\mu} \quad & f(\mu) = n\mu - \tfrac{1}{4} L^T (Q - \mu I)^\dagger L \\ & Q - \mu I \succeq 0. \end{aligned} \tag{D}$$

where $(\cdot)^\dagger$ is the matrix operator for the generalized inverse, better called pseudo-inverse. It can be shown that strong Lagrangian duality holds, i.e., $p^* = d^*$. Problem (D) has been deeply studied in nonlinear continuous optimization field and the main aim was the definition of efficient algorithms able to treat large scale problem and to exploit sparsity pattern. However, within a branch-and-bound scheme for an integer problem, this is not the case in our context: the dimension of the addressed problems is usually below one hundred. Therefore, some expensive preliminary operations, such as spectral decompositions of the quadratic term, can be easily accomplished.

In the following, we take advantage of the availability of the spectral decomposition of Q: let $\lambda_1, \ldots, \lambda_n$ the eigenvalues of Q, in not decreasing order, and $v_1, \ldots, v_n$ the corresponding eigenvectors (assumed orthonormalized). Therefore, we can express

$$Q = V \Lambda V^T,$$

where $V = [v_1 \ldots, v_n]$, with $V^T V = I$, and $\Lambda = \mathrm{Diag}(\lambda_1, \ldots, \lambda_n)$. By employing the transformation $y = V^T x$, we reformulate (P) as

$$\begin{aligned} p^* = \min_{y} \quad & y^T \Lambda y + \tilde{L}^T y \\ & \|y\|^2 = n, \end{aligned} \tag{DP}$$

where $\tilde{L} = P^T L$. In this situation, the cost per function evaluation reduces from $\mathcal{O}(n^2)$ to $\mathcal{O}(n)$. The diagonal form simplify the global optimality conditions, that are

$$\begin{cases} 2(\lambda_i - \mu)y_i = -\tilde{L}_i, \quad i = 1, \dots, n \\ \displaystyle\sum_{i=1}^{n} y_i^2 = n, \\ \mu \leq \lambda_1. \end{cases} \tag{5.2}$$

The spectral decomposition of Q can be also exploited to reduce the complexity of the dual, which can be rewritten as

$$\begin{aligned} d^* = \max_{\mu} \quad & f(\mu) = n\mu - \tfrac{1}{4}\tilde{L}^T(\Lambda - \mu I)^{\dagger}\tilde{L} \\ & \mu \leq \lambda_1. \end{aligned} \tag{D}$$

We recall that for a diagonal matrix we get the pseudo-inverse by taking the reciprocal of each non-zero element on the diagonal, leaving the zeros in place. This fact underlines that the pseudo-inverse is not a continuous operation: slight changes on the zero diagonal entries result in significant changes on the pseudo-inverse.
Based on $\tilde{L}$, the projection of the linear part L in the eigenspaces of Q, we define the following sets of indexes

$$\begin{aligned} I &= \left\{ i \in \{1, \dots, n\} : \tilde{L}_i \neq 0 \right\}, \\ J &= \{ i \in I : \lambda_i = \lambda_1 \}, \\ K &= \{ i \in I : \lambda_i > \lambda_1 \}. \end{aligned}$$

These sets allow to rewrite the dual function $f(\mu)$ in this way

$$f(\mu) = \begin{cases} n\mu - \displaystyle\sum_{i \in I} \frac{\tilde{L}_i^2}{4(\lambda_i - \mu)} & \text{if } \mu < \lambda_1, \\ n\mu - \displaystyle\sum_{i \in K} \frac{\tilde{L}_i^2}{4(\lambda_i - \mu)} & \text{if } \mu = \lambda_1, \end{cases}$$

in the interval $(\infty, \lambda_1]$. Therefore, for $\mu < \lambda_1$, we have first and second order derivatives expressed as

$$f'(\mu) = n - \sum_{i \in I} \frac{\tilde{L}_i^2}{4(\lambda_i - \mu)^2} \tag{5.3}$$

$$f''(\mu) = -\sum_{i \in I} \frac{\tilde{L}_i^2}{2(\lambda_i - \mu)^3}, \tag{5.4}$$

so that $f(\mu)$ is strictly concave in the interval (∞, λ_1). Likewise, $f'(\lambda_1)$ and $f''(\lambda_1)$ are similar to (5.3) and (5.4), just the sum is restricted to K instead of I.
Observe that $f(\mu)$ is always twice continuously differentiable in $(-\infty, \lambda_1)$. In case $J = \emptyset$, $f(\mu)$ is twice continuously differentiable in the larger space $(-\infty, \lambda_1]$, because I equals K.

Now we are ready to characterize the optimal solutions of (D) and (DP).

Proposition 5.1.1 *Respect to μ^*, the optimal solution of* (D)*, we have one of the following situations:*

(i) *If $J \neq \emptyset$, then $\mu^* < \lambda_1$ is given by $f'(\mu^*) = 0$.*

(ii) *If $J = \emptyset$ and $f'(\lambda_1) < 0$, then $\mu^* < \lambda_1$ is given by $f'(\mu^*) = 0$.*

(iii) *If $J = \emptyset$ and $f'(\lambda_1) \geq 0$, then $\mu^* = \lambda_1$.*

Proof Because μ^* is optimal for (D), there exists y^* such that conditions (5.2) are satisfied.
Consider case (i) with $J \neq \emptyset$. Without loss of generality, we can assume $\tilde{L}_1 \neq 0$. It is not hard to see that global optimality conditions (5.2) can not be satisfied with $\mu = \lambda_1$, so that $\mu^* < \lambda_1$. Moreover, because f is strictly concave, μ^* is given by $f'(\mu^*) = 0$.
As opposed, consider situations where $J = \emptyset$. Because f is strictly concave, if $f'(\lambda_1) < 0$ then $f(\mu) > f(\lambda_1)$ for any $\mu < \lambda_1$. It follows that $\mu^* < \lambda_1$ and $f'(\mu^*) = 0$, so that point (ii) is proved. On the contrary, if $f'(\lambda_1) \geq 0$ then $f(\mu) < f(\lambda_1)$ for any $\mu < \lambda_1$. Hence $\mu^* = \lambda_1$ and also last point follows. □

Corollary 5.1.2 *Given μ^* the optimal solution of* (D)*, define $y(\mu^*) \in \mathbb{R}^n$ as*

$$y_i(\mu^*) = \begin{cases} -\frac{\tilde{L}_i}{2(\lambda_i - \mu^*)} & \textit{if } i \in K \\ -\frac{\tilde{L}_i}{2(\lambda_i - \mu^*)} & \textit{if } i \in I - K \textit{ and } \mu^* < \lambda_1 \\ 0 & \textit{otherwise} \end{cases} \qquad i = 1, \ldots, n.$$

If $f'(\mu^*) = 0$ then $y(\mu^*)$ is optimal for (DP).

Proof Expression of $y(\mu^*)$ follows from first condition in (5.2). Moreover, if

$$0 = f'(\mu^*) = n - \sum_{i=1}^{n} y_i(\mu^*),$$

all conditions in (5.2) are satisfied. It follows that $y(\mu^*)$ is optimal for (DP). □

Proposition 5.1.1 and Corollary 5.1.2 give a solution strategy to find an optimal solution μ^* of (D) and probably y^* optimal solution of (DP):

1. If $J = \emptyset$ and $f'(\lambda_1) \geq 0$ then $\mu^* = \lambda_1$. Moreover, if $f'(\mu^*) = 0$ then $y^* = y(\mu^*)$.
2. Find $\mu^* < \lambda_1$ such that $f'(\mu^*) = 0$, using the algorithm presented in the Subsection 5.1.1.1. Then compute $y^* = y(\mu^*)$.

From the point of view of relaxation for (INC), we remark that in second case we do not need to find exactly the stationary point. Actually, by weak duality, any $\mu \leq \lambda_1$ provides a safe lower bound for (INC).

5.1.1.1 Solution approach

In the situations where the optimal solution $\mu^* < \lambda_1$, we need a numerical method to find the zero of the derivative of the dual function in the right interval. As we said before, problem (P) and its dual were deeply studied in nonlinear optimization, so that many different efficient algorithms were developed for their solution. Our algorithm is basically the one defined in [45] with a drastic reduction of many operations, thanks to the explicit use of the spectral decomposition of Q.

