UNIVERSITY OF CALIFORNIA, SAN DIEGO

Optimization over Convex Matrix Inequalities

A dissertation submitted in partial satisfaction of the
requirements for the degree Doctor of Philosophy
in Engineering Sciences (Aerospace Engineering)

by

Juan Francisco Camino

Committee in charge:

Professor Robert E. Skelton, Chairperson
Professor J. William Helton, Co-Chair
Professor Robert Bitmead
Professor Philip E. Gill
Professor William M. McEneaney

2003

The dissertation of Juan F. Camino is approved, and it is acceptable in quality and form for publication on microfilm:

Co-Chair

Chair

University of California, San Diego

2003

To my parents:

Cleonice and Leôncio

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

PREFACE

Since the earlier 90's, Matrix Inequalities (MIs) have become very important in engineering, particularly in control theory. It is unquestionable that the approaches proposed in the field of optimization and control theory based on matrix inequalities and semidefinite programming have become very important and promising. Indeed, matrix inequalities provide a nice set up for many engineering and related problems, and if the MIs are convertible to LMIs, the optimization problem is well behaved and interior point methods provide efficient algorithms.

This thesis provides tools which can solve a large class of optimization problems over matrix inequalities. This includes, for instance, systems and control problems, or any other type of engineering problem that can be posed as a matrix inequality. To use these tools, no knowledge of Linear Matrix Inequalities (LMIs) is required. Furthermore, these tools preserve the advantages of the LMI framework.

To understand the motivation for this task, one must expose some of the advantages and disadvantages of the LMI framework. The wide acceptance of LMIs stems from the following facts: 1) if a control problem is posed as an LMI, then any solution is a global optimum; 2) efficient LMI solvers are readily available; 3) once a control problem is posed as an LMI, any other constraints in the form of LMIs can be added to the problem. On the other hand, the LMI framework has the following disadvantages: 1) there is no systematic way to produce LMIs for general classes of problems; 2) there is no way to even know if it is possible to reduce a system problem to an LMI, without actually doing it; 3) the user must posses the knowledge of manipulating LMIs; 4) transformations via Schur complements can lead to a large LMI representation.

If one has the ability to check whether or not an MI is convex and convertible to an LMI, then the optimization problem can be solved by the many available LMI solvers; however, if one does not have the ability to deal with LMIs, it is not clear what one should do. An alternative is to restate the entire optimization problem in the form used by some particular numerical nonlinear optimization solver. In this case, since optimization over matrix functions are inherently not smooth, there is no guarantee of even a local minimum. Furthermore, the tedious process of reformulating a matrix optimization problem usually requires a high level of skill in algebra.

The main objective of this thesis is to provide a numerical solver for optimization problems over matrix inequalities that possesses similar advantages as the LMI framework,

but without its disadvantages. Our method has two components: 1) a numerical algorithm that solves a large class of matrix optimization problems; 2) a symbolic "Convexity Checker" which automatically provides a region on which the solution from 1) is a global optimum.

The convexity checker presented in Chapter 3 is one of the main contributions of this thesis. This symbolic convexity checker algorithm takes as input a matrix function $F(X)$ and gives as output a family of inequalities which determine a domain $\mathcal{G}(X)$ on which $F(X)$ is "matrix convex." In this way, if $\mathcal{G}(X)$ is convex, the checker gives a certificate that the solution obtained from the numerical solver is indeed a global minimum inside the region $\mathcal{G}(X)$.

The numerical optimization solver NCSDP, presented in Chapter 4, is another contribution of this thesis. This tool can be used to solve optimization problems over matrix inequalities, and does not require any knowledge of LMIs, or how to manipulate MIs to be expressed as LMIs. Therefore, there is no need to determine Schur complements in order to express the matrix constraints as LMIs. Moreover, since transformations via Schur complements can lead to a large LMI representation, the NCSDP solver can reduce the optimization time significantly when the dimensions of the matrices involved are large.

Putting together the convexity checker and the numerical optimization solver for matrix functions, one has available a very powerful tool to solve many engineering problems that can be posed using matrix inequalities. This approach also guarantees a region in which the solution is optimal. Suppose one has multiple matrix inequalities, then by utilizing the convexity checker it is possible to find a region on which all those inequalities happen to be convex. Once this region is determined, one can solve an optimization problem that takes into account this region of convexity. This tool addressed two important questions:

1. How one can determine if an MI is convex or not. (This is the convexity checker.)

2. If an MI is convex, how one can numerically solve an optimization problem without having to convert the MI into an LMI. (This is the NCSDP solver.)

It should be noticed, however, that many other types of engineering problems are intrinsically nonconvex, which makes the above tools no longer useful. Nonetheless, for a specific nonconvex problem frequently encountered in control designs, the simultaneous design of the plant parameters and the control law, this thesis provides a convexifying theory which allow us to approximate the solution of this nonconvex integrated structure and control design by iterating on a sequence of convex subproblems. The convex subproblem

is obtained by adding a potential function to the nonconvex constraints. In practice, this added potential function disappears at stationary solutions of the nonconvex problem. The convexifying theory is another contribution of this thesis, which is presented in Chapter 5.

It is important to understand whether or not the variables in the matrix inequalities should be treated as commutative or noncommutative (treating a matrix as a single variable). Suppose one has the following noncommutative matrix function:

$$A^T X A + X B^T B X + Q \tag{0.1}$$

This function has the same form regardless of the dimension of the defining matrices A, B, Q, and X. It is also possible to write a commutative version of the above matrix function, as a combination of known matrices L_0, L_1, ..., L_m, L_{11}, L_{12}, ..., L_{mm} in unknown real numbers $x_1, \ldots, x_m$ (which are the entries of X):

$$L_0 + \sum_{j=1}^{m} L_j x_j + \sum_{i=1}^{m} \sum_{j=1}^{m} L_{ij} x_i x_j. \tag{0.2}$$

The formulas for the L's depend on the dimension of the underlying matrices A, B, Q, and X. As the dimensions of the matrices increases, the more complicated the formulas for the L's becomes.

From the above two equivalent representations (0.1) and (0.2), it is unquestionable that the noncommutative approach for dealing with MIs provides a more elegant mathematical framework. In addition we shall show it is more powerful. First of all it allows one to compute efficiently directional derivatives. For instance, the second derivative, the Hessian, of (0.1) is easily computed as being $B^T B$. This is a clean expression that does not depend on the dimension of the matrices involved. On the other hand, the formula for the Hessian of (0.2) depends on the L's and consequently, on the dimensions of the matrices involved and it is messy.

We now note two interesting feature which arise in the noncommutative approach.

1. Noncommutativity is likely to be the only practical necessary and sufficient approach available for checking convexity of matrix functions, given that the ability of checking convexity of a matrix function is associated with the ability of determining positivity of its Hessian matrix at each point. This is a huge calculation, so it must be done symbolically. Even for problems with moderate number of (commuting) variables, the size would be overwhelming without using aggregated noncommutative structure.

2. The algebraic linear system of equations, that appears in the subproblem of our optimization solver, has basically the following special "Sylvester" structure:

$$\sum_{i}^{N} \mathcal{A}_i \delta_X \mathcal{B}_i + \sum_{i}^{N} \mathcal{B}_i^T \delta_X \mathcal{A}_i^T = \mathbb{Q}$$

where the $\mathcal{A}$'s and $\mathcal{B}$'s are obtained by collecting the terms on both the left and on the right side of the update direction δ_X that appears inside the "Hessian map." This Sylvester form is not unique and our research has shown that making N small saves considerable computer time; we give algorithms for doing this. This appealing Sylvester structure bears furthers study.

ACKNOWLEDGMENTS

I would like to thank my advisor's Bob Skelton and Bill Helton for their thoughtful suggestions and the time they have spent making this thesis possible. Their insistence on perfection was relentless and often painful. But his commitment to quality will serve as a guide for the rest of my life. I am grateful for their deep involvement in my work.

I am deeply thankful to "Coordenação de Aperfeiçoamento de Pessoal de Nível Superior - CAPES" for a full scholarship, which was my main source of financial support. I am also grateful to Professors Helton and Skelton for partial financial support; in particular to Professor Helton's grants from the NSF (Award Number–DMS–0100576), DARPA, and the Ford Motor Co., and to Professor Skelton's grants from AFOSR F49620-01-1-0141.

I am also grateful to my committee for their careful reading of my thesis. I would like to especially thank my colleague Maurício de Oliveira for his co-work and his many suggestions to my thesis.

Finally, I am glad to mention the people in my personal life that made the endless hours of work on this dissertation tolerable (for sharing this journey with me). These are my parents Leôncio and Cleonice, my brother Tiago, my sisters Ana Luisa and Carla Enoska, and my aunt Creuza. I would like also to thank my friends Maria da Conceição, Felipe and Kirsten Loureiro, and Alex Garcia for their ever present support.

VITA

July 30, 1970	Born, Recife, Pernambuco, Brazil
2003	Doctor of Philosophy University of California, San Diego
1997	MSc, Campinas State University - UNICAMP Campinas, Brazil
1994	BSc, Paraíba Federal University - UFPB Paraíba, Brazil

PUBLICATIONS

Journal papers

J. F. Camino, J. W. Helton, Robert E. Skelton, and Jieping Ye. Matrix inequalities: A symbolic procedure to determine convexity automatically. *Integral Equations and Operator Theory*, 46(4):399–454, July 2003.

J. F. Camino, M. C. de Oliveira, and R. E. Skelton. "Convexifying" linear matrix inequality methods for integrating structure and control design. *ASCE Journal of Structural Engineering: Special Issue on Structural Control*, 129(7):1–11, July 2003.

R. H. C. Takahashi, J. F. Camino, D. E. Zampieri, and P. L. D. Peres. Multiobjective Weighting Selection for Optimization-Based Control Design. *Journal of Dynamic Systems, Measurement, and Control*, 122(3):567–570, September 2000.

Conference papers

J. F. Camino, M. C. de Oliveira, and R. E. Skelton. Plant and control design using convexifying LMI methods. In *Proceedings of the XV IFAC World Congress on Automatic Control*, Barcelona, Spain, July 2002. T-We-M18 (cdrom).

J. F. Camino, M. C. de Oliveira, and R. E. Skelton. Convexifying LMI methods for integrating structure and control design. In Fabio Casciati, editor, *Third World Conference on Structural Control*, volume 2, pages 431–437, Como, Italy, April 2002. International Association for Structural Control (IACS), John Wiley & sons.

J. F. Camino, J. W. Helton, R. E. Skelton, and J. Ye. Analyzing matrix inequalities systematically: How to get Schur complements out of your life. In *5th SIAM Conference on Control & Its Applications*, San Diego, CA, July 2001.

M. C. de Oliveira, J. F. Camino, and R. E. Skelton. A convexifying algorithm for the design of structured linear controller. In *Proceedings of the 39th IEEE Conference on Decision and Control*, pages 2781–2786, Sydney, Australia, December 2000.

J. F. Camino, J. W. Helton, and R. E. Skelton. A symbolic algorithm for determining convexity of A matrix function: How to get Schur complements out of your life. In *Proceedings of the 39th IEEE Conference on Decision and Control*, pages 5023–5026, Sydney, Australia, December 2000.

J. F. Camino, D. E. Zampieri, and P. L. D. Peres. $\mathcal{H}_2$ and $\mathcal{H}_\infty$ optimization techniques applied to A quarter-car suspension model. In *Proceedings PACAM VI/DINAME 99*, Rio de Janeiro, RJ, Brazil, January 1999.

J. F. Camino, D. E. Zampieri, and P. L. D. Peres. Design of A vehicular suspension controller by static output feedback. In *Proceedings of the American Control Conference*, pages 3168–3172, San Diego, CA, June 1999.

R. H. C. Takahashi, J. F. Camino, D. E. Zampieri, and P. L. D. Peres. A multiobjective approach for $\mathcal{H}_2$ and $\mathcal{H}_\infty$ active suspension control. In *Proceedings of the American Control Conference*, volume 1, pages 48–52, Philadelphia, Pennsylvania, June 1998.

J. F. Camino, D. E. Zampieri, and P. L. D. Peres. Aplicação da Técnica de controle robusto $\mathcal{H}_2$ em uma suspensão veicular – análise de falha. In *V Congresso de Engenharia Mecânica Norte-Nordeste*, Fortaleza, CE, Brazil, September 1998.

J. F. Camino, R. H. C. Takahashi, D. E. Zampieri, and P. L. D. Peres. Optimal active suspension design via convex analysis. In *Proceedings of the 1997 IEEE International Conference on Control Applications*, pages 411–416, Hartford, CT, October 1997.

J. F. Camino, R. H. C. Takahashi, D. E. Zampieri, and P. L. D. Peres. $\mathcal{H}_2$ and LQR active suspension control schemes with uncertain parameters: A comparison. In *Proceedings of the 7th International Conference on Dynamic Problems in Mechanics - DINAME 97*, pages 226–228, Angra dos Reis, RJ, Brazil, March 1997.