As result of Propositions 5.1.1 and 5.1.2, the optimal solution $\mu^* < \lambda_1$ is given by the zero of

$$f'(\mu) = n - \sum_{i \in I} \frac{\tilde{L}_i^2}{4(\lambda_i - \mu)^2} = n - \|y(\mu)\|^2,$$

where we denote $g(\mu) = \|y(\mu)\|^2$. As opposed, the solution approach proposed in [45] suggested the use a dumped Newton method to find a zero of

$$\phi(\mu) = \frac{1}{\sqrt{n}} - \frac{1}{\sqrt{g(\mu)}},$$

in the interval $(-\infty, \lambda_1)$. Main motivation for this change is that $f'(\mu)$ is highly ill-conditioned nearby λ_1, while $\phi(\mu)$ is almost linear in $(-\infty, \lambda_1)$. Moreover

$$\phi'(\mu) = \frac{1}{2} g(\mu)^{-\frac{3}{2}} g'(\mu),$$

so that $\phi(\mu)$ is strictly increasing in $(-\infty, \lambda_1)$ and also strictly convex, as a combination of strictly convex functions.
In a Newton framework, given a point $\mu < \lambda_1$ the next point μ^+ is computed as

$$\mu^+ = \mu + d_\mu = \mu - (\phi'(\mu))^{-1}\phi(\mu) = \mu - \frac{2g(\mu)}{g'(\mu)} \left(\frac{\sqrt{g(\mu)} - \sqrt{n}}{\sqrt{n}} \right).$$

Of course, μ^+ generated by a pure Newton step can be outside of $(-\infty, \lambda_1)$, so that we need to safeguard Newton iterates. Fortunately, we can exploit current informations: because $\phi(\mu)$ is convex and strictly increasing for $\mu < \lambda_1$, if $\phi(\mu) > 0$ then Newton method produces a monotonically decreasing sequence converging to the zero of ϕ, otherwise $\mu^+ \geq \lambda_1$ or $\phi(\mu^+) \geq 0$. In the second case the new point may need to be redirect in $(-\infty, \lambda_1)$.
Let μ_l, μ_u such that $[\mu_l, \mu_u]$ is an interval of uncertainty which contains the optimal solution μ^*. Observe that $\mu_l \leq \mu^* < \lambda_1 < 0$ and $\mu_u \leq \lambda_1 < 0$. Then, we can **safeguard** μ in this way:

1. $\mu = \max(\mu, \mu_l)$;

2. $\mu = \min(\mu, \mu_u)$;

3. if $\mu \geq \lambda_1$ then $\mu = \max(1.001\mu_u, -\sqrt{\mu_l \mu_u})$.

The first two steps help on keeping μ in $[\mu_l, \mu_u]$, while the third step forces the interval to be reduced. If the length of interval remains bounded for zero, the third step can be applied just a finite number of times.
Therefore, it is essential to **update** μ_l, μ_u during the algorithm:

if $\mu < \lambda_1$ and $\phi(\mu) < 0$ then $\mu_l = \max(\mu, \mu_l)$,

else $\mu_u = \min(\mu, \mu_u)$.

Initial value for μ_l, μ_u are based on the following condition

$$\frac{\|\tilde{L}\|}{2(\lambda_n - \mu)} - \sqrt{n} \leq \|y(\mu)\| - \sqrt{n} \leq \frac{\|\tilde{L}\|}{2(\lambda_1 - \mu)} - \sqrt{n}.$$

If $\mu < \lambda_1 - \frac{\|\tilde{L}\|}{2\sqrt{n}}$ then $\|y(\mu)\| < \sqrt{n}$. It follows that $\mu_l = \lambda_1 - \frac{\|\tilde{L}\|}{2\sqrt{n}}$.

If $\mu > \lambda_n - \frac{\|\tilde{L}\|}{2\sqrt{n}}$ then $\|y(\mu)\| > \sqrt{n}$. It follows that $\mu_u = \min\left(\lambda_n - \frac{\|\tilde{L}\|}{2\sqrt{n}}, \lambda_1\right)$.

In summary, we get the following algorithm, which we stress it is a simplification of the algorithm given in [45] designed to compute the trust-region direction in nonlinear programming.

Newton method for (D)

```
Data. $\tilde{L}$, $\lambda_1, \dots, \lambda_n$, $\rho > 0$, $\mu^0$.
Initialization. $\mu_l = \lambda_1 - \frac{\|\tilde{L}\|}{2\sqrt{n}}$, $\mu_u = \min\left(\lambda_n - \frac{\|\tilde{L}\|}{2\sqrt{n}}, \lambda_1\right)$,
For $k = 0, 1, \dots$
    1. Safeguard $\mu^k$.
    2. Compute $f(\mu^k)$ and $\phi(\mu^k)$.
    3. Update $\mu_l, \mu_u$.
    4. If $\left|\phi(\mu^k)\right| \le \rho$ then stop.
    5. Compute $\mu^{k+1} = \mu^k - (\phi'(\mu^k))^{-1}\phi(\mu^k)$.
End For
```

From the point of view of relaxation for (IQP), we make the following considerations:

- we keep in memory scalar μ_M which maximizes $\max f(\mu^k)$;
- the algorithm is iterated for a maximum number of iterations;
- the algorithm is forced to quit whenever $f(\mu_M) > z_{ub}$, where z_{ub} is an upper bound on (INC).

5.1.1.2 Choice of the Ellipsoid

For sake of description, so far we have simply considered the relaxation of (INC) defined by the sphere. In truth, this may not be the best choice. For instance, consider the small example in Figure 5.1: 2 variables restricted to $\{-1, 0, 1\}$ values and an indefinite quadratic function. The relaxation can be strengthened squeezing the sphere to a more flat ellipsoid, so that the lower bound is improved. Moreover, the gain obtained by a more general ellipsoid is not paid by an extra-cost for solving the corresponding relaxation. Actually, we recall that from the solution point of view there no difference between the sphere and a general ellipsoid.

In this part, we describe 2 rules to choose the ellipsoid H defining the relaxation for (INC). In particular, we would like to have $H \succ 0$ such that

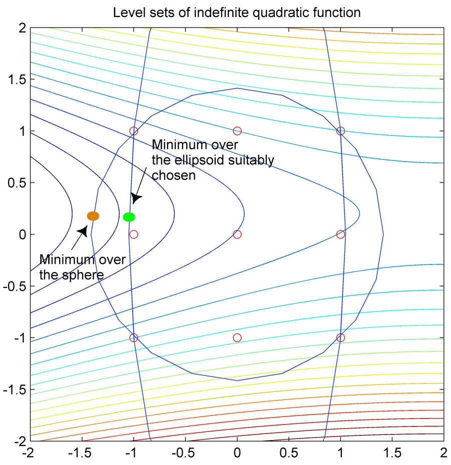

Figure 5.1: Lower Bounds - Different ellipsoids enclosing the integer feasible region

(c1) $[-1, 1]^n \subseteq E(H)$;

(c2) $p^*(H) = \min\{q(x) : \; x \in E(H)\}$ being as high as possible;

(c3) H is computed in a reasonable time.

In order to simplify this choice, we restrict our attention to diagonal H because point (c1) is easier to be satisfied. Actually, if we consider H in

$$\mathcal{H}_\epsilon = \{H \in \mathbb{D}^n : \; I \bullet H = n, \, H \geq \epsilon\},$$

with parameter $\epsilon \in (0, 1]$, then $H \succ 0$ and (c1) is easily satisfied.

The first rule is based on condition (c2): the choice of H is driven by the problem

$$\max_{H \in \mathcal{H}_\epsilon} p^*(H). \tag{5.5}$$

The set $\mathcal{H}_\epsilon$ is a simplex set, so convex, compact and non-empty. Function $p^*(H)$ is not continuously differentiable at points $H \in \mathcal{H}_\epsilon$ where the corresponding problem admits more that one optimal solution. For a given $H \in \mathcal{H}_\epsilon$, if x^* is a primal optimal solution of the related problem and μ^* the associated multiplier, then the following conditions are satisfied

$$\begin{cases} 2(Q - \mu^* H)x^* = -L, \\ {x^*}^T H x^* = n, \\ Q - \mu^* I \succeq 0, \end{cases}$$

which represent a generalization of (5.1) for a generic ellipsoid H. The scalar product between the first condition and x^* gives

$$\mu^* = \frac{2x^{*T}Qx^* + L^T x^*}{2x^{*T}Hx^*} = \frac{2x^{*T}Qx^* + L^T x^*}{2n}.$$

Combining this closed-form expression for μ^* with the feasibility of x^*, we rewrite

$$p^*(H) = q(x^*) + \mu^*(n - x^{*T}Hx) = \frac{L^T x^*}{2} + n\frac{2x^{*T}Qx^* + L^T x^*}{2x^{*T}Hx^*}.$$

Therefore, an element of the generalized gradient then is given by

$$S_{ii}(H) = -n\frac{2x^{*T}Qx^* + L^T x^*}{2(x^{*T}Hx^*)^2}x_i^{*2} = -\mu^*\ x_i^{*2} \quad i = 1, \ldots, n, \tag{5.6}$$

where just the diagonal components need to be expressed as $\mathcal{H}_\epsilon \subset \mathbb{D}^n$.

Moreover, we conjecture on the concavity of $p^*(H)$ over $\mathcal{H}_\epsilon$, because of the definition as a minimization problem, although we were not able to prove it yet. If this were true, then (5.6) would define a supergradient for $p^*(H)$.

The idea to solve (very) approximately problem (5.5) is to use the Subgradient method briefly described in Appendix A, specialized for simplex sets as $\mathcal{H}_\epsilon$. The uncertainty on concavity of the objective function, as well as limits of accuracy for Subgradient methods, suggests to understate the maximization. Therefore, the final ellipsoid H can be chosen as the best point produced (in terms of $p^*(H)$) after a limited number of iterations.

The second rule to choose H is related to (c3), namely a cheap way to compute the ellipsoid. If we assume by restriction that the linear part in $q(x)$ does not appear, then problem (5.5) reduces to

$$\max_{H \in \mathcal{H}_\epsilon} \lambda_1(Q, H), \tag{5.7}$$

where $\lambda_1(Q, H)$ is the generalized smallest eigenvalue for the pencil (Q, H). By definition this is equivalent to

$$\begin{aligned} \max_{H,t} \quad & t \\ & Q - tH \succeq 0, \\ & H \in \mathcal{H}_\epsilon. \end{aligned} \tag{5.8}$$

Because $Q \not\succeq 0$ then $\lambda_1(Q, H) < 0$ and hence any optimal solution of (5.8) must have $t^* < 0$. Making the transformation $\mu = -\frac{1}{t}$, we rewrite (5.8) as

$$\begin{aligned} \max_{H,t} \quad & \mu \\ & \mu Q + H \succeq 0 \\ & H \in \mathcal{H}_\epsilon. \end{aligned} \tag{5.9}$$

This problem is a linear SDP problem, where strict feasibility holds for it and for its dual. Therefore, the second rule is given by applying an SDP Interior Point method to (5.9), in order to find a solution for (5.7).

5.1.1.3 Scaling quadratic problems

Consider the integer quadratic problem

$$\begin{aligned} \min_x \quad & q(x) = x^T Q x + L^T x \\ & x \in [l, u] \cap \mathbb{Z}^n, \end{aligned}$$

and the continuous relaxation

$$\begin{aligned} \min_x \quad & q(x) = x^T Q x + L^T x \\ & x \in [l, u]. \end{aligned}$$

The relaxation can be rewritten as

$$\begin{aligned} \min_x \quad & q(x) = x^T \bar{Q} x + \bar{L}^T x + \bar{c} \\ & x \in [-1, 1]^n, \end{aligned}$$

using the transformation of variables $t : [l, u] \to [-1, 1]^n$ defined as

$$t_j(x) = \frac{2x_j - (l_j + u_j)}{u_j - l_j}, \quad j = 1, \dots, n.$$

The inverse operation $t^{-1} : [-1, 1]^n \to [l, u]$ is given by

$$t_j^{-1}(x) = \frac{y_j(u_j - l_j) + (l_j + u_j)}{2}, \quad j = 1, \dots, n.$$

By simple operations, the matrix $\bar{Q} \in \mathbb{R}^{n \times n}$ is given by

$$\bar{Q} = [\bar{Q}_{ij}] = \left[\frac{Q_{ij}}{4}(u_i - l_i)(u_j - l_j) \right],$$

while the linear term $\bar{L} \in \mathbb{R}^n$ is computed as

$$\bar{L} = [\bar{L}_i] = \left[\sum_{j=1}^{n} \frac{Q_{ij}}{2}(u_i - l_i)(u_j + l_j) + \frac{L_i}{2}(u_i - l_i) \right].$$

The remaining terms merge into the constant $\bar{c}$ which takes the form

$$\bar{c} = \sum_{i=1}^{n} \sum_{j=1}^{n} \frac{Q_{ij}}{4}(u_i + l_i)(u_j + l_j) + \sum_{i=1}^{n} \frac{L_i}{2}(u_i + l_i).$$

5.1.2 Convex case

In this section, we are interested in an integer convex quadratic problem

$$\begin{aligned} p^* = \min_x \quad & q(x) = x^T Qx + L^T x \\ & x \in \mathbb{Z}^n \cap X, \end{aligned} \tag{IC}$$

so with $Q \succeq 0$ and where $X = [l, u]$. In this situation, we apply exactly the techniques presented in [13]. For simplicity, we can assume $Q \succ 0$ and that are available Q^{-1}, the Cholesky factor B of Q and its inverse B^{-1}.

The main ingredient is the computation of the lower bound obtained by neglecting all the constraints, including integrality. Possibly, the lower bound can be improved by exploiting the integrality of the variables using suitably-defined lattice-free ellipsoids.

Given $H \succ 0$, $\hat{x} \in \mathbb{R}^n$ and $\alpha \in \mathbb{R}$, we consider the following ellipsoid

$$E(H, \hat{x}) = \left\{x \in \mathbb{R}^n : \ (x - \hat{x})^T H(x - \hat{x}) \leq 1\right\},$$

and the same ellipsoid scaled by α, namely

$$E(H, \hat{x}, \alpha) = \left\{x \in \mathbb{R}^n : \ (x - \hat{x})^T H(x - \hat{x}) \leq \alpha^2\right\}.$$

Solving the optimization problem

$$\mu(H, \hat{x}) = \min_\alpha \{\alpha : \ E(H, \hat{x}, \alpha) \cap \mathbb{Z}^n \cap X \neq \emptyset\}$$

means finding the scaling factor α such that $E(H, \hat{x}, \alpha)$ contains some integer feasible point on its border but no integer feasible point in its interior.
Defined the unconstrained minimizer of $q(x)$, namely $\bar{x} = -Q^{-1}L/2$, then (IC) is equivalent to compute $\mu(Q, \bar{x})$. Actually, the level sets of $q(x)$ are the border of the ellipsoid $E(Q, \bar{x}, \alpha)$ for some α. It is easy to visualize the fact that finding the integer point in X that minimizes q is equivalent to increase the scaling factor α from 0 and to stop as soon as the border of $E(Q, \bar{x}, \alpha)$ contains an integer feasible point.
Therefore, we can not use Q to define the ellipsoid, because $\mu(H, \bar{x})$ is hard to compute as solving (IC).

In order to simplify the procedure, the same scaling is done for some other ellipsoid $E(H, \bar{x})$ and then $E(Q, \bar{x})$ is scaled in turn until it touches the border of the first ellipsoid. This requires one is able to compute $\mu(H, \bar{x})$ as well as the maximum α such that $E(Q, \bar{x}, \alpha)$ is contained in $E(H, \bar{x})$, that is

$$\lambda(Q, H) = \max_\alpha \{\alpha : \ E(Q, 0, \alpha) \subseteq E(H, 0)\}.$$

Leaving out computational details, we have the following valid lower bound for (IC)(see [13]): given any $H \succ 0$

$$p^* \geq f(\bar{x}) + \lambda^2(Q, H)\mu^2(H, \bar{x}).$$

Difficulties hide on the choice of H and the computation of each term defining the above lower bound.

In equivalent way $\lambda(Q, H)$ can be computed as the square root of

$$\max_{\alpha} \{\alpha : \ Q - \alpha H \succeq 0\}, \tag{5.10}$$

a linear SDP problem which can be done efficiently. Moreover, in order to have stronger lower bounds, (5.10) shows that H should be chosen as similar as possible to Q. Nevertheless, the main guideline for choosing H should be that $\mu(H, \bar{x})$ is easy to be computed.

In [13] it has been shown that for $t \in \mathbb{Z}^n/\{0\}$ a computable lower bound on $\mu(tt^T, \bar{x})$ is given, namely

$$\mu(tt^T, \bar{x}) \geq \bar{\mu}(t, \bar{x}) = \max\left\{|t^T\bar{x} - \lfloor t^T\bar{x}\rceil|, \ t^T\bar{x} - s, \ z - t^T\bar{x}\right\},$$

where s and z are respectively the maximum and the minimum of t^Tx over X(or better over $X \cap \mathbb{Z}^n$).
On the other side, one would like to find $t \in \mathbb{Z}^n/\{0\}$ such that $\lambda(Q, tt^T)$ is large as possible. This corresponds to a flat direction, namely a direction along it is minimized the width of $E(Q, 0)$. Because the width of $E(Q, 0)$ along t is given by $\|t^TB^{-1}\|$, finding a flat direction is equivalent to finding $t \in \mathbb{Z}^n/\{0\}$ yielding a shortest non-zero vector in the lattice generated by the columns of $(B^{-1})^T$, which is well-known to be $\mathcal{NP}-$Hard.

A natural heuristic to compute short vectors is obtained by taking as candidates the vectors in a reduced basis of the lattice. Let $t_1, \ldots, t_n \in \mathbb{Z}^n/\{0\}$ be the columns of the corresponding transformation matrix T, from the original basis to the reduced basis. In particular, as proposed in [13], a good ellipsoid H is defined as

$$H = \sum_{i=1}^{n} \lambda^2(Q, t_it_i^T)t_it_i^T,$$

because of the following result

$$\mu(H, \bar{x}) \geq \hat{\mu}(T, \bar{x}) = \sqrt{\sum_{i=1}^{n} \lambda^2(Q, t_it_i^T)\bar{\mu}^2(t_i, \bar{x})}.$$

5.2 Branch-and-Bound scheme

The straightforward way to use the results of Section 5.1 within a branch-and-bound scheme is the following: branch on a variable x_j by fixing it to a particular value within its domain $\{l_j, \ldots, u_j\}$. The resulting problem is of the same type as (IQP), except that the dimension decreases by one. In particular, we compute a lower bound in each node by solving the corresponding continuous relaxation (R). Clearly, this approach yields an algorithm to solve problem (IQP) to global optimality in finite time, as every variable has to be fixed to a finite number of variables.

In order to enumerate subproblems as quickly as possible, our aim is to perform the most time-consuming computations in a preprocessing phase. For this purpose we define a branch-and-bound scheme with a priori order on fixing variables. For simplicity, we assume the order $1, 2, \ldots, n-1, n$.