ABSTRACT OF THE DISSERTATION

Optimization over Convex Matrix Inequalities

by

Juan F. Camino
Doctor of Philosophy in Engineering Sciences
University of California, San Diego, 2003
Professor R. E. Skelton, Chair
Professor J. W. Helton, Co-Chair

This thesis provides tools which can solve a large class of optimization problems over Matrix Inequalities (MIs). This includes, for instance, many systems and control problems. To use these tools, no knowledge of Linear Matrix Inequalities (LMIs) is required. Furthermore, these tools preserve the advantages of the LMI framework.

To understand the motivations for this task, one must expose some of the advantages and disadvantages of the LMI framework. The wide acceptance of LMIs stems from the following facts: 1) if a control problem is posed as an LMI, then any solution is a global optimum; 2) efficient LMI solvers are readily available; 3) once a control problem is posed as an LMI, any other constraints in the form of LMIs can be added to the problem. On the other hand, the LMI framework has the following disadvantages: 1) there is no systematic way to produce LMIs for general classes of problems; 2) there is no way of knowing whether or not it is possible to reduce a system problem to an LMI without actually doing it; 3) the user must possess the knowledge of manipulating LMIs; 4) transformations via Schur complements can lead to a large LMI representation.

The main objective of this thesis is to provide a numerical optimization solver that possesses similar advantages to the LMI framework, but without its disadvantages. Our method has two components: 1) a numerical algorithm that solves a large class of matrix optimization problems; 2) a "Convexity Checker" which automatically provides a region, which if convex, guarantees that the solution from 1) is a global optimum on that region.

Unfortunately, not all MIs are convex, and a different strategy is required for nonconvex problems. In this thesis, the nonconvex problem of interest is the simultaneous design of the plant parameters and the control law. To overcome the difficulty of nonconvex MIs, a convexifying theory is presented which allow us to approximate the solution of this nonconvex integrated structure and control design by iterating on a sequence of convex subproblems. The convex subproblem is obtained by adding a potential function to the nonconvex constraints.

Chapter 1

General Introduction

Since the early 90's, Matrix Inequalities (MIs) have become very important in engineering, particularly, in control theory. It is unquestionable that the approaches that have been proposed in the field of optimization and control theory based on Linear Matrix Inequalities (LMIs) and semidefinite programming (SDP) have become very important and promising, since the same framework can be used for a large set of problems (Boyd et al. (1994); El-Ghaoui and Niculescu (1999); Rockafellar (1997); Skelton and Iwasaki (1995); Skelton et al. (1998)). Indeed, matrix inequalities provide a nice set up for many engineering and related problems, and if the MIs are convex, then the optimization problem is well behaved and the interior point methods provide efficient algorithms which are effective on moderate sized problems (Alizadeh et al. (1998); Nesterov and Nemirovskii (1994); Vandenberghe and Boyd (1996)).

In practice, optimization problems in engineering present matrix inequalities that require a large effort to determine their convexity. Nevertheless, using Schur complements and change of variables, many standard control problems have been found to possess an equivalent LMI formulation. A large collection of such control problems that can be posed as LMI, and the algorithms used to solve them, can be found in Boyd et al. (1994); Colaneri et al. (1997). In particular, the book by Skelton et al. (1998) shows that many control problems can be posed in the following linear form in X

$$AXB + B^T X A^T + Q < 0. \tag{1.1}$$

The parameterization of all feasible solutions X and the expression for the existence conditions for the above matrix inequality (1.1) are also given.

1.1 Optimization over Matrix Functions

If one has the ability to check whether or not an MI is convex and convertible to an LMI, then the optimization problem can be solved by the many available LMI solvers; however, if one does not have the ability to deal with LMIs, then it is not clear what one should do. An alternative is to restate the entire optimization problem in the form used by some particular numerical nonlinear optimization package. In this case, given that optimization over matrix functions are inherently not smooth, there is no guarantee of even a local minimum. Furthermore, the tedious process of reformulating a matrix optimization problem usually requires a high level of skill in algebra. Therefore, what has been missing up to now are the tools that can be used to reliably and certifiably solve optimization problems over matrix functions.

To convey what is meant by optimization over matrix functions, we give an example. Suppose one is given matrices of compatible dimensions A and S, and one needs to solve the following problem for symmetric matrices X and $Y > 0$ inside the unit ball:

$$\max_{X,Y} \operatorname{Tr}\{X\} \tag{P}$$

subject to

$$\begin{aligned} XA^TY^{-1}AX - AX(XA^TY^{-1}AX - Y)^{-1}XA^T - (Y^{-1}XA^TY^{-1}AXY^{-1} - Y^{-1})^{-1} \\ - AX(I + Y^{-1}XA^TY^{-1}AX)^{-1} - (I + XA^TY^{-1}AXY^{-1})^{-1}XA^T - S < 0. \end{aligned}$$

With the matrices A and S given by

$$A = \begin{bmatrix} 1 & -1 \\ 0 & 2 \end{bmatrix}, \qquad S = \begin{bmatrix} 2 & 0 \\ 0 & 1 \end{bmatrix}.$$

Later in Chapter 2, we give a physical example.

The above type of matrix optimization problem (P) can be solved using the tools provided in this thesis. These tools can solve a large class of convex optimization problems over matrix functions. This includes, for instance, many systems and control problems, or any other type of engineering problem that can be posed as matrix inequalities. To use these tools, no knowledge of Linear Matrix Inequalities is required.

To understand the motivations for this task, one must expose some of the advantages and the disadvantages of the LMI framework.

1.2 Advantage and Disadvantage of Linear Matrix Inequalities

The wide acceptance of LMIs stems from the following facts: a) if a control problem is posed as an LMI, then any solution is a global optimum; b) efficient LMI solvers are readily available; c) once a control problem is posed as an LMI, any other constraints in the form of LMIs can be added to the problem.

On the other hand, the LMI framework has the following disadvantages: 1) there is no systematic way to produce LMIs for general classes of problems; 2) there is no way of knowing whether or not it is possible to reduce a system problem to an LMI without actually doing it; 3) the user must possess the knowledge of manipulating LMIs; 4) transformations via Schur complements can lead to a large LMI representation.

Extending more on disadvantage 1), each area has a few special tricks which convert "lucky problems" into LMIs. Before there is any hope of producing LMIs systematically, it is important to know which types of constraint sets convert to LMIs and which do not. This appears to be a fundamental subject which remains to be explored. Although we do not deal with this question in this thesis, the papers by Helton (2002) do address some of these issues.

1.3 Our Approach to Solving Matrix Inequalities

Our method has two components: 1) a numerical algorithm, called NCSDP, that solves a large class of matrix optimization problems; 2) a symbolic "Convexity Checker" which automatically provides a region on which the solution from 1) is a global optimum.

The convexity checker presented in Chapter 3 is one of the main contributions of this thesis. The symbolic convexity checker algorithm takes as input a matrix function $F(X)$ and gives as output a family of inequalities that determine a domain $\mathcal{G}(X)$ on which $F(X)$ is "matrix convex." In this way, the checker guarantees that the solution obtained from the numerical solver is indeed a global minimum inside the region $\mathcal{G}(X)$, provided that $\mathcal{G}(X)$ is convex.

The numerical NCSDP optimization solver presented in Chapter 4 can be used to solve optimization problems corresponding to matrix inequalities. This approach does not require any knowledge of LMIs or any knowledge of how to manipulate MIs to be expressed

as LMIs. Consequently, there is no need to determine Schur complements in order to express the matrix constraints as LMIs. Moreover, since transformations via Schur complements can lead to an LMI representation with large matrices, the NCSDP solver has the potential to reduce the optimization time significantly when the dimensions of the matrices involved are large (see Section 1.4.3 in this introduction).

Putting together the convexity checker from Chapter 3 and the NCSDP solver from Chapter 4, we have a set of very powerful tools to solve many engineering problems that can be posed as matrix inequalities. These tools also provide a region, which if convex, guarantees that the solution of the NCSDP solver is a global minimum on that region. Suppose one has multiple matrix inequalities, then by using the convexity checker, it is possible to find a region on which all those inequalities happen to be convex. Once this region is determined, one can solve an optimization problem which takes into account this region of convexity. These tools answered two important questions:

1. How one can determine if an MI is or is not convex. (This is the convexity checker.)
2. If an MI is convex, how one can numerically solve an optimization problem without having to convert the MI into an LMI. (This is the NCSDP solver.)

In some sense, there is a parallel between the conventional "LMI approach" and our noncommutative approach. In the former, one needs to be able to convert the optimization problem over matrix functions into an equivalent LMI problem, so that some available LMI solver can be used. In the latter, the convexity checker is used to find a region $\mathcal{G}$ on which the MIs happen to be convex, and the NCSDP solver is used to solve the optimization problem inside this region $\mathcal{G}$. The next Section 1.4 describes this approach.

1.4 Introducing our Approach by an Example

We begin by describing our method in terms of an example which seem to conveniently illustrate our approach for dealing with matrix inequalities. For this purpose, we choose the optimization problem over matrix functions (P) and show that no knowledge of LMIs is required in order to solve it. (A realistic engineering problem is presented in Chapter 2.) Recall the optimization problem (P): suppose one is given matrices of compatible dimensions A and S, and one needs to solve the following problem for symmetric matrices X and $Y > 0$ inside the unit ball:

$$\max_{X,Y} \operatorname{Tr}\{X\} \tag{P}$$

subject to

$$XA^TY^{-1}AX - AX(XA^TY^{-1}AX - Y)^{-1}XA^T - (Y^{-1}XA^TY^{-1}AXY^{-1} - Y^{-1})^{-1}$$
$$- AX(I + Y^{-1}XA^TY^{-1}AX)^{-1} - (I + XA^TY^{-1}AXY^{-1})^{-1}XA^T - S < 0.$$

For this example, the unit ball can be represented by the following convex constraints:

$$XX < I \quad \text{and} \quad YY < I.$$

The matrices A and S are given by

$$A = \begin{bmatrix} 1 & -1 \\ 0 & 2 \end{bmatrix}, \qquad S = \begin{bmatrix} 2 & 0 \\ 0 & 1 \end{bmatrix}.$$

Using our methodology, this type of matrix optimization problem can be solved quite easily, while knowing nothing about LMIs. The steps are:

1. to determine the domain $\mathcal{G}$ on which the above problem is convex;

2. to solve the optimization problem using the numerical NCSDP solver over $\mathcal{G}$.

These steps enforce that the solution from the code is guaranteed to be a global minimum inside the region $\mathcal{G}$, provided that $\mathcal{G}$ is convex.

1.4.1 Step 1. Checking convexity

The region $\mathcal{G}$ is easily determined by invoking the **NCConvexityRegion[ ]** introduced in Part I of Chapter 3. We describe this step using standard TEX notation. For this purpose, let us define the domain:

$$\mathcal{G} = \{(X, Y, A, S) : F(X, Y, A, S) < 0, \quad Y > 0, \quad XX < I, \quad \text{and} \quad YY < I\} \tag{1.2}$$

with

$$F(X, Y, A, S) = -AX(XA^TY^{-1}AX - Y)^{-1}XA^T - (Y^{-1}XA^TY^{-1}AXY^{-1} - Y^{-1})^{-1} - S$$
$$- AX(I + Y^{-1}XA^TY^{-1}AX)^{-1} - (I + XA^TY^{-1}AXY^{-1})^{-1}XA^T + XA^TY^{-1}AX$$

for $X = X^T$ and $Y = Y^T$. The region of convexity for $\mathcal{G}$ is evidently the region where the function $F(X, Y, A, S)$ is matrix convex inside the unit ball for all $Y > 0$.

Since the convexity checker works at the noncommutative symbolic level, we must set the symbols appearing in the expression for $F(X, Y, A, S)$ to be noncommutative. Thus, we treat X, Y, A, and S symbolically as noncommutative indeterminate. To check the region of convexity for $F(X, Y, A, S)$, we apply the command:

NCConvexityRegion$[F,\ \{X,\ Y\}]$

which outputs the list

$$\{2Y^{-1},\ -2(XA^TY^{-1}AX-Y)^{-1},\ 2Y^{-1},\ 0,\ 0,\ 0,\ 0,\ 0,\ 0,\ 0,\ 0,\ 0\}.$$

Based on our theory, we know that $F(X,Y,A,S)$ will be convex in the domain that makes each entry in the list a positive definite expression, which, in this case, is the domain given by

$$2Y^{-1}>0 \qquad \text{and} \qquad -2(XA^TY^{-1}AX-Y)^{-1}>0.$$

Thus, from this output, we conclude that whenever A, S, X, and Y are matrices of compatible dimension, $F(X,Y,A,S)$ is simultaneously "matrix convex" in X and Y on the domain $\mathcal{G}_F$ given by

$$\mathcal{G}_F := \{(X,Y,A,S) : Y>0 \quad \text{and} \quad XA^TY^{-1}AX<Y\}. \tag{1.3}$$

To find if the above domain $\mathcal{G}_F$ is itself simultaneously convex in X and Y, we run the convexity checker once more on the function $G(X,Y,A)=XA^TY^{-1}AX-Y$,

NCConvexityRegion$[XA^TY^{-1}AX-Y,\ \{X,\ Y\}]$.

This command outputs the list $\{2Y^{-1},\ 0\}$. Thus, the region of convexity is $Y>0$, and consequently the domain $\mathcal{G}_F$ in (1.3) is convex. The optimization problem (P) will ultimately be convex inside the domain $\mathcal{G}\bigcap\mathcal{G}_F$.

1.4.2 Step 2. Invoking the NCSDP solver

The optimization problem (P) can now be solved with no great difficulty using the NCSDP solver provided in Chapter 4. To enforce that this optimization problem is convex, we need to add the following convex constraint:

$$XA^TY^{-1}AX<Y. \tag{1.4}$$

It should be realized that by adding the above constraint (1.4), we are not solving exactly the original problem, but instead, we are solving problem (P) inside its region of convexity. To proceed, let us define the objective for this optimization problem as

$$\text{obj} := -\operatorname{Tr}\{X\},$$

and let the G_i, representing the constraint $G_i < 0$, be given by

$$\begin{aligned} G_1 &:= F(X, Y, A, S) \\ G_2 &:= -Y \\ G_3 &:= YY - I \\ G_4 &:= XX - I \\ G_5 &:= XA^T Y^{-1} AX - Y \end{aligned}$$

The numerical data for this problem are

$$A = \begin{bmatrix} 1 & -1 \\ 0 & 2 \end{bmatrix}, \qquad S = \begin{bmatrix} 2 & 0 \\ 0 & 1 \end{bmatrix}.$$

With this data, we solve the minimization problem (P) using the NCSDP solver. In a figurative syntax[1], the call in Matlab is

NCSDP(obj, $\{G_1, G_2, G_3, G_4, G_5\}$, $\{X, Y\}$)

The solver returns the global optimum values for the unknowns X and Y given by

$$X^* = \begin{bmatrix} 0.3421 & 0.0263 \\ 0.0263 & 0.0788 \end{bmatrix}, \qquad Y^* = \begin{bmatrix} 0.8107 & 0.0016 \\ 0.0016 & 0.4255 \end{bmatrix}.$$

The optimal cost is therefore $\mathrm{Tr}\{-X^*\} = -0.4208$.

It is important to emphasize that the extra constraint $XA^T Y^{-1} AX < Y$ obtained from the convexity checker, allows us to solve a convex instance of the original problem, i.e., to solve the original problem inside its largest region of convexity, namely, closure($\mathcal{G} \bigcap \mathcal{G}_F$). This ensures that the solution is a global minimum inside this convex domain. Moreover, for the specific data in our example, the only constraint active at the optimal solution X^* and Y^* is $F(X, Y, A, S) \leq 0$.

1.4.3 Timing chart

We have just presented ideas on how to use the NCSDP solver and demonstrated some of its advantages as well. The question we address here is the comparison of speed;

[1]This is not the actual call for the solver. In its present implementation, the code receives as input the expressions for the constraints as Matlab strings. In this way, we could parse the data to Mathematica for symbolic manipulations.

since it is important to present a comparative analysis of the NCSDP solver with other SDP solvers. For this purpose, we present some of the results obtained in Section 4.7 from Chapter 4. The optimization problem used in this comparison, is not problem (P) from the previous section, but rather, another problem that also has an LMI representation. We do not restate the problem here, since the point is to show a qualitative comparison (see Section 4.7 for details).

The results are presented in Figure 1.1. The semidefinite programming solvers used were: SeDuMi, SP, SDPHA, and LMILab. These professional solvers are well known to be efficient for matrices of moderate size. The label MCLMI stands for a crude implementation of the method of centers for LMIs. It is important to emphasize that while the NCSDP solver is basically implemented using standard Matlab functions, most of the other solvers have their core subroutines compiled.

From Figure 1.1, one can conclude that, as the size of the matrices increase beyond 16×16, the timing of the NCSDP solver approximates the timing of the LMILab solver (Matlab LMI Toolbox) which was the fastest of the LMI solvers. Probably, for matrices of dimensions larger than 64×64, the NCSDP might be faster than the LMILab solver. We did not run this experiment for matrices of dimensions larger than 64×64 since the time would be extremely long.

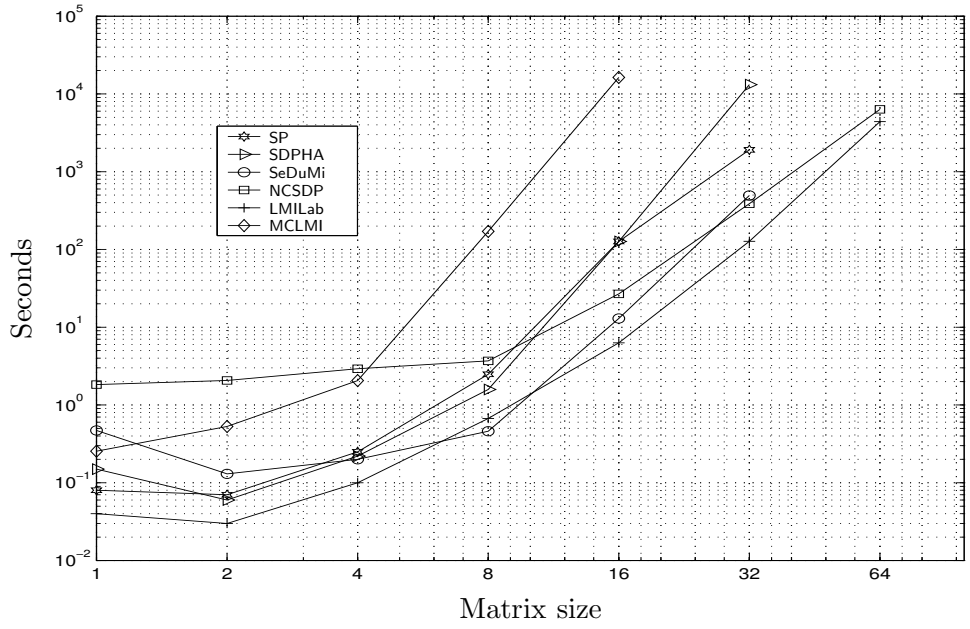

Figure 1.1: Performance of LMI Solvers

1.5 Convexity of Matrix Inequalities

At this point, one should realize that the main ingredient of our discussion concerning MIs and LMIs is convexity – specifically, the ability of determining if a matrix function is or is not convex (similarly, concave). If an MI can be similarly cast as an LMI, then it is evident that one has a convex problem. To achieve this, one must be able to recognize what type of transformation has to be applied to the original MI. Usually, this is accomplished by applying Schur complements. Sometimes, this transformation is clear, but in many other cases, it may not be immediately obvious. An example, is the matrix inequality $F(X, Y, A, S) < 0$ used in the previous optimization problem (P). This MI is given by

$$\begin{aligned} XA^TY^{-1}AX - AX(XA^TY^{-1}AX - Y)^{-1}XA^T - (Y^{-1}XA^TY^{-1}AXY^{-1} - Y^{-1})^{-1} \\ - AX(I + Y^{-1}XA^TY^{-1}AX)^{-1} - (I + XA^TY^{-1}AXY^{-1})^{-1}XA^T - S < 0 \end{aligned} \tag{1.5}$$

with $X \in \mathbb{S}$ and $S, Y \in \mathbb{S}_{++}$. Even though it may not be obvious at first glance, this MI is convex in X and Y, and can be transformed via Schur complements to the equivalent LMI

$$\begin{bmatrix} -S & AX + Y & 0 & XA^T \\ XA^T + Y & -Y & XA^T & 0 \\ 0 & AX & -Y & 0 \\ AX & 0 & 0 & -Y \end{bmatrix} < 0. \tag{1.6}$$

As seen from (1.6), this transformation led to an LMI representation four times larger than the original MI representation (1.5). Once the problem is in the linear form given by (1.6), there are many available LMI solvers. To cite a few of them: Gahinet et al. (1995); Sturm (1999); Vandenberghe and Balakrishnan (1997); Vandenberghe and Boyd (1995), and references therein.

For this example, we succeed in producing the LMI counterpart of the MI, however, as already emphasized, there is no systematic way to produce LMIs for general classes of problems. Thus, a natural question, is "how to identify if a matrix inequality is or is not convex." This is quite important, since applying Schur complements to determine convexity[2] of MIs may take time, cleverness, and large efforts. Yet, if one fails to set the problem as an LMI, it does not necessarily imply that the MI is nonconvex. The answer to this type of question is presented in Chapter 3, where a convexity checker for determining convexity of matrix functions is made available.

[2] A matrix inequality $F(X, Y) < 0$ been convex, means that the domain $\mathcal{G} := \{(X, Y) : F(X, Y) < 0\}$ is convex. This in turns implies that the function $F(X, Y)$ itself is matrix convex in X and Y.

1.6 Noncommutative Matrix Inequalities

It is important to understand if the variables in the matrix inequalities should be treated as commutative or noncommutative, and which mathematics one has available to deal with those formulas. For this purpose, let us discuss the various ways one could write a matrix inequality. As an example, the following convex MI

$$A^T X A + X B^T B X + Q < 0 \tag{1.7}$$

has the same form regardless of the dimension of the defining matrices A, B, Q, and X. In other words, if we take the matrices A, B, Q, and X to have compatible dimensions (regardless of what those dimensions are), then this inequality is meaningful and its form does not change. On the other hand, for this same example, once the dimensions of the matrices A, B, Q, and X are specified, it is also possible to write the above MI as a combination of known matrices L_0, L_1, ..., L_m, L_{11}, L_{12}, ..., L_{mm} of dimension $p \times p$ in unknown real numbers $x_1, \ldots, x_m$:

$$L_0 + \sum_{j=1}^{m} L_j x_j + \sum_{i=1}^{m} \sum_{j=1}^{m} L_{ij} x_i x_j < 0. \tag{1.8}$$

For example, if we assume that the dimension of the matrices in the inequality (1.7) are $A \in R^{2\times 2}, B \in R^{1\times 2}, Q \in R^{2\times 2}$, then $X^T = X \in R^{2\times 2}$ and consequently $m = 3$. The unknowns in the inequality (1.7) are the numbers x_i in

$$X = \begin{pmatrix} x_1 & x_2 \\ x_2 & x_3 \end{pmatrix}$$

For this set of matrices, the Ls are given by

$$L_{11} = \begin{pmatrix} b_{11}^2 & 0 \\ 0 & 0 \end{pmatrix} \qquad L_{22} = \begin{pmatrix} b_{12}^2 & b_{11}b_{12} \\ b_{11}b_{12} & b_{11}^2 \end{pmatrix} \qquad L_{21} = L_{12}^T$$

$$L_{12} = \begin{pmatrix} b_{11}b_{12} & b_{11}^2 \\ 0 & 0 \end{pmatrix} \qquad L_{23} = \begin{pmatrix} 0 & b_{12}^2 \\ 0 & b_{11}b_{12} \end{pmatrix} \qquad L_{31} = L_{13}^T$$

$$L_{13} = \begin{pmatrix} 0 & b_{11}b_{12} \\ 0 & 0 \end{pmatrix} \qquad L_{33} = \begin{pmatrix} 0 & 0 \\ 0 & b_{12}^2 \end{pmatrix} \qquad L_{32} = L_{23}^T$$

$$L_1 = \begin{pmatrix} a_{11}^2 & a_{11}a_{12} \\ a_{11}a_{12} & a_{12}^2 \end{pmatrix} \qquad L_3 = \begin{pmatrix} a_{21}^2 & a_{21}a_{22} \\ a_{21}a_{22} & a_{22}^2 \end{pmatrix} \qquad L_0 = Q$$

$$L_2 = \begin{pmatrix} 2a_{11}a_{21} & a_{12}a_{21} + a_{11}a_{22} \\ a_{12}a_{21} + a_{11}a_{22} & 2a_{12}a_{22} \end{pmatrix}$$

Now, if we consider that the dimensions of the matrices involved are $A \in R^{3\times 3}$, $B \in R^{3\times 3}$, $Q \in R^{3\times 3}$, and $X \in R^{3\times 3}$, then the formulas for the Ls, whose relationship to (1.8) takes a little while to figure out, are more complicated. This shows that formulas like (1.8) do not scale simply with the dimension of the matrices, while formula like (1.7) does. Thus, formulas like (1.7) are easier to manipulate with a pencil and paper (or with NCAlgebra) than formula (1.8). In this way, it is plausible to say that noncommutative inequalities behave better than commutative ones.