Given a certain node s of the branch-and-bound tree, at level $d \in \{0, \ldots, n-1\}$, the first d variables will be already fixed to integer values, $x_i = r_i^{(s,d)}$ with $i = 1, \ldots, d$. The reduced objective function $q^{(s,d)} : \mathbb{R}^{n-d} \to \mathbb{R}$ is of the form

$$q^{(s,d)}(x) = q(r^{(s,d)}, x) = x^T Q^{(d)} x + {L^{(s,d)}}^T x + c^{(s,d)},$$

where

$$L_{j-d}^{(s,d)} = L_j + 2 \sum_{i=1}^{d} q_{ij} r_i^{(s,d)}, \quad j = d+1, \ldots, n,$$

and

$$c^{(s,d)} = c + \sum_{i=1}^{d} L_i r_i^{(s,d)} + \sum_{i=1}^{d} \sum_{j=1}^{d} q_{ij} r_i^{(s,d)} r_j^{(s,d)}.$$

As opposed, the matrix $Q^{(d)}$ does not depend on the values at which the first d variables are fixed and it is obtained from Q by deleting the first d rows and columns. Therefore, $Q^{(d)}$ does not depend on s, but just on the level of depth relative to the node. If follows that several nodes at the same level share the same quadratic term. Likewise, let $L^{(d)}$ be the common part in the linear term in common with nodes at the same level d. In particular $L^{(d)}$ is obtained from L by deleting the first d components.

As described in Section 5.1, in order to solve the relaxation of the integer subproblem at the node, either for the convex or non-convex case, some expensive operations on $Q^{(d)}$ need to performed in order to simplify the solution of the relaxation. For this reason we do not change the order of fixing variables, i.e., we always fix the first unfixed variable according to an order that is determined before starting the enumeration. This implies that in total we only have to consider n different matrices $Q^{(d)}$, which we know in advance as soon as the fixing order is determined. If the variables to be fixed were chosen freely, the number of such matrices would be exponential.

As typical in a branch-and-bound scheme, we consider a dynamic list of unsolved integer subproblems, where are reported an index for the node, the level of depth in the tree and a subvector relative to the components already fixed to feasible integer values. At each iteration we pick a subproblem in the list, smaller in terms of dimension (hence a depth-first enumeration strategy) and we solve the relative continuous relaxation, so that the node is pruned or a bunch of subproblems (obtained by fixing the first free variable to feasible integer values) are added to the list. Moreover, at each iteration an integer solution is computed by rounding the primal solution of the relaxation and in case the current best solution is updated. The algorithm goes on until no more subproblems are in the list, so that the current best solution is proven to be optimal. For an outline of the algorithm consider the scheme below.

```
Branch-and-Bound scheme GQIP for (IQP)

Data. $Q \in \mathbb{S}^n$, $L \in \mathbb{R}^n$, $l, u \in \mathbb{Z}^n$.

Pre-processing. Common operations on $Q$.

Initialization. $z_{ub} = \infty$, $p^* = ()$, $\mathcal{L} = \{[s = 1, d = 0, r^{(s,d)} = ()]\}$.

While $\mathcal{L} \neq \emptyset$

        1. Remove problem $[s, d, r^{(s,d)}]$ with $d$ maximal from $\mathcal{L}$.
        2. Compute $L^{(s,d)}$ and $c^{(s,d)}$.
        3. Solve relaxation corresponding to the triple
               $(Q^{(d)}, L^{(s,d)}, c^{(s,d)})$,
           with output $z_{lb}$ and $\bar{x} \in \mathbb{R}^{n-d}$.
        4. Compute integer solution $p = (r^{(s,d)}, \lceil \bar{x}_1 \rfloor, \ldots, \lceil \bar{x}_{n-d} \rfloor)$.
        5. If $q(p) < z_{ub}$, then $p^* = p$ and $z_{ub} = q(p)$.
        6. If $z_{lb} < z_{ub}$ and $n - d > 1$, then set $j = 1$ and do for all $f \in [l_{d+1}, u_{d+1}] \cap \mathbb{Z}$
               (i) add $[s + j, d + 1, (r^{s,d}, f)]$ to $\mathcal{L}$;
               (ii) set $j = j + 1$.

End While

Return. $p^*$, $q^* = q(p^*)$.
```

In the preprocessing phase we carry out all computational tasks which are part of the initialization or are level dependent. In this way, we safe all the operations which are in common between different nodes of the same level. In order, these are the tasks performed in the preprocessing phase:

- **Last non-convex level $\bar{d}$.** Compute maximal d such that $Q^{(d)} \not\succeq 0$.
- **Scaled data for the non-convex level d.** Compute $(\bar{Q}^{(d)}, \bar{L}^{(d)})$ from $(Q^{(d)}, L^{(d)})$ scaling variables in $[-1, 1]$.
- **Ellipsoid for the non-convex level d.** Choose among
  1. $H^{(d)} = I$.
  2. $H^{(d)}$ solution of $\max\limits_{H \in \mathcal{H}_\epsilon} \lambda_1(\bar{Q}^{(d)}, H)$.
  3. $H^{(d)}$ approximated solution of $\max\limits_{H \in \mathcal{H}_\epsilon} p^*(H)$.
  4. $H^{(d)}$ derived for the previous ellipsoid $H^{(d-1)}$.

  Update $\bar{Q}^{(d)} = {H^{(d)}}^{-\frac{1}{2}} \bar{Q}^{(d)} {H^{(d)}}^{-\frac{1}{2}}$ and $\bar{L}^{(d)} = {H^{(d)}}^{-\frac{1}{2}} \bar{L}^{(d)}$.
- **Spectral decomposition for the non-convex level d.** Compute $(\Lambda^{(d)}, V^{(d)})$, eigenvalues-eigenvectors of $\bar{Q}^{(d)}$.
- **Initial solution for the relaxation of the non-convex level d.** Find an optimal solution $y^{(d)}$ of

$$\begin{aligned} \min_{y} \quad & y^T \Lambda^{(d)} y + (\bar{L}^{(d)})^T V^{(d)} y \\ & \|y\|^2 = n - d. \end{aligned}$$

- **Derived matrices for the convex level d.** Compute
  1. $A^{(d)} = {Q^{(d)}}^{-1}$.
  2. B such that $Q^{(d)} = B^T B$ and B^{-1}.
  3. $T^{(d)}$, transformation matrix from original basis of ${B^{-1}}^T$ to the reduced one.
- **Ellipsoid for the convex level d.** Compute
  1. $\lambda_i^{(d)} = \lambda(Q^{(d)}, t_i t_i^T)$, for any column t_i of $T^{(d)}$.
  2. $H^{(d)} = \sum\limits_{i=1}^{n-d} {\lambda_i^{(d)}}^2 t_i t_i^T$.

3. $\bar{\lambda}^{(d)} = \lambda(Q^{(d)}, H^{(d)})$.

- **Linear problems for the convex level d.** Compute

1. $s_i^{(d)} = \max\{x^T t^{(i)} : \; x \in \left[l^{(d)}, u^{(d)}\right]\}$, for any column t_i of $T^{(d)}$.
2. $z_i^{(d)} = \min\{x^T t^{(i)} : \; x \in \left[l^{(d)}, u^{(d)}\right]\}$, for any column t_i of $T^{(d)}$.

Consider now the nodes-processing phase: given a node s at level d, the main task is to solve the relaxation with data $(Q^{(d)}, L^{(s,d)}, c^{(s,d)})$. We need to distinguish between non-convex and convex level, namely if $d \leq \bar{d}$ or not.

Non-convex relaxation

1. Compute $(\bar{L}, \bar{c})$ from $(Q^{(d)}, L^{(s,d)}, c^{(s,d)})$ scaling variables in $[-1, 1]$.
2. Compute $\tilde{L} = V^{(d)^T} H^{(d)^{-\frac{1}{2}}} \bar{L}$.
3. Find (approximated) optimal solution $\bar{\mu}$ of

$$\begin{aligned} \max_{\mu} \quad & f(\mu) = (n-d)\mu - \tfrac{1}{4}\tilde{L}^T(\Lambda^{(d)} - \mu I)^{\dagger}\tilde{L} \\ & \mu \leq \lambda_1^{(d)}. \end{aligned} \tag{5.11}$$

starting from the point

$$\mu = \frac{2y^{(d)^T}\Lambda^{(d)}y^{(d)} + \tilde{L}^T y^{(d)}}{2(n-d)}.$$

Observe that the maximization can be stopped earlier if $f(\mu) \geq z_{ub} - \bar{c}$ for $\mu \leq \lambda_1^{(d)}$.

4. Compute $\bar{x} = H^{(d)^{-\frac{1}{2}}} V^{(d)} \bar{y}$, where

$$\bar{y} = -\frac{(\Lambda^{(i)} - \bar{\mu}I)^{\dagger}\tilde{L}}{2}.$$

5. Scale $\bar{x}$ variables from $[-1, 1]$ to $[l, u]$, namely

$$\bar{x}_j = \frac{\bar{x}_j(u_{j+d} - l_{j+d}) + (l_{j+d} + u_{j+d})}{2}, \quad j = 1, \ldots, n-d.$$

6. Return $z_{lb} = f(\bar{\mu}) + \bar{c}$ and $\bar{x}$.

Convex relaxation

1. Compute $\bar{x} = -Q^{(d)}L^{(s,d)}/2$ and $f(\bar{x}) = c^{(s,d)} - L^{(s,d)}Q^{(d)}L^{(s,d)}/4$.
2. Compute $\bar{\mu}_i = \bar{\mu}(t_i, \bar{x})$, for any column t_i of $T^{(d)}$.