Commutativity also has its own advantages, given that unscalable formulas like (1.8) tend to hold more generally than scalable ones and that they do not contain too much special structure. Moreover, formulas like (1.7) have the disadvantage to be intrinsically noncommutative, so that a person must have skill with noncommutative calculations. Nevertheless, the properties that originated from the commutative point of view do not appear to provide any useful advantage regarding our approach for dealing with matrix inequality, on the other hand, noncommutativity provides a powerful setup to manage matrix inequality symbolically. Therefore to develop a basis for computer algebra packages which could assist engineers in manipulating matrix inequalities, one needs to use the theory behind noncommutative rational functions, which is addressed in great detail in Chapter 3.

1.6.1 Noncommutativity as the Only Option for Checking Convexity

We have just shown that noncommutativity for dealing with MIs provides a more elegant mathematical framework. In addition, it allows us to efficiently compute directional derivatives, to collect and to simplify terms in an expression, and thus, to generate the algebraic linear system of equation which provides the Newton direction. But, perhaps, the greatest advantage of the noncommutative framework is that noncommutativity is likely the only practical necessary and sufficient approach available for checking convexity of rational functions over matrices.

It is hard to imagine a way to implement a convexity checker other than by using symbolic computation, even on modest size problems. Moreover, with commuting variables, the Hessian matrix is big and its positivity must be checked at every point. Even if a sum of squares algorithm is successfully developed to check positivity of the Hessian, it would be practical with only a dozen or so variables. The advantage of noncommutative representations is that one letter Z stands for a matrix with n^2 commuting variables. This is a tremendous saving, which in most problems means the difference between being able to or not being able to run a convexity checker.

1.7 Solving the Linear System

As we will see in Chapter 4, another important advantage of using the theory behind noncommutative rational functions is that the Newton direction (implemented in our numerical solver) is obtained as the solution of a "matrix" algebraic linear equation. Basically, the algebraic linear system of equations, $\mathbb{H}(\delta_X) = \mathbb{Q}$, to be solved have the following structure:

$$\sum_i^N \mathcal{A}_i \delta_X \mathcal{B}_i + \sum_i^N \mathcal{B}_i^T \delta_X \mathcal{A}_i^T = \mathbb{Q} \tag{1.9}$$

where the $\mathcal{A}$'s and $\mathcal{B}$'s are obtained by collecting the terms on the left and on the right side of the update direction δ_X that appear inside the "Hessian map," denoted by $\mathbb{H}(\delta_X)$. The integer $2N$ has been defined as the Sylvester index by Konstantinov et al. (2000).

The above equation (1.9) provides the necessary conditions that the direction δ_X must satisfy in order to be a Newton direction. An important question, which is still open, is how this linear system can be solved efficiently. We present a rudimentary approach, which uses the vec operation. Using the properties of the vec operation, the matrix system (1.9) can be transformed in the equivalent vector form

$$\mathcal{H} v = g \tag{1.10}$$

where $\mathcal{H}$ is the Hessian matrix given by $\mathcal{H} = \sum_i^N \mathcal{B}_i^T \otimes \mathcal{A}_i + \sum_i^N \mathcal{A}_i \otimes \mathcal{B}_i^T$, the vector g is the gradient given by $g = \mathrm{vec}(\mathbb{Q})$, and v is the vector of unknowns given by $v = \mathrm{vec}(\delta X)$.

The final equation (1.10) is now in the conventional vector form, and can be solved by any conventional linear system solver. However, this "brute force" procedure does not take advantage of the particular structure of the Hessian map $\mathbb{H}(\delta_X)$, which is readily evident from (1.9). Naturally, after applying the vec operation, the linear system (1.10) somehow contains this "nice" structure, but from the knowledge of the author, there are no practical algorithm that can solve the linear system (1.10) taking into account the structure of $\mathbb{H}(\delta_X)$ for any arbitrary Sylvester index N.

For the simpler case of when $N = 1$, so that $\mathbb{H}(\delta_X) = \mathcal{A}\delta_X\mathcal{B} + \mathcal{B}^T\delta_X\mathcal{A}^T$, we have a "Lyapunov" type of algebraic equation, for which there are many available numerical and analytical results (see Chapter 7 of Golub and Loan (1983) and references therein). However, in the most general form where the Sylvester index can be any number, there is no satisfactory numerical algorithm, or even theoretical results, that can take advantage of the structure of the system. Some works in this area are Konstantinov et al. (2000). Since most of the running time of the optimization code is spent on solving the above linear

system, a satisfactory theory and algorithm would be valuable. We leave this major open area of work for others.

1.7.1 Improving Function Evaluations

Even though we will not present in great details how our optimization code is implemented, we feel that it is important to expose the main idea behind the implementation, mainly the fact that the algorithm can be split into two parts: a symbolic and a numerical part.

Roughly speaking, at the symbolic level, Mathematica computes the first and second directional derivatives of a log-barrier function that incorporate the objective and the constraints. From those derivatives one obtains the maps for the Hessian $\mathbb{H}(\delta_X)$ and for the Gradient $\mathbb{Q}$, building in this way an algebraic linear equation like (1.9). At this stage, the most important question is how one can symbolically simplify the final expressions such that when we substitute matrices for the symbols, the time spent on function evaluations can be minimized.

To attain this goal, we should observe that even if two symbolic rational functions may at first glance look different, they, in fact, can be totally equivalent. This frequently happens inside noncommutative rational functions containing a large number of terms. It is also important to collect terms in an expression. We show this with a very simple example, which appears, in practice, in a more complex fashion. Suppose one has an expression like

$$A_1\delta_X + \cdots + A_p\delta_X$$

To evaluate this expression, once that δ_X and the A_i are replaced by matrices, we need p matrix additions and also p matrix multiplications. On the other hand, collecting this expression in δ_X we obtain

$$\left(\sum_{i=1}^{p} A_i\right)\delta_X.$$

In this case, we need p matrix additions and only one matrix multiplication. In this example, the Sylvester index has dropped from p to 1. This represents a huge saving on the numerical cost of evaluating the above expression. Thus, the ability to decrease the Sylvester index by collecting factors in an expression, plays a very important role. The work by Helton et al. (1998) provides an efficient theory and algorithm to simplify noncommutative rational functions.

It is also true that at the symbolic level of Mathematica, the process of collecting terms on an expression and the process of simplifying rational functions, can consume a considerable amount of time. However, this computation is done only once in the begin of the run. This is in contrast with the numerical part, where the evaluation of the expression to provide the update direction takes place at each inner iteration (which occurs many times).

1.7.2 More on Collecting

Since the ability to collect terms on an expression significantly reduces the time spent on function evaluations, we show one more example, and we leave for Section 4.6 the numerical results showing how much time can actually be gained. Let us illustrate with the next example exactly what we mean by collecting terms in an expression. Suppose that the expression for the Hessian map $\mathbb{H}(\delta_X)$ is given by

$$\mathbb{H}(\delta_X) = A\delta_X A^T + X^T\delta_X X + B\delta_X B^T - A\delta_X X - X^T\delta_X A^T + B\delta_X A^T + A\delta_X B^T.$$

The Sylvester index in this case is seven. This expression can be collected in at least two different ways, having the same number of terms. One possibility is

$$\mathbb{H}(\delta_X) = (A - X^T)\delta_X(A - X^T)^T + (A + B)\delta_X(A + B)^T - A\delta_X A^T = \sum_{i=1}^{3} \mathcal{A}_i\delta_X\mathcal{B}_i$$

for $\mathcal{A}_i$ and $\mathcal{B}_i$ given by

$$\begin{aligned} \mathcal{A}_1 &= (A - X^T), & \mathcal{A}_2 &= (A + B), & \mathcal{A}_3 &= -A \\ \mathcal{B}_1 &= (A - X^T)^T, & \mathcal{B}_2 &= (A + B)^T, & \mathcal{B}_3 &= A^T \end{aligned}$$

Another one is

$$\mathbb{H}(\delta_X) = (A + B - X^T)\delta_X(A + B - X^T)^T + B\delta_X X + X^T\delta_X B^T = \sum_{i=1}^{3} \mathcal{A}_i\delta_X\mathcal{B}_i$$

for $\mathcal{A}_i$ and $\mathcal{B}_i$ given by

$$\begin{aligned} \mathcal{A}_1 &= (A + B - X^T), & \mathcal{A}_2 &= B, & \mathcal{A}_3 &= X^T \\ \mathcal{B}_1 &= (A + B - X^T)^T, & \mathcal{B}_2 &= X, & \mathcal{B}_3 &= B^T \end{aligned}$$

In both cases, the Sylvester index is now three, going down by less than half. It is now quite easy to see that a large reduction in the Sylvester index might happen. It is also evident from the above example, that this process is not at all unique.

1.8 Nonconvex Matrix Inequalities

Unfortunately, not all MIs are convex, and a different strategy is therefore required to deal with nonconvex MIs. An example of an engineering problem that is posed using nonconvex matrix inequality is presented in Chapter 5. The nonconvex problem of interest in this thesis is the simultaneous design of the plant parameters and the control law. It can be shown that the joint integrated structure and control design problem has the same mathematical structure as a decentralized output feedback control problem with the control gain matrix being diagonal, which is well known to be hard to solve.

We now give an idea of what makes the integrated structure and control problem nonconvex. Suppose our dynamic system is described by the following first-order differential equation

$$\dot{x} = Ax + Bu$$

where u is the control input and A, B are given matrices describing the dynamics of the system. To find a stabilizing state feedback control law, given by $u = Kx$, it suffices to solve the following matrix inequality for X and F

$$AX + XA + BF + F^T B^T < 0,$$

where the change of variable $F = KX$ was performed. Once X and F are determined, the control gain is promptly given by $K = FX^{-1}$. If A and B are fixed matrices, then the problem is clearly linear in X, however, if either A or B contain unknown terms, then the problem is nonconvex. This is exactly the type of structure that appears in this simultaneous design, since matrices A and B are no longer fixed matrices, as they may contain unknown parameters to be optimized, as an example: the mass, the stiffness and the damping ratio of the structure.

To overcome the difficulty of nonconvex MIs, a convexifying theory is presented in Chapter 4, which will allow us to approximate the solution of the nonconvex integrated structure and control design by iterating on a sequence of convex subproblems. The convex subproblem is obtained by adding a certain potential function to the nonconvex constraints. In practice, this added convexifying function disappears at stationary solutions of the original nonconvex problem.

Even though the convexifying theory presented in Chapter 4 is applied to the integrated structure and control design problem, the idea is broader and can be used to solve many other type of nonconvex control problems (de Oliveira et al. (2000)).

1.9 Summarizing the Main Ideas of Each Chapter

1.9.1 The Convexifying Theory

This section summarizes the main ideas behind the convexifying theory, which are presented in detail in Chapter 5. Let $\mathcal{V} \subset \mathbb{R}^{p\times q}$. Suppose one wishes to solve the following nonconvex optimization problem:

$$\min_{x\in\Omega} f(X), \quad \Omega := \{X \in \mathcal{V} : G(X) \leq 0\} \tag{1.11}$$

where $f(X) : \mathcal{V} \to \mathbb{R}$ is a linear function on the unknown $X \in \mathcal{V}$ and $G(x) : \mathcal{V} \to \mathbb{S}^n$ is a nonconvex matrix function. Then, we define a convexifying matrix function $H(X, Y) : \mathcal{V} \times \mathcal{V} \to \mathbb{S}^n$ for all $X, Y \in \mathcal{V}$ such that

$$G(X) + H(X, Y)$$

is now convex in X. This potential matrix function $H(X, Y)$ must posses some extra properties (Section 5.3):

i) the matrix $H(X, Y)$ is positive semidefinite for all X, Y;

ii) for all X, Y satisfying $\|X-Y\| < \delta$, there exists $\epsilon > 0$ such that $H(X, Y) \leq \epsilon\|X-Y\|$;

iii) for all X, Y satisfying $\|X-Y\| < \delta$, there exists $\epsilon > 0$ such that $H'(X, Y) \leq \epsilon\|X-Y\|$, where $H'(X, Y)$ is the derivative of $H(X, Y)$ in X.

Using these ideas, a simple algorithm for finding suboptimal solutions to the above nonconvex optimization problem is given by

Algorithm 1.9.1 *Let $\epsilon > 0$, $X^0 \in \Omega$ and a convexifying matrix function $H(X, Y)$ be given:*

1. *For $k = 0, 1, 2, \ldots,$ solve the convex optimization problem*

$$X^{k+1} = \arg\min_{X\in\Omega_k} f(X), \quad \Omega_k := \left\{X \in \mathcal{V} : G(X) + H(X, X^K) \leq 0\right\}. \tag{1.12}$$

2. *Until converges, go back to 1.*

The above convex problem is significantly simpler than (1.11), and we assume that its solution can be obtained by some available convex programming technique. Under quite strong assumptions, the X^K will converge to a stationary solution of the original nonconvex problem. Experiments have shown that this algorithm has been successful for the integrated structure and control problem presented in Chapter 5.

1.9.2 The Convexity Checker Algorithm

In Chapter 3, we provide a computer algebra algorithm that can be used to find the domain $\mathcal{G}$ of convexity of a noncommutative rational function F. This algorithm produces sufficient, and with some weak hypotheses, necessary conditions for convexity.

We now very loosely introduce the idea behind the algorithm even though we have not set down any formal definitions. Let F be the noncommutative rational function to be analyzed. Say F is a function of the noncommutative variables, $A_1, \ldots, A_m, X_1, \ldots, X_k$. The main steps of the algorithm are:

1. The second directional derivative with respect to $X_1, \ldots, X_k$, the Hessian $\mathcal{H}F$ of the function F, is computed.

2. As the Hessian is always a quadratic function of the update directions, it can be associated with a symmetric matrix $M_{\mathcal{H}F}$ with noncommutative entries.

3. The noncommutative $\mathrm{LDL}^{\mathrm{T}}$ factorization is applied to the coefficient matrix $M_{\mathcal{H}F}$.

4. And finally specifying positivity of the resulting diagonal[3] matrix $D(A_1, \ldots, A_m, X_1, \ldots, X_k)$ gives inequalities describing a region $\mathcal{G}$ of variables on which F is matrix convex.

While determining convexity of conventional commutative functions is extremely straightforward, noncommutativity imposes rather interesting complications. In particular, proving that the largest symbolic inequality domain on which F is matrix convex requires a substantial proof, mixing both linear algebra and algebraic representation type arguments.

1.9.3 The Optimization Solver for Matrix Functions

In Chapter 4, we propose a methodology where we can numerically solve convex optimization problems over matrix functions. The constrained optimization problem (COP) we are interested in can be posed as:

find f^*, if one exists, such that

$$f^* = \min \{ f(X) : X \in \text{closure}(\mathcal{G}) \} \tag{COP}$$

[3]If D is not diagonal, it contains 2×2 blocks which are never positive definite. See Section 3.4.

where the feasibility domain $\mathcal{G}$ is given by

$$\mathcal{G} = \Big\{ X \in \mathcal{C} : F_i(X) > 0,\ i = 1, \ldots, m \Big\}$$

and $\mathcal{C}$ is a bounded convex domain. The function $f(X) : \mathcal{C} \to \mathbb{R}$ is linear and the map $F_i : \mathcal{C} \to \mathbb{S}^{n_i}$ for each i is concave.

The idea behind the method is to replace the above constrained problem by a sequence of unconstrained convex minimization problems whose solutions eventually tend to the set of optimal solutions of (COP). In order to accomplish this, we need to define a barrier function for the feasibility domain $\mathcal{G}$. This barrier function, which we denote by $\Theta(X)$, has to be a smooth strongly convex function such that $\Theta(X) \to \infty$ for points converging to the boundary of the set $\mathcal{G}$. A usual barrier is the one given by

$$\Theta(X) = -\sum_{i=1}^{m} \log\det F_i(X) : \mathcal{G} \to \mathbb{R}.$$

With the barrier $\Theta(X)$ as defined above, the original problem (COP) could be approximated by a family of unconstrained problems of the form

$$X^*(\gamma) = \arg\min \Big\{ \log\left(1/(\gamma - f(X)\right) + \Theta(X) : X \in \mathcal{G}_\gamma \Big\} \tag{1.13}$$

where the feasibility set $\mathcal{G}_\gamma$ is given by

$$\{ X \in \mathcal{G} : f(X) < \gamma \}.$$

Under some mild conditions, the solution $X^*(\gamma)$ of (1.13) approaches the set of optimal solutions of (COP) for an appropriate sequence of decreasing centralization parameter γ.

1.10 The Layout of the Thesis

For clarity of presentation we have tried to make the chapters as independent from each other as possible. Thus, the reader will not need to be flipping pages over and over, and consequently, each chapter can be read in the order that is most suitable for him. For this purpose, most of the chapters have an introduction section and a notation section, producing a few overlaps of material. Sometimes, in order to avoid repeating certain definitions, there will be a reference to an earlier chapter that already provided them.

In this way, we believe the contributions of this thesis are more easily accessed, since each main contribution will be presented in an individual chapter. These contributions, as

seen from a bigger picture are: The convexifying theory and its application to the integrated structure and control design; the convexity checker; and a theory and a computer algorithm that solves convex optimization problem over matrix functions. Naturally, these are the main results, but many other new results are also derived in order to support these main ideas: for instance, the theory and implementation of an LDU algorithm for noncommutative rational functions.

This thesis is organized as follows. Chapter 2 illustrates the application of our method for solving matrix inequalities to an engineering design: an $\mathcal{H}_2/\mathcal{H}_\infty$ criteria for the design of active suspension control. Chapter 3 provides the convexity checker algorithm; its theory and its symbolic implementation. In Chapter 4, the theory behind the NCSDP solver is presented, along with many numerical experiments. Chapter 5 presents the convexifying theory, with an application to the integrated structure and control design. Chapters A-D are appendices which provide, among other results, Matlab codes and a list of testing problems for the NCSDP solver.

We reinforce that the notation used in Chapter 3 for the convexity checker is somewhat inconsistent with the notation used in the other chapters. In most of the chapters, we have used the same notation to stand either for a symbolic variable or for a variable which is a matrix, i.e., X could stand either for a noncommutative element or for a matrix of fixed dimension. On the other hand, given that in Chapter 3 we will be constantly substituting noncommutative elements by matrices of compatible dimensions, a more refined notation is needed. Thus, in Chapter 3, Euler-Script letters are frequently used to indicate the substitution of noncommutative elements by matrices of compatible dimensions. As an example, $F(X)$ means a noncommutative rational function whose argument X is a symbolic element; on the other hand, the Euler-Script $\mathcal{X}$ is used in $F(\mathcal{X})$ when $\mathcal{X}$ is a matrix in $\mathbb{R}^{n\times m}$.

Chapter 2

An $\mathcal{H}_2/\mathcal{H}_\infty$ Criteria for the Design of Active Suspension Control

To motivate the use of the tools provided in this thesis, the present chapter demonstrates how to solve an engineering design problem posed as matrix inequalities. No knowledge of LMIs or how to manipulate MIs to be expressed as LMIs is required. The optimization problem considered here is the design of an $\mathcal{H}_2/\mathcal{H}_\infty$ guaranteed cost controller for a vehicular suspension.

This will not be a comprehensive presentation on how to design active suspension systems, but rather on providing efficient tools for solving matrix inequalities. The reader which are not interested in the engineering setup and wants to see the application of our tools, might want to start from Section 2.5.

2.1 Introduction

Since the early 80's, optimal control techniques for improving vehicular suspensions have been fairly investigated. Those techniques can improve the performance of an automotive suspension significantly as shown in Camino et al. (1999); Hrovat (1991, 1993); Sharp and Crolla (1987). A vehicular suspension basically supports the vehicle weight, maintains stability along different types of maneuvers, offer a relative margin of comfort, and minimizes the effect of forces arising from the road disturbances. To describe those dynamics, a model suitable for control design (see Chalasani (1987); Takahashi et al. (2000); Thompson (1976)), which is largely used in the literature, is the quarter-car showed in Figure 2.1.

2.2 Dynamics of a Vehicular Suspension

For the two degree of freedom model presented in Figure 2.1, the coordinates x_1 represents the displacement from equilibrium of the unsprung mass m_1, the state x_2 is the displacement from equilibrium of the sprung mass m_2. The states x_3 and x_4 are their

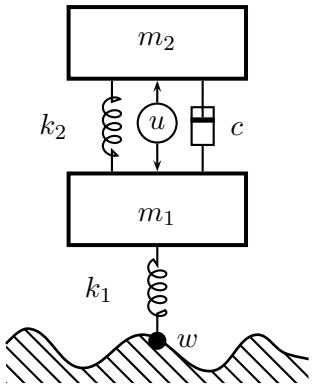

Figure 2.1: A 2-DOF suspension car

respective velocities. The road disturbance applied to the tire is w. The stiffness of the tire is k_1. For the suspension system, the damping coefficient is c and the spring stiffness is k_2. The force produced by the actuator is denoted by u. With these specifications, the equation of motions derived using Newton's laws is given by

$$\ddot{x}_1 = -\frac{c}{m_1}(\dot{x}_1 - \dot{x}_2) - \frac{k_2}{m_1}(x_1 - x_2) - \frac{k_1}{m_1}(x_1 - w) - \frac{1}{m_1}u$$
$$\ddot{x}_2 = \frac{c}{m_2}(\dot{x}_1 - \dot{x}_2) + \frac{k_2}{m_2}(x_1 - x_2) + \frac{1}{m_2}u.$$

These equations can be represented in the state-space form:

$$\dot{x} = Ax + B_u u + B_w w$$

where B_w is

$$B_w^T = \begin{bmatrix} 0 & 0 & k_1/m_1 & 0 \end{bmatrix}$$

and the matrices A, B_u are given by

$$A = \begin{bmatrix} 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ -(k_2+k_1)/m_1 & k_2/m_1 & -c/m_1 & c/m_1 \\ k_2/m_2 & -k_2/m_2 & c/m_2 & -c/m_2 \end{bmatrix}, \qquad B_u = \begin{bmatrix} 0 \\ 0 \\ -1/m_1 \\ 1/m_2 \end{bmatrix}$$

2.3 Control Strategy

The control design for a vehicular suspension system always involves a trade-off among several conflicting objectives, which are usually expressed by: 1) road holding ability associated with the tire deflection; 2) required rattle space, i.e., the relative space between the axle and the body of the car; 3) user discomfort associated with the acceleration of the sprung mass m_2. The index used to quantify those objective are respectively given by $x_1 - w$, $x_1 - x_2$, and the control effort u. Based on these criteria, one design methodology is to find a control law which minimizes the following quadratic cost function

$$J = \frac{1}{2}\int_0^\infty \left\{\alpha(x_1 - w)^2 + \beta(x_1 - x_2)^2 + \rho u^2\right\} dt \tag{2.1}$$

with the weights given by α, β, and ρ.

Naturally, for the appropriate choice of weighting matrices $Q = C_2^T C_2$ and $R = D_{2u}^T D_{2u}$ this cost function gives rise to an LQR design. This was the design proposed in Thompson (1976), where a suitable change of variable (to include the disturbance w) was applied to the system so that the designed closed-loop system would be of type I, having a zero steady-state offset to a step input w (similar to a regular spring-damper automobile suspension system). After applying the change of variables, the system and the cost function are given by

$$J = \frac{1}{2}\int_0^\infty z_2^T z_2 \; dt = \frac{1}{2}\int_0^\infty \left\{\hat{x}^T C_2^T C_2 \hat{x} + u^T D_{2u}^T D_{2u} u\right\} dt,$$

$$\dot{\hat{x}} = A\hat{x} + B_u u, \quad z_2 = C_2\hat{x} + D_{2u}u$$

where

$$D_{2u}^T D_{2u} = \rho, \qquad C_2^T D_{2u} = 0$$

and

$$C_2^T C_2 = \begin{bmatrix} \alpha+\beta & -\beta & 0 & 0 \\ -\beta & \beta & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix}, \qquad \hat{x} = \begin{pmatrix} x_1 - w \\ x_2 - w \\ x_3 \\ x_4 \end{pmatrix}$$

A main disadvantage of this LQR design is that the road disturbance must be available for feedback. To overcome this difficulty, we propose a different methodology based on the mixed $\mathcal{H}_2/\mathcal{H}_\infty$ control problem. In our approach, instead of a change of variable as done in the above design, we add an integrator given by $\dot{x}_5 = x_1 - x_2$, thereby enforcing that

the rattle space will have a zero steady-state offset to a step input. In this new approach, the states can be easily obtained from the suspension stroke and through integration of the acceleration of both the unsprung and the sprung mass. Allowing in this way a more realistic implementation.

Let us denote by $\mathcal{H}_{wz_1}$ the transfer function from the disturbance w to the unsprung mass displacement x_1. Since w is no longer available, and consequently the tire deflection $x_1 - w$ can no longer be included in the cost function for the new design, we bound the closed-loop gain of $\mathcal{H}_{wz_1}$ by a factor of 95% of the gain provided from the nominal system. From Figure 2.2, we found that $\|\mathcal{H}_{wz_1}\|_\infty^{NOM}$ for the nominal system is 1.2. Note that the magnitude in this Figure is given in dB, thus we have $20\log_{10}(1.2) = 1.5836$ dB.