3. Compute $r = \sqrt{\sum_{i=1}^{n-d} \lambda_i^{(d)^2} \bar{\mu}_i^2}$.

4. Return $z_{lb} = f(\bar{x}) + (\bar{\lambda}^{(d)})^2 r^2$ and $\bar{x}$.

Preprocessing reduces drastically the computational cost per iteration. Actually, assuming a fixed maximum number of iteration on solving problem (5.11), running time per node of our algorithm has order of n^2.

It is clear that the require time for solving the relaxation for non-convex nodes are much time-consuming than on the relaxation for convex nodes. Moreover, we conjecture on the quality of the lower bounds, as relatively stronger for convex nodes.
All these considerations provide incentives to find better order of the variables as the first step in the preprocessing phase: the goal is to make appear as soon as possible convex nodes. In an equivalent way, one would like to find the permutation of columns and rows of Q such that the last non-convex level $\bar{d}$ is small as possible.
On the other side, because we are not totally aware of consequences relative to the optimal choice in that sense, it is reasonable to think about some heuristic idea. In particular we exploit the level of diagonal dominance on Q

$$d_i = q_{ii} - \sum_{\substack{j=1 \\ j \neq i}}^{n} |q_{ij}|, \quad i = 1, \ldots, n.$$

If all d_i were strictly positive, then the matrix Q would be definite positive. The idea is choose as the first variable to be fixed the index l with the smallest diagonal dominant factor. Then we update the diagonal dominant factors deleting the l-th column and row and afterwards we choose the second element to be fixed as the least diagonal dominant. In a recursive manner we obtain the order of the variables as they should be considered in the branch-and-bound scheme.

In conclusion of this section we want to underline values and faults of our type of algorithm. Indirectly, we highlight which kind of instances of (IQP) are more suitable to be solved by algorithm `GQIP`.
As we structured our branch-and-bound, in case the current node is not pruned, a bunch of subproblems are generated and added to the list. For example, if the j$-th$ variable has to be fixed then new subproblems are added to the list, one for each integer value in $[l_j, u_j]$, so in number at least $u_j - l_j$. Therefore, roughly speaking, the expected number of nodes generated by our algorithm will be proportional to the width of the box $[l, u]$ of the integer problem (IQP).

As opposed, for a binary branch-and-bound scheme just two subproblems are generated, where at each iteration new bounds on one variable are fixed. In this case the expected number of nodes is less influenced by the width of the box $[l, u]$.
On the other side, with the proposed branch-and-bound the processing time per node is negligible, just paying off an extra cost in the preprocessing phase. With a binary branch-and-bound scheme the preprocessing phase is not such useful, as the data of for a node is generally not in common with other nodes. So we expect more costly operations to solve the relaxation for a node.
Therefore, if the box $[l, u]$ is kept down in terms of width, then the high number of nodes is fully counterbalanced by the rapidity of our algorithm in processing each node. For larger width of the box the speed of our algorithm is not enough to tackle a too huge number of nodes.
For this reason, in the numerical tests, we consider just ternary instances of (IQP), where each variables is constrained to $\{-1, 0, 1\}$. To solve more generally instance we should pass to a binary branch-and-bound and quit with the advantages of preprocessing. Anyway, the relaxations proposed in Section 5.1 continue to hold, just the cost for solving them is higher.

5.3 Numerical results

In this section we describe our computational experience on `GQIP` for solving non-convex integer quadratic instance. All the experiments have been run on the same PC with 4 Gb of RAM and 3.16 Ghz.

Algorithm `GQIP` are implemented both in C++ and Fortran. In particular, the branch-and-bound scheme is written in C++ as well as algorithms for solving the continuous relaxation for convex nodes, while algorithms for solving continuous relaxation for non-convex nodes and algorithms for finding good ellipsoids are both implemented in Fortran.

To build our test problems we defined a random generator for the triple (Q, L, c): for chosen dimension n and level of non-convexity $p \in [0, 1]$, namely the percentage of negative eigenvalues of the matrix, the following operations are performed:

- generation of the eigenvalues $\Lambda \in \mathbb{D}^n$, $\lfloor pn \rfloor$ eigenvalues randomly in $[-1, 0]$ and the remaining randomly in $[0, 1]$;

- generation of $V \in \mathbb{R}^{n \times n}$, randomly in $[-1, 1]$ and orthonormalized;

- computation of $Q = V\Lambda V^T$;
- generation of $L \in \mathbb{R}^n$, randomly in $[-1, 1]$;
- set c=0.

The parameter p allows to control the level of convexity: for the extreme cases, we have $p = 0$ for convex problems, while $p = 1$ for concave problems.
So we defined our test set in this way: we generated instances with dimension n in $\{10, 20, 30, 40, 50\}$ and p in $\{0, 0.1, 0.2, \dots, 1\}$. In particular, for each pair (n, p) we generated 10 instances. Moreover, as we said before, for any instance (Q, L, c), the integer problem is given by restricting each variable to $\{-1, 0, 1\}$ values.

In our experiments we make internal as well as external comparisons in solving ternary instances.
On the one side, we would like to understand which strategy should be used to choose the ellipsoid for the non-convex relaxation. To simplify the number of possibilities, we consider this choice just at the root node, while in the other nodes the ellipsoid is derived from the one at the root node. The first possibility is to take the most natural and easy-to-compute ellipsoid, namely the sphere obtained by setting $H = I$. We refer with GQIP$_1$ to the inherent algorithm. As opposed, in GQIP$_2$ the ellipsoid H is given by the solution of (5.9), namely following the second rule proposed in Subsection 5.1.1.2. In order to solve this problem we use CSDP ([10]) as linear SDP solver. Finally, in GQIP$_3$, following instead the first rule in Subsection 5.1.1.2, the ellipsoid is given by the approximated solution of (5.5), where a projected subgradient is applied for a few number of iterations.
On the other side, we test also the best codes in literature in the main classes of methods for solving problem (IQP), namely COUENNE and Q-MIST. The first code, referred in [6] and downloadable from the web page[1], implements a very general purpose algorithm and is able to solve any mixed-integer nonlinear constrained optimization problems with no assumption of convexity. The code Q-MIST refers to the approach in [18] and works for general mixed-integer quadratic programming. In both cases we compare with more general approaches than GQIP.

Overall performances of the 5 algorithms over the entire test set are reported in Table 5.1. For sake of description, we give only general results respect to dimension of the instances: for each dimension and each algorithm, we report the number of instances out of 110 solved in a time limit of 1 hour and respect to the solved instances we give

[1] https://projects.coin-or.org/Couenne

the average time, the minimum time, the maximum time and also the average number of nodes. For GQIP we give also the average time spent in the preprocessing.

n	alg	avg pre-time	min time	max time	avg time	avg node	solved
10	COUENNE		0.0	0.4	0.1	18.0	110
	Q-MIST		0.0	1.0	0.0	9.1	110
	$\texttt{GQIP}_1$	0.0	0.0	0.0	0.0	34.1	110
	$\texttt{GQIP}_2$	0.0	0.0	0.0	0.0	37.5	110
	$\texttt{GQIP}_3$	0.0	0.0	0.1	0.0	28.7	110
20	COUENNE		0.8	51.4	13.0	3822.0	110
	Q-MIST		0.0	1.0	0.2	53.5	110
	$\texttt{GQIP}_1$	0.0	0.0	0.0	0.0	1105.0	110
	$\texttt{GQIP}_2$	0.0	0.0	0.0	0.0	1143.5	110
	$\texttt{GQIP}_3$	0.1	0.0	0.2	0.1	680.9	110
30	COUENNE		26.2	3567.3	1476.7	181127.6	78
	Q-MIST		0.0	10.0	2.0	199.7	110
	$\texttt{GQIP}_1$	0.0	0.0	1.1	0.1	27933.6	110
	$\texttt{GQIP}_2$	0.0	0.0	1.4	0.2	28184.1	110
	$\texttt{GQIP}_3$	0.4	0.0	0.8	0.4	7378.7	110
40	COUENNE		876.3	3148.3	2012.3	99500.0	2
	Q-MIST		1.0	106.0	16.1	831.6	110
	$\texttt{GQIP}_1$	0.1	0.1	54.3	5.8	952880.4	110
	$\texttt{GQIP}_2$	0.1	0.1	48.0	5.2	847748.3	110
	$\texttt{GQIP}_3$	0.9	0.1	13.4	1.8	123163.7	110
50	COUENNE		***	***	***	***	0
	Q-MIST		3.0	1593.0	186.0	5463.7	110
	$\texttt{GQIP}_1$	0.2	0.4	1449.9	190.0	27182336.9	109
	$\texttt{GQIP}_2$	0.2	0.4	878.8	165.2	24577033.7	108
	$\texttt{GQIP}_3$	1.9	0.4	361.2	34.5	4211735.9	110

Table 5.1: Results for ternary instances

First of all, we observe that COUENNE can solve all the instance just for small dimensions. Actually, since $n = 30$, it can solve more or less just two-thirds of the instances, only two for $n = 40$ and nothing for $n = 50$. As opposed, the other algorithms are all able to solve the entire set of instances, with just one failure for $\texttt{GQIP}_1$ and two for $\texttt{GQIP}_2$ for the largest dimension. So big difference in performance is due to the fact that COUENNE is very general purpose algorithm, so that it can not really exploit the particular structure of (IQP).

Looking at our algorithms results, performances improve proportionally to the effort done to strength the non-convex relaxation at the root node. Actually, with a bit more time spent in the preprocessing to find better ellipsoids, the average number of nodes decreases consistently comparing GQIP$_1$ and GQIP$_2$ with GQIP$_3$. This reduction of nodes leads to huge savings in terms of time. This big reduction of time and nodes can be explained by the persistent and consistent improvement of the lower bound provided by the non-convex relaxation at the root node (which may be also profitable for the other nodes), comparing the one given in GQIP$_1$ and the one in GQIP$_3$. In Figure 5.2 we consider the histogram corresponding to the percentage variation of the lower bound provided by these two algorithms.

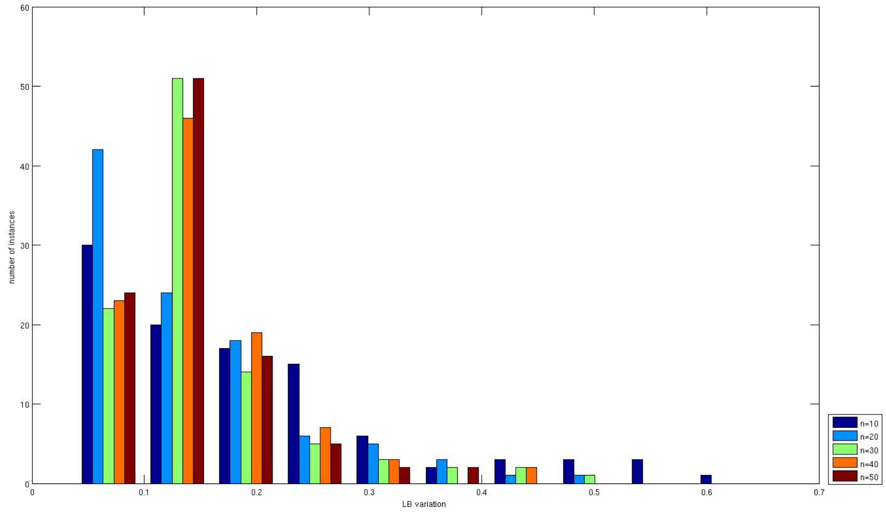

Figure 5.2: Lower bound improvement - Histogram of the lower bound variation from GQIP$_1$ to GQIP$_3$

We do not report any detailed comparison with GQIP$_2$ because the conclusions would be roughly the same. In particular, the SDP rule used to choose the ellipsoid in GQIP$_2$ does not make any difference respect to sphere used GQIP$_1$. This fact can be explained observing that in the SDP rule only the quadratic term is considered, hence neglecting the linear term. This does not guarantee any direct improvement of the lower bound. Overall, we conclude that defining strong relaxations, for a small extra cost in the preprocessing, provides the right counterbalance for an easy-to-solve relaxation in terms of number of generated nodes.

As matter of fact, just GQIP$_3$ really outperforms the other algorithm Q-MIST. On the one side, GQIP$_3$ produces much more subproblems than Q-MIST, which makes think that our non-convex relaxation is provably weaker than the SDP relaxation used for Q-MIST. On the other side, because for our algorithm the time for solving a single node is negligible, it turns out that GQIP$_3$ results much faster that Q-MIST, on average 5-6 times more. Moreover, we also compare what happens to the running times respect to level of non-convexity: in Figure 5.3 we report average running times (in logarithmic scale) respect to the percentage p of negative eigenvalues.

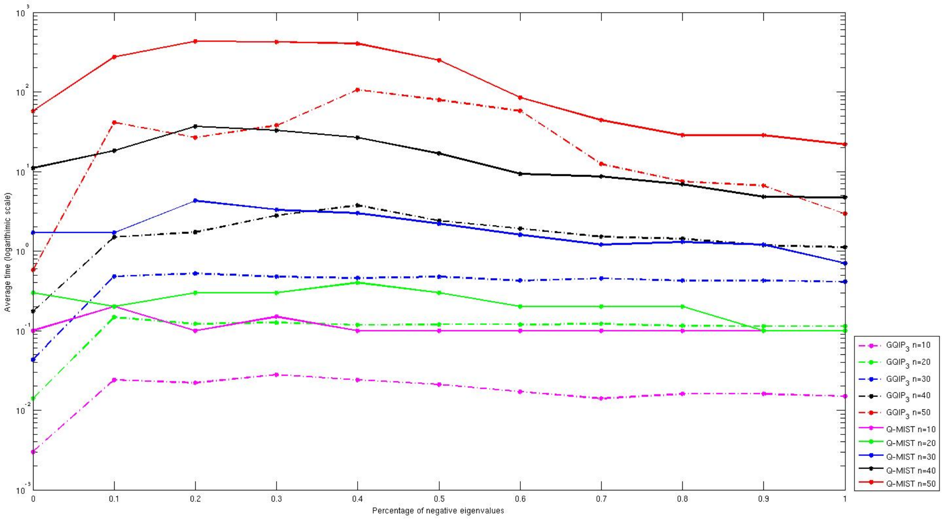

Figure 5.3: Average times - detailed comparison between GQIP$_3$ and Q-MIST

Dashed lines correspond to GQIP$_3$ and full line to Q-MIST, with different colors relative to different dimensions. Also in this analysis, it comes out that GQIP$_3$ performs better than Q-MIST for any level of convexity, which seems to less influence the performance of our algorithm.

In conclusion, the proposed exact algorithm for (IQP) is extremely suitable to solve ternary instances. For this case no other approach can compare with GQIP when the ellipsoid relaxation is chosen as strong as possible.

Going to more general case of (IQP), where variables can take a wider range of integer values, our algorithm fails quite often, while Q-MIST seems to be less affected by the width of integer feasible values. This is the reason we have not given any results about more general instances, such as $\{-10, \ldots, 10\}$. Actually, for large instances we get almost all failures, while `Q-MIST` can solve in a reasonable time much more than 50% of the instances.

As we discuss in the end of Section 5.2, in order to solve more general instances we should switch to another kind of branch-and-bound, getting rid of advantages of preprocessing.

Appendix A

Subgradient method over the Simplex

Consider the problem

$$\begin{aligned} \min_{x} \quad & f(x) \\ & x \in S = \{x \in \mathbb{R}^n : \ u^T x = n, \ x \geq 0\}, \end{aligned} \tag{A.1}$$

where $f(x)$ is convex and not differentiable over S. We assume that for any $x \in S$ the function value $f(x)$ and a subgradient $g(x)$ can be computed. In this situation Subgradient methods can be used to solve approximately (A.1).

Subgradient Methods were developed for unconstrained non-smooth optimization (see e.g. [57]) and later extended to constrained convex case with the use of projection operators (see e.g. [2]): the idea is to move in the opposite direction of the subgradient for a certain step-length and then project it back on the feasible set.
Theoretical convergence of Subgradient Methods has to be intended as convergence to the optimal value. Convergence proofs depend basically on the distance from an optimal solution, which can be controlled by choosing carefully the step-length.
We recall that the exact projection over the simplex can be computed in $\mathcal{O}(nlogn)$, as it is explained in [20]. We denote with $\mathcal{P}_S(\cdot)$ the corresponding operator.
For sake of completeness, we outline a possible subgradient method for problem (A.1).

```
Projected Subgradient algorithm for (A.1)

Initialization. $x^0 = u$.

For $k = 0, 1, \ldots$
    1. Compute function $f(x^k)$ and subgradient $s(x^k)$.
    2. If $s(x^k) = \beta u$ for some $\beta \in \mathbb{R}$ then stop.
    3. Choose carefully $\alpha^k > 0$.
    4. Set $z^{k+1} = x^k - \alpha^k s^k$.
    5. compute $x^{k+1} = \mathcal{P}_S(z^{k+1})$.
End For
```

In conclusion, we give some more explanations about the algorithm. The condition in step 2 is sufficient for optimality in problem (A.1), so that if it holds we can return directly with the optimal solution. Generally, we do not expect this condition to be satisfied, so we run the algorithm for a finite number of iterations, afterwards we return with the best point computed in terms of $f(x^k)$.

Practically speaking, Subgradient methods are quite good at the beginning, but they get slower and slower within execution. This fact is because of the step-length must be chosen smaller and smaller, in order to guarantee convergence to an optimal solution. Therefore, this algorithm is useful if and only if an exact optimal solution of (A.1) is not really needed, but it is sufficient an approximated one.

Appendix B

Backround material

For reference of the topics dealt in this appendix we recommend [37].