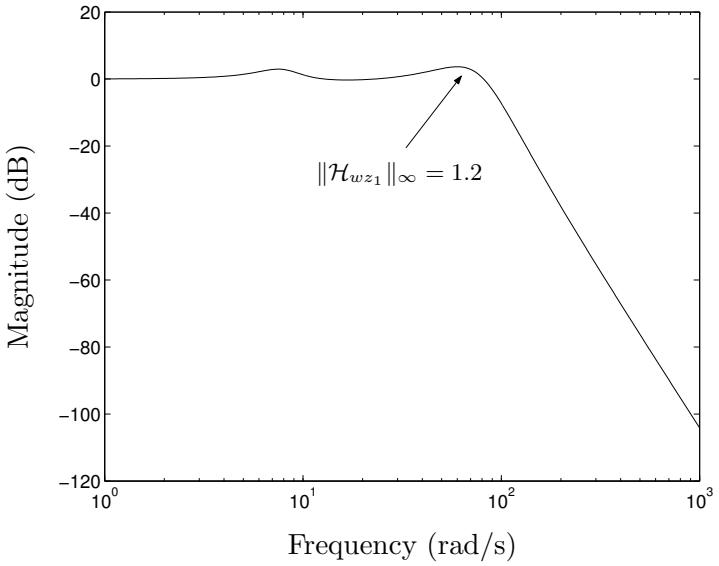

Figure 2.2: Bode diagram of the nominal system $\mathcal{H}_{wz_1}$

To achieve our purposes, we choose the controlled output $z_1 = C_1 x + D_{1w} w$ to represent the displacement of x_1, i.e.,

$$C_1 = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 \end{bmatrix} \qquad \text{and} \qquad D_{1w} = 0$$

This provides the $\mathcal{H}_\infty$ performance. For the $\mathcal{H}_2$ criteria, the cost function (2.1) becomes

$$J = \frac{1}{2}\int_0^\infty \left\{\overline{\alpha} x_1 + \zeta x_5^2 + \overline{\rho} u^2\right\} dt \tag{2.2}$$

where the new state x_5 is included. In this way, we have obtained a mixed $\mathcal{H}_2/\mathcal{H}_\infty$ control problem.

2.4 Multi-objective $\mathcal{H}_2/\mathcal{H}_\infty$ Control Design

Our control problem is depicted in Figure 2.3 below, where w is the disturbance acting on the system, the available vector y contains measurements, the vector u is the control law, and the output z_2 and z_1 are the controlled output for the $\mathcal{H}_2$ and $\mathcal{H}_\infty$ performance criteria.

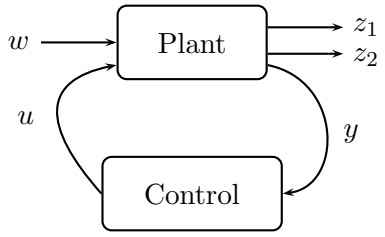

Figure 2.3: Mixed $\mathcal{H}_2/\mathcal{H}_\infty$ control problem

The state-space representation for this system is given by

$$\dot{x} = Ax + B_u u + B_w w$$
$$z_2 = C_2 x + D_{2u} u$$
$$z_1 = C_1 x + D_{1u} u + D_{1w} w$$

We assume that all the states are available for feedback, thus the control law is given by

$$u = Kx$$

with K being the constant gain to be determined. The closed-loop system is now given by

$$\dot{x} = (A + B_u K)x + B_w w$$
$$z_2 = (C_2 + D_{2u} K)x$$
$$z_1 = (C_1 + D_{1u} K)x + D_{1w} w$$

For this configuration, we can pose our mixed $\mathcal{H}_2/\mathcal{H}_\infty$ control problem as: find a control gain K, if one exists, such that

$$\alpha^* = \min_K \|H_{wz_2}\|_2 \qquad \text{subject to}$$
$$\|H_{wz_1}\|_\infty < \eta$$

where η is a given positive number, and H_{wz_2} and H_{wz_1} are the transfer functions from the disturbance w to z_2 and from w to z_1, given respectively by

$$H_{wz_2} := (C_2 + D_{2u} K)\left[sI - (A + B_u K)\right]^{-1} B_w$$
$$H_{wz_1} := (C_1 + D_{1u} K)\left[sI - (A + B_u K)\right]^{-1} B_w + D_{1w}$$

Thus, the idea behind our control design is to minimize the $\mathcal{H}_2$ norm of the transfer function of the closed-loop system meanwhile its $\mathcal{H}_\infty$ norm is bounded by a constant scalar $\eta > 0$. We pose the mixed $\mathcal{H}_2/\mathcal{H}_\infty$ control problem using the parameterization for the $\mathcal{H}_2$ and $\mathcal{H}_\infty$ performance given in Iwasaki and Skelton (1994); Skelton et al. (1998). To accomplish this, we show how one can compute the $\mathcal{H}_2$ and $\mathcal{H}_\infty$ performance criteria using matrix inequalities.

For a given control gain K, the $\mathcal{H}_2$ norm of the closed-loop system does not exceed a positive scalar μ, if symmetric matrices Q and $X_2 > 0$ exist such that:

$$\begin{gathered}\mathrm{Tr}\,\{Q\} < \mu^2 \\ Q - (C_2 + D_{2u}K)X_2(C_2 + D_{2u}K)^T > 0 \\ (A + B_uK)X_2 + X_2(A + B_uK)^T + B_wB_w^T < 0\end{gathered}$$

These equations are obtained from the controllability Grammian of the closed-loop system. In a similar way, for a given control gain K, the $\mathcal{H}_\infty$ norm of the closed-loop system does not exceed a positive scalar η, if a symmetric matrix $X_\infty > 0$ exists such that:

$$\begin{aligned}(A + B_uK)X_\infty + X_\infty(A + B_uK)^T + B_wB_w^T \\ + \left[X_\infty(C_1 + D_{1u}K)^T + B_wD_{1w}^T\right] R^{-1} \left[X_\infty(C_1 + D_{1u}K)^T + B_wD_{1w}^T\right]^T < 0\end{aligned}$$

with $R = \eta^2 I - D_{1w}D_{1w}^T > 0$.

In order to solve this problem, it is a usual compromise to require that the above Lyapunov matrices X_2 and X_∞ be identical. Thus, $X = X_2 = X_\infty$ for both performance objectives. Applying the change of variable $Y = KX$, we can pose our $\mathcal{H}_2/\mathcal{H}_\infty$ guaranteed cost control problem as:

$$\begin{gathered}\min \mathrm{Tr}\,\{Q\} \\ X > 0 \\ Q - (C_2X + D_{2u}Y)X^{-1}(C_2X + D_{2u}Y)^T > 0 \\ AX + XA^T + B_uY + Y^TB_u^T + B_wB_w^T \\ + \left[XC_1^T + Y^TD_{1u}^T + B_wD_{1w}^T\right] R^{-1} \left[XC_1^T + Y^TD_{1u}^T + B_wD_{1w}^T\right]^T < 0\end{gathered} \tag{2.3}$$

with $R = \eta^2 I - D_{1w}D_{1w}^T > 0$.

2.5 Solving the Design Problem

An important issue now is how one can solve the above MI problem given in (2.3). If one has the ability to check wheter or not this problem is convertible to an LMI, then the

optimization problem can be solved by the many available LMI solvers; however, if one does not have the ability to deal with LMIs, it is not clear what one should do; since optimization over matrix functions are inherently not smooth.

Using our methodology, this type of matrix inequality optimization problem can be solved quite easily, while knowing nothing about LMIs. The steps are just:

1. to determine the domain $\mathcal{G}$ on which the above MI problem is convex;
2. to solve the optimization problem using the numerical NCSDP solver over $\mathcal{G}$.

These steps, enforces that the solution from the code is guaranteed to be a global minimum on the region $\mathcal{G}$, provided that $\mathcal{G}$ is convex .

2.5.1 Step 1. Checking convexity

The region $\mathcal{G}$ is easily determined by invoking the **NCConvexityRegion[]** introduced in Part I of Chapter 3. We describe this step using the standard notation in Mathematica/NCAlgebra. For this purpose, let us define the domain:

$$\mathcal{G} = \{X \mid F_1(X,Q,Y) > 0 \quad \text{and} \quad F_2(X,Q,Y) > 0\}$$

with

$$\begin{aligned} F_1(X,Q,Y) &:= Q - (C_2X + D_{2u}Y)X^{-1}(C_2X + D_{2u}Y)^T \\ F_2(X,Q,Y) &:= -\Big(AX + XA^T + B_uY + Y^TB_u^T + B_wB_w^T \\ &\quad + \left[XC_1^T + Y^TD_{1u}^T + B_wD_{1w}^T\right] R^{-1} \left[XC_1^T + Y^TD_{1u}^T + B_wD_{1w}^T\right]^T\Big) \end{aligned}$$

The region of convexity $\mathcal{G}$ is evidently the region where the functions $F_1(X,Q,Y)$ and $F_2(X,Q,Y)$ are matrix concave.

Since the NCConvexityRegion command works at the noncommutative symbolic level, we must set the symbols appearing in the expression for $F_1(X,Q,Y)$ and $F_2(X,Q,Y)$ as noncommutative. In Mathematica/NCAlgebra this is done by:

```
In[1]:= SNC[Q, X,Y,A, C2, D2u, Bu, Bw, C1, R, D1u, D1w];
```

Now, let us define the function $F_1(X,Q,Y)$ and $F_2(X,Q,Y)$ in Mathematica:

```
In[2]:= F1 = Q - (C2 ** X + D2u ** Y) ** inv[X] ** (X ** tp[C2] + tp[Y] ** tp[D2u]);
```

In[3]:= F2 = -(A ** X + X ** tp[A] + Bu ** Y + tp[Bu ** Y] + Bw ** tp[Bw] + (X ** tp[C1] + tp[D1u ** Y] + Bw ** tp[D1w]) ** inv[R] ** (C1 ** X + D1u ** Y + D1w ** tp[Bw]));

To check the region of convexity for $F_1(X, Q, Y)$, we apply the command:

In[4]:= **NCConvexityRegion**[F1, {X,Y,Q}]

which outputs the list:

$$\{-2X^{-1},\ 0\}$$

From this output, we conclude that the function $F_1(X, Q, Y)$ is matrix concave on the domain $\mathcal{G}_1$ given by

$$\mathcal{G}_1 := \{(X, Y, Q)\ :\ X > 0\}.$$

To check the region of convexity for $F_2(X, Q, Y)$, we apply the command:

In[5]:= **NCConvexityRegion**[F2, {X,Y,Q}]

which outputs the list:

$$\{-2C_1^T R^{-1} C_1,\ 0\}$$

since $C_1^T R^{-1} C_1 \geq 0$ by assumption, we conclude that the function F_2 is matrix concave.

This result tells us that the optimization problem as stated in (2.3) is convex for all $X > 0$. Moreover, whichever solution our NCSDP solver returns, for the above problem (2.3) with the constraint $X > 0$, this solution is guaranteed to be a global minimum.

2.5.2 Step 2. Invoking the NCSDP solver

We now provide the numerical data used for simulation purpose. The nominal parameters of the system are given by $m_1 = 28.58$ kg, $m_2 = 288.90$ kg, $k_1 = 155900$ N/m, $k_2 = 19960$ N/m, and $c = 1861$ Ns/m. For the $\mathcal{H}_2$ performance, we used $\overline{\alpha} = 2.5$, $\zeta = 100$, and $\overline{\rho} = 8 \times 10^{-10}$. The imposed bound η on the $\mathcal{H}_\infty$ norm of the transfer function from w to z_1 was $\eta = 0.95 \times 1.2 = 1.14$. With these data, we solve the control problem posed in (2.3) using the NCSDP solver. The solver returns the global optimum values for the unknowns Q, Y, and X given by:

$$Q = 10^2 \times \begin{bmatrix} 0.598437 & 0.381385 & -0.241154 \\ 0.381385 & 1.563809 & -0.063589 \\ -0.241154 & -0.063589 & 0.262757 \end{bmatrix},$$

$$Y = 10^7 \times \begin{bmatrix} -0.014988 & -0.054751 & 7.248507 & -0.358297 & -0.008526 \end{bmatrix}$$

and

$$X = 10^5 \times \begin{bmatrix} 0.000695 & 0.000191 & -0.018557 & 0.005479 & 0.000025 \\ 0.000191 & 0.000710 & -0.015702 & -0.001408 & 0.000050 \\ -0.018557 & -0.015702 & 3.516382 & -0.167338 & -0.001861 \\ 0.005479 & -0.001408 & -0.167338 & 0.066279 & 0.000145 \\ 0.000025 & 0.000050 & -0.001861 & 0.000145 & 0.000006 \end{bmatrix}$$

Thus, for the imposed bound $\eta = 1.14$, the guaranteed $\mathcal{H}_2$ performance is $\sqrt{\mathrm{Tr}\,\{Q\}} = \sqrt{242.5} = 15.57$. This is also seen from the iteration log of the code presented in Table 2.1. The controller $K_{\mathcal{H}_2/\mathcal{H}_\infty} = YX^{-1}$ is given by

$$K_{\mathcal{H}_2/\mathcal{H}_\infty} = 10^5 \times \begin{bmatrix} 0.760705 & -0.696518 & 0.000586 & -0.089321 & 3.535528 \end{bmatrix}$$

Iteration log of the Code

The iteration log for this optimization problem is presented in Table 2.1, where the first column NeNe shows the number of Newton steps required to compute the analytic center, within an accuracy of 10^{-3} (i.e. $\tau < 10^{-3}$). The second column shows the norm of the gradient vector g, the third column presents the step length σ, the fourth column shows the value of $\mathrm{Tr}\,\{Q\}$, and the last two columns present the minimum and the maximum eigenvalue of the Hessian matrix $\mathcal{H}$. The code stops when the upper bound γ (centralization parameter) between two successive iterations, $\gamma^{k+1} - \gamma^k$, is less than 10^{-5}. Note that the table does not show every iteration. For a more detailed description of these parameters see Section 4.4.4.

Table 2.1: An $\mathcal{H}_2/\mathcal{H}_\infty$ control design for a vehicular suspension car

NeNe	$\|g\|$	τ	σ	$\mathrm{Tr}\,\{Q\}$	$\lambda_{\min}(\mathcal{H})$	$\lambda_{\max}(\mathcal{H})$
Iteration 1				γ = 5.8425445E+09		
1	9.8E-03	3.9E+00	0.2	2.7590497E+09	1.0E-19	2.1E-03
2	8.3E-03	3.7E+00	0.2	2.5719360E+09	8.1E-20	1.8E-03
3	7.0E-03	3.5E+00	0.2	2.3625135E+09	6.3E-20	1.5E-03
4	5.9E-03	3.2E+00	0.2	2.1396897E+09	5.1E-20	1.2E-03

continued on next page

continued from previous page

NeNe	$\|g\|$	τ	σ	$\mathrm{Tr}\{Q\}$	$\lambda_{\min}(\mathcal{H})$	$\lambda_{\max}(\mathcal{H})$
⋮		⋮		⋮		
24	2.5E-04	9.6E-01	0.5	1.2784479E+09	3.9E-20	6.9E-05
25	1.7E-04	4.3E-01	0.7	1.2884245E+09	5.2E-20	6.8E-05
26	7.5E-05	1.0E-01	1.0	1.2913954E+09	6.3E-20	6.7E-05
27	6.4E-06	3.4E-03	1.0	1.2913429E+09	6.6E-20	6.6E-05
Iteration 2				γ = 3.7976539E+09		
1	1.8E-08	1.9E+00	0.3	1.1406628E+09	6.6E-20	6.6E-05
2	1.4E-05	1.5E+00	0.4	9.8653968E+08	6.5E-20	6.7E-05
3	2.9E-05	9.8E-01	0.5	8.5279649E+08	6.4E-20	6.9E-05
4	3.7E-05	4.8E-01	0.7	7.6725520E+08	6.2E-20	7.1E-05
5	2.4E-05	1.4E-01	1.0	7.3283248E+08	6.0E-20	7.3E-05
6	2.1E-06	3.3E-03	1.0	7.3359799E+08	6.0E-20	7.4E-05
			⋮	⋮		
Iteration 66				γ = 2.4250082E+02		
1	1.8E+05	2.0E+00	0.3	2.4250061E+02	3.4E-10	4.3E+13
2	1.3E+05	1.7E+00	0.4	2.4250059E+02	4.5E-10	5.0E+13
3	8.7E+04	1.3E+00	0.4	2.4250058E+02	4.4E-10	6.0E+13
4	5.3E+04	8.4E-01	0.5	2.4250057E+02	4.9E-10	7.3E+13
5	2.5E+04	4.2E-01	0.7	2.4250056E+02	4.5E-10	8.8E+13
6	7.6E+03	1.3E-01	1.0	2.4250055E+02	6.0E-10	1.0E+14
7	1.3E+01	2.2E-04	1.0	2.4250055E+02	6.4E-10	1.1E+14
Iteration 67				γ = 2.4250068E+02		
1	2.9E+05	2.0E+00	0.3	2.4250055E+02	6.1E-10	1.1E+14
2	2.1E+05	1.7E+00	0.4	2.4250054E+02	7.3E-10	1.3E+14
3	1.4E+05	1.3E+00	0.4	2.4250053E+02	8.0E-10	1.5E+14
4	8.3E+04	8.4E-01	0.5	2.4250052E+02	7.5E-10	1.8E+14
5	4.0E+04	4.2E-01	0.7	2.4250052E+02	5.8E-10	2.2E+14
6	1.2E+04	1.3E-01	1.0	2.4250051E+02	9.9E-10	2.6E+14
7	2.1E+01	2.3E-04	1.0	2.4250051E+02	1.2E-09	2.7E+14

2.6 Comparing the $\mathcal{H}_2/\mathcal{H}_\infty$ Design against the LQR Design

For the purpose of comparison, we also calculate a LQR controller using the cost function given in (2.1), which we denote by K_{LQR}. The weights used for the LQR were $\alpha = 10$, $\beta = 1$, and $\rho = 8 \times 10^{-10}$. The LQR controller is given by

$$K_{LQR} = 10^4 \times \begin{bmatrix} 5.247840 & -2.064051 & 0.039623 & -0.263124 \end{bmatrix}$$

Figure 2.4 shows the Bode diagram of $\mathcal{H}_{wz_1}$, which is related to the road holding ability for the nominal system and for the new system using the $\mathcal{H}_2/\mathcal{H}_\infty$ controller.

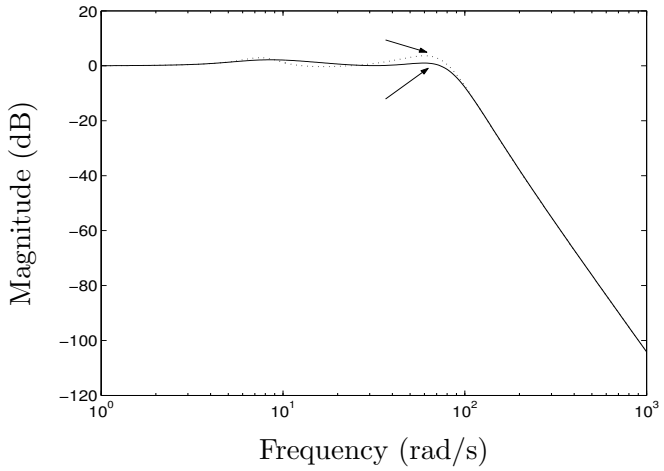

Figure 2.4: Bode diagram of $\mathcal{H}_{wz_1}^{NOM}$ and $\mathcal{H}_{wz_1}^{\mathcal{H}_2/\mathcal{H}_\infty}$

The mixed $\mathcal{H}_2/\mathcal{H}_\infty$ design is presented in solid line (-) and the passive design is shown by the dotted line ($\cdot$). As one can see, the new system has a smaller $\mathcal{H}_\infty$ norm, within the magnitude of $20\log_{10}(1.14) = 1.1381$, which was the imposed upper bound $\eta = 1.14$. Consequently, a better margin of safety for the vehicular suspension system is provided.

We also simulate the step response of the closed-loop system using the $\mathcal{H}_2/\mathcal{H}_\infty$ controllers and the LQR design. These results are presented in Figure 2.5 for the displacement x_1 of the unsprung mass m_1 and for the suspension stroke $x_1 - x_2$. The LQR design is plotted using a dash-dot line ($\cdot-$). From this plot, we see that the mixed $\mathcal{H}_2/\mathcal{H}_\infty$ control design possesses a very similar performance to the LQR design for a step input, giving a slightly better performance regarding the rattle space.

When compared to the nominal system, with no control, both designs have significantly better performance. It is also important to observe the acceleration of the sprung

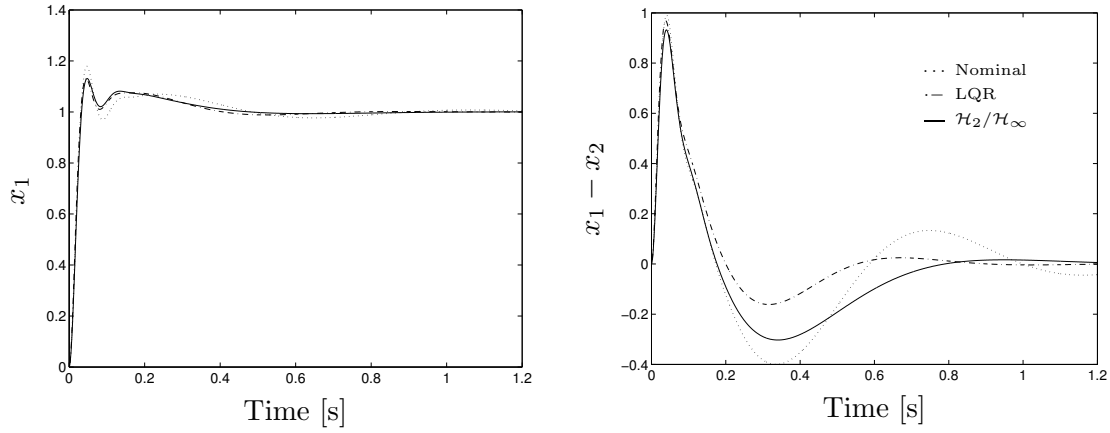

Figure 2.5: Step response for x_1 and $x_1 - x_2$

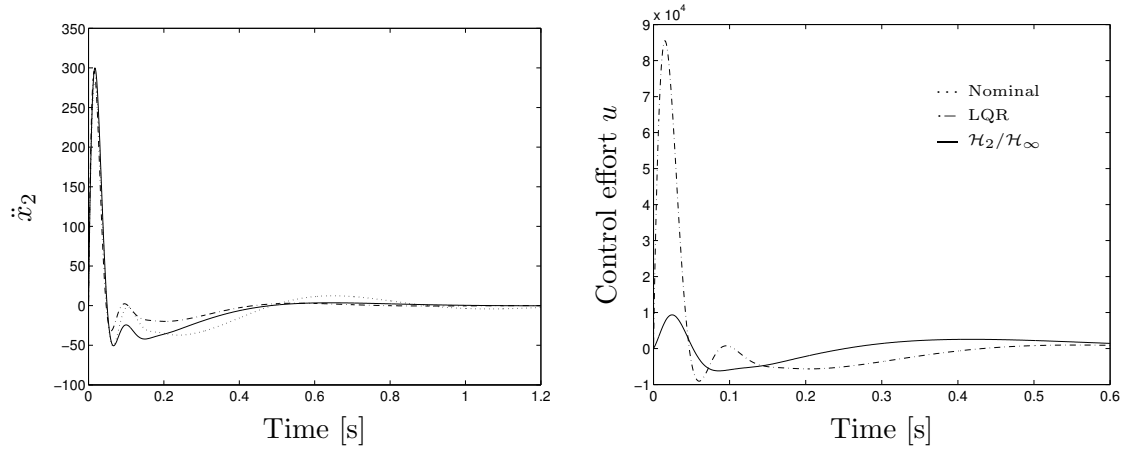

Figure 2.6: Sprung mass acceleration $\ddot{x}_2$ and control effort u

mass x_2, which is related to the user ride comfort. This plot is presented in Figure 2.6 along with the control effort u. The nominal system and the active systems provide similar ride comfort. However, the mixed $\mathcal{H}_2/\mathcal{H}_\infty$ requires significantly less control effort u than the counterpart LQR design, by a factor of 8. This is a large saving.

It is not surprising that active suspensions can overcome the performance of the counterpart passive system. One disadvantage of active suspension designs are that they need actuators able to produce large forces. However, our design requires significantly less control effort. Another important advantage is that our design does not require measurement of the road disturbance.

Chapter 3

Convexity Checker

3.1 Introduction

This chapter is split into two parts. Part I of this chapter presents our algorithm, describes its implementation and illustrates its effect on a few examples. We prove in Part II that the region $\mathcal{G}$ of convexity which our algorithm determines is the largest possible in a certain sense. The results in Part II give a satisfying theory of "matrix convexity" and of "matrix positivity" of noncommutative quadratic functions of a certain type. This part contains a bit of redundancy in order to maximize the reader base. Throughout the presentation of our algorithm we insert actual calls to symbolic routines in NCAlgebra, since this makes clear exactly what can be computed automatically.

Part I should be accessible to readers from many areas, from operator or matrix theory, from symbolic computation, and from engineering who work with matrix inequalities. It is organized as follows. Section 3.2 gives preliminary definitions about noncommutative rational functions, convexity, positivity, and derivatives. Section 3.3 concerns quadratic noncommutative functions $\mathcal{Q}$. It gives a representation for $\mathcal{Q}$ in terms of a symmetric matrix $M_{\mathcal{Q}}$ with noncommutative entries and it provides an algorithm to compute the LDU decomposition of $M_{\mathcal{Q}}$. Section 3.4 gives the convexity algorithm that provides the tools for checking the positivity and presents some examples. Section 3.5 illustrates how the algorithm when implemented using the noncommutative algebra package NCAlgebra can be used to find the region of convexity of a noncommutative rational function.

Now we describe the organization of Part II. Section 3.6 formally states and proves a theorem to the effect that our Convexity Algorithm produces a domain $\mathcal{G}$ in which the

given function is convex. This is easy and informative. Section 3.7 gives formal definitions. Section 3.8 states theorems to the effect that $\mathcal{G}$ is the biggest domain of convexity in a certain sense. Sections 3.9 and Section 3.10 gives proofs of the theorems stated in Section 3.8 and Section 3.3.