B.1 Eigenvalues and Semidefinite matrices

First, we recall definition and properties of eigenvalues of a symmetric matrix $A \in \mathbb{S}^n$.

Definition B.1.1 *A scalar $\lambda \in \mathbb{R}$ is called an eigenvalue of A if the following condition holds*

$$Ay = \lambda y,$$

for a non-null $y \in \mathbb{R}^n$, which is called eigenvector of A associated to λ.

Let $m \leq n$ the number of distinct eigenvalues of A. For each distinct eigenvalue λ_i, it is associated the eigenspace

$$\mathcal{Y}_i = \ker\left(A - \lambda_i I\right),$$

the subspace of all eigenvectors corresponding to that eigenvalue. Symmetry makes eigenvector associated to different eigenvalue being automatically orthogonal,

$$\lambda_i y_j^T y_i = y_i^T A y_j = \lambda_j y_j^T y_i.$$

Because A is symmetric and real, the union of the eigenspaces represents the entire $\mathbb{R}^n$ as

$$\dim(\mathcal{Y}_1) + \cdots + \dim(\mathcal{Y}_m) = n.$$

Moreover, A is diagonalizable by an orthonormalized basis of eigenvectors to the matrix of eigenvalues, repeated in their own multiplicities. More formally we have the following theorem.

Theorem B.1.2 *If A is symmetric and real then*

(i) $R^n = \mathcal{Y}_1 \oplus \cdots \oplus \mathcal{Y}_m$

(ii) $A = Y\Lambda Y^T$, *with* $Y \in \mathbb{R}^{n\times n}$ *orthogonal and* $\Lambda \in \mathbb{D}^n$.

Second, we consider definite symmetric matrices and their own properties.

Definition B.1.3

(i) X *is semidefinite positive* ($X \succeq 0$) *if* $y^T Xy \geq 0$ *for all* $y \in \mathbb{R}^n$

(ii) X *is definite positive* ($X \succ 0$) *if* $y^T Xy > 0$ *for all* $y \in \mathbb{R}^n/\{0\}$

Theorem B.1.4 *Given* $X \in \mathbb{S}^n$, *the following statements are equivalent:*

(i) $X \succeq 0$;

(ii) *the eigenvalues of* X *are nonnegative;*

(iii) *all principal minors of* X *are nonnegative;*

(iv) *there exists* $V \in \mathbb{R}^{n\times n}$ *such that* $X = VV^T$ *and* $\text{rank}(X) = \text{rank}(V)$.

Theorem B.1.5 *Given* $X \in \mathbb{S}^n$, *the following statements are equivalent:*

(i) $X \succ 0$;

(ii) *the eigenvalues of* X *are positive;*

(iii) *the determinant of any principal sub-matrix of* X *is positive;*

(iv) *there exists full rank* $V \in \mathbb{R}^{n\times n}$ *such that* $X = VV^T$.

Using this characterization is possible to show that

$$\{X \in \mathbb{S}^n : \ X \succeq 0\}$$

is a closed convex cone, with interior defined as

$$\{X \in \mathbb{S}^n : \ X \succ 0\}.$$

In the following, we recall some useful properties for semidefinite matrices.

Proposition B.1.6 *Given* $X \in \mathbb{S}^n$, *the following properties hold.*

1. *$X \succeq 0$ if and only if $Y \bullet X \geq 0$ for any $Y \succeq 0$.*

2. *If $X, Y \succeq 0$ then $Y \bullet X \geq 0$. Moreover, $X \bullet Y = 0$ if and only if $XY = 0$.*

3. *If $X \succ 0$ the set $\{Y \succeq 0 : \ X \bullet Y \leq \beta\}$ is compact for any $\beta > 0$.*

4. *If $X \succeq 0$ with rank r, then there exist $Q \in \mathbb{R}^{n \times r}$ and $E \in \mathbb{S}^r$, with $E \succ 0$, such that $X = QEQ^T$.*

5. *If $X \succeq 0$ then it possible to define $X^{\frac{1}{2}}$ such that $X^{\frac{1}{2}}X^{\frac{1}{2}} = X$.*

6. *If $\lambda_1, \ldots, \lambda_n$ are the eigenvalues of X, then*

$$I \bullet X = \sum_{i=1}^{n} \lambda_i,$$

$$\det(X) = \prod_{i=1}^{n} \lambda_i,$$

$$\operatorname{rank}(X) = \|\lambda\|_0.$$

B.2 Kronecker product

The kronecker product is a useful operation between matrices and facilitates the handling of matrix equations and derivatives.

Definition B.2.1 *Given $A \in \mathbb{R}^{n \times m}$ and $B \in \mathbb{R}^{p \times q}$, then the Kronecker product is defined as the matrix*

$$\begin{pmatrix} a_{11}B & \cdots & a_{1m}B \\ \vdots & \ddots & \vdots \\ a_{n1}B & \cdots & a_{nm}B \end{pmatrix} \in \mathbb{R}^{np \times mq}.$$

Quite often, the kronecker product is combined with the vec operator.

Definition B.2.2 *Given $A \in \mathbb{R}^{n \times m}$ then the* vec *operator is defined as the vector*

$$\begin{pmatrix} A_{\cdot 1} \\ \vdots \\ A_{\cdot m} \end{pmatrix} \in \mathbb{R}^{nm},$$

obtained by stacking columns of A.

We report some important properties of the Kronecker product.

Proposition B.2.3 *Given matrices A, B, C, D, the following properties hold.*

1. $(A \otimes B)^T = (A^T \otimes B^T)$.
2. $A \otimes B \otimes C = (A \otimes B) \otimes C = A \otimes (C \otimes A)$.
3. $(A + B) \otimes (C + D) = A \otimes C + A \otimes D + B \otimes C + B \otimes D$.
4. $(A \otimes B)(C \otimes D) = (AC \otimes BD)$.
5. *If* $A, B \succeq 0$ *then* $(A \otimes B) \succeq 0$.
6. $\mathrm{vec}(ABC) = (C^T \otimes A)\mathrm{vec}(B)$.
7. $\mathrm{vec}(ABC) = (C^T \otimes A)\mathrm{vec}(B)$.
8. $\mathrm{vec}(AB + BC) = (I \otimes A + C^T \otimes A)\mathrm{vec}(B)$.
9. $A_{\underset{i}{:}} = (e_i \otimes I)^T \mathrm{vec}(A)$.

References

[1] P.A. Absil, F. Bach, M. Journée, and R. Sepulchre. Low-rank optimization for semidefinite convex problems. *SIAM Journal on Optimization*, 20(5):2327–2351, 2010. 16, 22, 37, 48

[2] Ya.I. Alber, A.N. Iusem, and M.V. Solodov. On the projected subgradient method for nonsmooth convex optimization in a Hilbert space. *Mathematical Programming*, 81(1):23–35. 121

[3] R. Andreani, E.G. Birgin, J.M. Martínez, and M.L. Schuverdt. On augmented lagrangian methods with general lower-level constraints. *SIAM Journal on Optimization*, 18:1286–1309, 2007. 88

[4] R. Andreani, E.G. Birgin, J.M. Martínez, and M.L. Schuverdt. Second-order negative-curvature methods for box-constrained and general constrained optimization. *Computational Optimization and Applications*, 45(2):209–236, 2010. 24, 70, 88

[5] F. Barahona, M. Grötschel, M. Jünger, and G. Reinelt. An application of combinatorial optimization to statistical physics and circuit layout design. *Operations Research*, 36:493–513, 1988. 29

[6] P. Belotti. Couenne: a user's manual. Technical report, Lehigh University. 95, 115

[7] S.J. Benson, Y. Ye, and X. Zhang. Solving large-scale sparse semidefinite programs for combinatorial optimization. *SIAM Journal on Optimization*, 10:443–461, 1998. 47

[8] D. Bertsekas. *Nonlinear Programming.* Athena Scientific, 2nd edition, 1999. 24, 25

[9] P. Bonami, L.T. Biegler, A.R. Conn, and A. W´achter. An algorithmic framework for convex mixed integer nonlinear programs. *Discrete Optimization*, 5:186–204, 2008. 95

[10] B. Borchers. CSDP, a C library for semidefinite programming. *Optimization Methods and Software*, 11-12(1-4):613–623, 1999. 88, 115

[11] E. Boros, P.L. Hammer, and G. Tavares. Local search heuristics for Quadratic Unconstrained Binary Optimization (QUBO). *Journal of Heuristics*, 13(2):99–132, 2007. 31

[12] S.P. Boyd and L. Vandenberghe. *Convex Optimization*. Cambridge University Press, 2nd edition, 2004. 5

[13] C. Buchheim, A. Caprara, and A. Lodi. An Effective Branch-and-Bound Algorithm for Convex Quadratic Integer Programming. *Integer Programming and Combinatorial Optimization*, 2011. to appear. 95, 107, 108

[14] S. Burer and C. Choi. Computational enhancements in low-rank semidefinite programming. *Optimization Methods and Software*, 21(3):493–512, 2006. 16, 24, 25, 36, 48, 89

[15] S. Burer and R.D.C. Monteiro. A projected gradient algorithm for solving the maxcut SDP relaxation. *Optimization Methods and Software*, 15:175–200, 2001. 16

[16] S. Burer and R.D.C. Monteiro. A nonlinear programming algorithm for solving semidefinite programs via low-rank factorization. *Mathematical Programming Ser. B*, 95(2):329–357, 2003. 16, 17, 19, 21, 24, 25, 35, 36, 38, 48, 89

[17] S. Burer and R.D.C. Monteiro. Local Minima and Convergence in Low-Rank Semidefinite Programming. *Mathematical Programming Ser. A*, 103(3):427–444, 2005. 16, 24, 25, 36, 48, 89

[18] C. Buchheim and A. Wiegele. Semidefinite Relaxations for Non-Convex Quadratic Mixed-Integer Programming. *Optimization on-line*, 2011. 95, 115

[19] E.D. Dolan and J.J. Morè. Benchmarking optimization software with performance profile. *Mathematical Programming, Ser. A*, 91:201–213, 2002. 48

[20] J. Duchi, S. Shalev-Shwartz, Y. Singer, and T. Chandra. Efficient projections onto the 1 -ball for learning. *Proc. Int. Conf. Mach. Learn. ICML '08*, 25(307):272–279, 2008. 121

[21] K. Fan. On a theorem of Weyl concerning eigenvalues of a linear trasformations. *Mathematics*, 36:31–35, 1950. 59

[22] P.A. Fillmore and J.P. Williams. Some convexity theorems for matrices. *Glasgow Mathematical Journal*, 1971. 59

[23] I. Fischer, G. Gruber, F. Rendl, and R. Sotirov. Computational experience with a bundle approach for semidefinite cutting plane relaxations of Max-Cut and equipartition. *Mathematical Programming*, 105:451–469, 2006. 45

[24] M.X. Goemans and D.P. Williamson. Improved approximation algorithms for MAX-CUT and satisfiability problems using semidefinite programming. *Journal of the ACM*, 42(6):1115–1145, 1995. 31, 32, 34

[25] L. Grippo and M. Sciandrone. Nonmonotone globalization techniques for the Barzilai- Borwein gradient method. *Computational Optimization and Applications*, 23(2):143–169, 2002. 47, 91

[26] L. Grippo, L. Palagi, M. Piacentini, V. Piccialli, and G. Rinaldi. SpeeDP: A new algorithm to compute the SDP relaxations of Max-Cut for very large graphs. DII-UTOVRM Technical Report 13.10, University of Rome Tor Vergata. 51

[27] L. Grippo, L. Palagi, and V. Piccialli. Necessary and sufficient global optimality conditions for NLP reformulations of linear SDP problems. *Journal of Global Optimization*, 44(3):339–348, 2008. 16, 19, 21, 35

[28] L. Grippo, L. Palagi, M. Piacentini, and V. Piccialli. An unconstrained approach for solving low rank SDP relaxations of $\{-1, 1\}$ quadratic problems. Technical Report 13, Dip. di Informatica e sistemistica A. Ruberti, Sapienza Università di Roma, 2009. 38

[29] L. Grippo, L. Palagi, and V. Piccialli. An unconstrained minimization method for solving low rank SDP relaxations of the maxcut problem. *Mathematical Programming, Ser. A*, 126(1):119–146, 2009. 35, 37, 47, 48

[30] M. Grötschel, L. Lovàsz, and A. Schrijver. *Geometric Algorithms and Combinatorial Optimization.* Springer Verlag. 6

[31] C. Helmberg and F. Rendl. Solving quadratic $(0, 1)$-problems by semidefinite programs and cutting planes. *Mathematical Programming, Ser. A*, 82(3):291–315, 1998. 32

[32] C. Helmberg and F. Rendl. A Spectral Bundle Method for Semidefinite Programming. *SIAM Journal on Optimization*, 10:673–696, 2000. 15, 34, 47

[33] C. Helmberg, B. Mohar, S. Poljak, and F. Rendl. A spectral approach to bandwidth and separator problems in graphs. *Linear and Multilinear Algebra*, 39:73–90, 1995. 72

[34] V. Hernández, J.E. Román, A. Thomás, and V. Vidal. A survey of software for sparse eigenvalue problems. Technical Report 6, Universidad Politecnica de Valencia. 72

[35] R.D. Hill and S.R. Waters. On the cone of semidefinite matrices. *Linear Algebra and its Applications*, 90:81–88, 1987. 9

[36] S. Homer and M. Peinado. Design and Performance of Parallel and Distributed Approximation Algorithms for Maxcut. *Journal of Parallel and Distributed Computing*, 46(1):48–61, 1997. 16, 17, 38

[37] R.A. Horn and C.R. Johnson. *Matrix Analysis.* Cambridge University Press, 1985. 123

[38] ILOG. Inc. ilog cplex 12.1. 95
http://www.ilog.com/products/cplex, 2009.

[39] E. De Klerk. *Aspects of Semidefinite Programming: Interior Point Algorithms and Selected Applications.* Kluwer Academic, 2002. 7

[40] A.S. Lewis and M.L. Overton. Eigenvalue optimization. *Acta Numerica*, pages 149–190, 1996. 57

[41] F. Liers, M. Jünger, G. Reinelt, and G. Rinaldi. *Computing exact ground states of hard Ising spin glass problems by branch-and-cut*, pages 47–69. Wiley-VCH Verlag, 2004. 31

[42] S. Lucidi. New results on a continuously differentiable exact penalty function. Technical Report 277, Istituto di Analisi di Sistemi e Informatica del CNR, Roma, Italy. 26

[43] S. Lucidi. New results on a continuously differentiable exact penalty function. *SIAM Journal on Optimization*, 2(4):558–574, 1989. 26

[44] H.D. Mittelmann. An independent benchmarking of SDP and SOCP solvers. *Mathematical Programming, Ser. B*, 95(2):407–430, 2003. 49

[45] J.J. Moré and D.C. Sorensen. Computing a Trust Region Step. *SIAM Journal on Scientific and Statistical Computing*, 4(3):553–572, 1983. 98, 101, 102

[46] Y. Nesterov. Quality of semidefinite relaxation for nonconvex quadratic optimization. Discussion Papers 1997019, 1997. 32

[47] M.R. Oskoorouchi and J.L. Goffin. A matrix generation approach for eigenvalue optimization. *Mathematical Programming Ser. A*, 109:155–179, 2007. 58

[48] M.L. Overton. Large-Scale Optimization of Eigenvalues. *SIAM Journal on Optimization*, 2:88–120, 1992. 58, 61

[49] M.L. Overton and R.S. Womersley. Optimality conditions and duality theory for minimizing sums of the largest eigenvalues of symmetric matrices. *Mathematical Programming*, 62:321–357, 1991. 58, 59, 60

[50] G. Pataki. On the rank of extreme matrices in semidefinite programming and the multiplicity of optimal eigenvalues. *Mathematics of Operations Research*, 23(2): 339–358, 1998. 11, 12, 58

[51] S. Poljak and Z. Tuza. Maximum cuts and largest bipartite subgraphs. *Discrete Mathematics and Theoretical Computer Science, Ser. DIMACS*, 20:181–244, 1995. 29

[52] F. Rendl and H. Wolkowicz. A semidefinite framework to trust region subproblems with applications to large scale minimization. *Mathematical Programming*, 77(1): 273–299, 1977. 98

[53] F. Rendl, G. Rinaldi, and A. Wiegele. Solving Max-Cut to Optimality by Intersecting Semidefinite and Polyhedral Relaxations. *Mathematical Programming*, 121: 307–335, 2010. 32

[54] G. Rinaldi. Rudy-graph generator. 48, 51
http://www-user.tu-chemnitz.de/∼helmberg/sdp_software.html, 1998.

[55] N.V. Sahinidis and M. Tawarmalani. *BARON 9.0.4: Global Optimization of Mixed-Integer Nonlinear Programs, User's Manual*, 2010. 95

[56] R. Saigal, L. Vandenberghe, and H. Wolkowicz. *Handbook of Semidefinite Programming: theory, algorithms, and applications.* Kluwer, 2000. 1

[57] N.Z. Shor, K.C. Kiwiel, and A. Ruszcayǹski. *Minimization Methods for Non-differentiable Functions.* Springer-Verlag, 1985. 121

[58] R.J. Stern and H. Wolkowicz. Indefinite Trust Region Subproblems and Non-symmetric Eigenvalue Perturbations. *SIAM Journal on Optimization*, 5:286–313, 1995. 98

[59] M.J. Todd. Semidefinite Optimization. *Acta Numerica*, 10:515–560, 2001. 4

[60] S.A. Vavasis. *Nonlinear Optimization.* Oxford University Press, 1991. 98

[61] M. Yamashita, K. Fujisawa, M. Fukuda, K. Kobayashi, K. Nakata, and K. Goto. A high-performance software package for semidefinite programs: SDPA 7. Research Report B-460, Dept. of Mathematical and Computing Science, Tokyo Institute of Technology, Tokyo, Japan, 2010. 88

[62] M. Yamashita, K. Fujisawa, M. Fukuda, K. Kobayashi, K. Nakata, and M. Nakata. Latest developments in the sdpa family for solving large-scale sdps. *Handbook on Semidefinite, Conic and Polynomial Optimization*, 166(3):687–713, 2012. 88