Section A is an appendix which describes a computer algorithm for representing a noncommutative quadratic function of k variables $H_1, \ldots, H_k$ in terms of a matrix $M_{\mathcal{Q}}$. This matrix plays an important role in determining the positiveness of a noncommutative rational function.

3.2 Notational Section for the Convexity Checker

The operator $(\cdot)^{-1}$ and $(\cdot)^T$ means the inverse and the transpose respectively. In order to make an expression symmetric the operator sym, defined as $sym[M] = M + M^T$, is used. The arrow over a variable is used to indicate that the variable is a list of elements $\vec{X} = \{X_1, \ldots, X_k\}$. If $\vec{X}$ contains only one indeterminate, then the notation is $\vec{X} = X$. Roman upper case letters will commonly represent symbolic elements, and also matrices when it is clear by context. Euler-Script letters are frequently used to indicate the substitution of noncommutative elements by matrices of compatible dimensions. As an example, $\Gamma(X)$ means a noncommutative rational function whose argument X is a symbolic element; on the other hand, the Euler-Script $\mathcal{X}$ is used in $\Gamma(\mathcal{X})$ when $\mathcal{X}$ is a matrix in $\mathbb{R}^{n\times m}$. Another example appears in the definition of the set $\mathcal{R}_L^x := \{(\mathcal{H}Lx) : \text{ all } \mathcal{H} \in \mathbb{R}^{n\times m}\}$ where L is a noncommutative rational function evaluated on certain matrices, $\mathcal{H}$ is a matrix, and x is a vector. Note that we do not use the Euler-Script font for vectors and functions. Even if the argument of the function L is a matrix $\mathcal{Z}$ rather than an indeterminate Z, we would have used $L(\mathcal{Z})$ instead of $\mathcal{L}(\mathcal{Z})$, and often we abbreviate $L(\mathcal{Z})$ to L. We reinforce that this notation is somewhat inconsistent with the notation used in the other chapters. It is more refined in that it carefully distinguishes between symbolic (noncommutative) variables and variables which are matrices.

3.2.1 Noncommutative symmetric rational functions

In this section we present useful definitions and facts about noncommutative rational functions. In fact, the development in this section follows Helton and Merino (1997) and Helton and Merino (1998).

We begin with definitions of noncommutative rational functions, of derivatives of noncommutative functions, and of convexity. Next, the procedure to represent a quadratic function together with the noncommutative LDU decomposition is illustrated. Also the idea behind necessary and sufficient conditions for positivity of noncommutative quadratic functions is introduced. Later in Section 3.4, our Convexity Algorithm is described and then, in Section 3.5, it is illustrated by some examples.

What occurs in practice are functions Γ which are polynomial or rational in noncommutative variables (often referred to as indeterminates) with coefficient which are real numbers. Noncommutative rational functions of X are polynomials in X and in inverses of polynomials in X. Examples of noncommutative symmetric functions are

$$\Gamma(A,B,X) = AX + XA^T - \frac{3}{4}XBB^TX, \qquad X = X^T,$$

$$\Gamma(A,D,X,Y) = X^TAX + DYD^T + XYX^T, \qquad Y = Y^T \quad \text{and} \quad A = A^T, \tag{3.1}$$

and

$$\Gamma(A,D,E,X,Y) = A(I + DXD^T)^{-1}A^T + E(YXY^T)E^T, \qquad X = X^T. \tag{3.2}$$

We also assume there is an involution on these rational functions which we denote superscript T, and which will play the role of transpose later when we substitute matrices for the indeterminates.

Often we shall think of some indeterminates as knowns and other indeterminates as unknowns and be concerned primarily about a function's properties with respect to unknowns. For example, in function (3.2) when we are mainly concerned about behavior such as convexity of Γ in X, Y we write $\Gamma(A,D,E,X,Y)$ simply as $\Gamma(X,Y)$. We also use $\vec{Z}$ to abbreviate all indeterminates which appear in the function, for example, in (3.2) we have $\vec{Z} = \{A,D,E,X,Y\}$. Often we distinguish knowns $\vec{A} = \{A_1,\ldots,A_m\}$ from unknowns $\vec{X} = \{X_1,\ldots,X_k\}$ by writing $\vec{Z} = \{\vec{A},\vec{X}\}$. Throughout this chapter, letters near the beginning of the alphabet denote knowns, while the letters X, Y stand for unknowns.

We call a noncommutative function $\Gamma(\vec{A},\vec{X})$ **symmetric** provided that $\Gamma(\vec{A},\vec{X})^T = \Gamma(\vec{A},\vec{X})$. If all $X_1^T, X_2^T, \ldots, X_k^T$ in $\Gamma(\vec{A},\vec{X})$ appear to the left of every X_1, X_2, $\ldots X_k$ variable, then the noncommutative function $\Gamma(\vec{A},\vec{X})$ is said to be **hereditary** [1] in $\vec{X}$. Our algorithm when restricted to hereditary noncommutative functions is easier to describe and the theory is easier.

[1] Note that in our definition of hereditary the variables X_j can not be constrained to be symmetric.

3.2.2 First derivatives

Conventional convexity of a function can be characterized by the second derivative being positive. As we shall see in Section 3.2.4, this is also the case with "noncommutative convex functions" and so we review a notion of second derivative which is suitable for symbolic computation. We begin with first derivatives rather than second derivatives. Later we study convexity tests which are based on derivatives of Γ and their transposes.

Directional derivatives of noncommutative rational $\Gamma(\vec{A}, \vec{X})$ with respect to $\vec{X}$ in the direction $\vec{H}$ are defined in the usual way

$$D\Gamma(\vec{X})[\vec{H}] \;:=\; \lim_{t\to 0} \frac{1}{t}\left(\Gamma(\vec{X}+t\vec{H}) - \Gamma(\vec{X})\right) = \frac{d}{dt}\,\Gamma(\vec{X}+t\vec{H})\Big|_{t=0}.$$

For example, the derivative of Γ in (3.1) with respect to X is

$$D_X\Gamma(X,Y)[H] = H^TAX + X^TAH + HYX^T + XYH^T.$$

and the derivative of Γ in (3.2) with respect to Y is

$$D_Y\Gamma(X,Y)[K] = E(KXY^T + YXK^T)E^T.$$

It is easy to check that derivatives of symmetric noncommutative rational functions always have the form

$$D\Gamma(X)[H] = sym\left[\sum_{\ell=1}^{k} A_\ell H B_\ell\right].$$

The noncommutative algebra command to generate the directional derivative of the function $\Gamma(X,Y)$ with respect to X, which is denoted by $D_X\Gamma(X,Y)[H]$, is:

NCAlgebra Command: DirectionalD[Function Γ, X, H].

3.2.3 Second derivatives

To obtain sufficient conditions for optimization we must use the second order terms of a Taylor expansion of $\Gamma(\vec{X}+t\vec{H})$ about $t=0\in\mathbb{R}$:

$$\Gamma(\vec{X}+t\vec{H}) = \Gamma(\vec{X}) + D\Gamma(\vec{X})[\vec{H}]\,t + \vec{\mathcal{H}}\Gamma(\vec{X})[\vec{H}]\,t^2 + \dots$$

Where the Hessian $\mathcal{H}\Gamma$ of Γ is defined by

$$\mathcal{H}\Gamma(\vec{X})[\vec{H}] := \frac{d^2}{dt^2}\Gamma(\vec{X}+t\vec{H})\Big|_{t=0}.$$

One can easily show that the second derivative of a hereditary symmetric noncommutative rational function Γ with respect to one variable X has the form

$$\mathcal{H}\Gamma(X)[H] = sym\left[\sum_{\ell=1}^{k} A_\ell H^T B_\ell H C_\ell\right].$$

And an analogous more general expression holds for more variables. For example, the second derivative of Γ in (3.2) with respect to X is

$$\begin{aligned}&\mathcal{H}_X\Gamma(X,Y)[H] = \\ &\quad 2\,(A(I+DXD^T)^{-1}DHD^T(I+DXD^T)^{-1}DHD^T(I+DXD^T)^{-1}A^T).\end{aligned}$$

Once the Hessian $\mathcal{H}\Gamma(\vec{X})[\vec{H}]$ is computed, the only variable of interest is $\vec{H}$. Thus, for convenience, the variables $\vec{X}$ and $\vec{A}$ are gathered in $\vec{Z}$, producing a function $\mathcal{Q}$,

$$\mathcal{Q}(\vec{Z})[\vec{H}] := \mathcal{H}\Gamma(\vec{X})[\vec{H}],$$

which is quadratic in $\vec{H}$. Here of course, a noncommutative polynomial in variables H_1, H_2, $\ldots$, H_k is said to be **quadratic** if each monomial in the polynomial expression is of order two in the variables H_1, H_2, $\ldots$, H_k.

We emphasize that for our convexity considerations once the Hessian is computed the fact that $\vec{X}$ played a special role has no influence.

NCAlgebra Command: Hessian[function Γ, $\{X_1,\ H_1\},\ldots,\{X_k,\ H_k\}$].

3.2.4 Matrix convex functions

There are several (almost equivalent) notions of noncommutative convexity, and hence we describe two familiar matrix versions. We begin by defining **matrix convex functions** as it is the definition used throughout the chapter, and later we define **geometrically matrix convex functions** as it is a common definition for convexity although we do not use it.

We shall be focusing on symmetric noncommutative functions Γ of $\vec{Z}$ defined on a domain $\mathcal{G}$ given by "inequalities" on symmetric noncommutative rational functions ρ_j, $j = 1,\ldots,r$. The tuple $\vec{Z}$ denotes all noncommutative variables $A, B, C, X, \ldots$ which appear in Γ. (Frequently we just denote $\vec{\mathcal{Z}} = \{\mathcal{Z}_1,\ldots,\mathcal{Z}_v\}$). We write the formal expression

$$\mathcal{G}_\rho := \{\vec{Z} = \{Z_1,\ldots,Z_v\} : \rho_j(\vec{Z}) \geq 0, j = 1,\ldots,r\}$$

and call such an expression a **Symbolic Inequality Domain – SID**. An example is

$$\mathcal{G} := \{\vec{Z} = \{A, C, X\} : -A^T X - XA - C^T C \geq 0, X \geq 0\}$$

Note that the $\vec{Z}$ are just formal symbols. Since our ultimate interest is matrices we introduce $\mathcal{M}(\mathcal{G}_\rho)$ the set of all matrix tuple $\vec{\mathcal{Z}} = \{\mathcal{Z}_1, \ldots, \mathcal{Z}_v\}$ which satisfy

$$\rho_j(\vec{\mathcal{Z}}) \textit{ is a positive semidefinite matrix for all } j = 1, \ldots, r.$$

Denote by $\mathcal{M}_\Delta$ all tuple of matrices $\vec{\mathcal{Z}}$ of size Δ. Denote by $\mathcal{M}_\Delta(\mathcal{G})$ the set of all matrices of size Δ which are in $\mathcal{M}(\mathcal{G})$, that is, $\mathcal{M}_\Delta(\mathcal{G}) = \mathcal{M}_\Delta \bigcap \mathcal{M}(\mathcal{G})$. See section 3.7.2 for a more complete statement.

Our main definitions of positivity are:

1. A noncommutative rational function $Q(\vec{Z})[\vec{H}]$ which is quadratic in $\vec{H}$ is said to be **matrix positive quadratic** (resp. **matrix strictly positive quadratic**) on a SID $\mathcal{G}_\rho$ provided that $Q(\vec{\mathcal{Z}})[\vec{\mathcal{H}}]$ is a positive semidefinite matrix (resp. positive definite matrix) whenever tuple of matrices $\vec{\mathcal{Z}}$ in $\mathcal{M}(\mathcal{G}_\rho)$ and $\vec{\mathcal{H}}$ are substituted for $\vec{Z}$ and $\vec{H}$.
2. The function $\Gamma(\vec{A}, \vec{X})$ is said to be **matrix convex** with respect to variable $\vec{X}$ on a SID $\mathcal{G}_\rho$ provided its Hessian $\mathcal{H}\Gamma(\vec{\mathcal{X}})[\vec{\mathcal{H}}]$ is a positive semidefinite matrix for all $\vec{\mathcal{A}}, \vec{\mathcal{X}}$ in $\mathcal{M}(\mathcal{G}_\rho)$ and all $\vec{\mathcal{H}}$; in other words, when its Hessian is matrix quadratic.

One Symbolic Inequality Domain $\mathcal{G}_\rho$ **contains another** $\mathcal{G}_{\tilde{\rho}}$, means that whenever tuple of matrices $\vec{\mathcal{Z}}$ of compatible dimension satisfy the inequalities $\tilde{\rho}_j(\vec{\mathcal{Z}}) \geq 0$, for $j = 1, \ldots, \tilde{r}$, then they also satisfy the inequalities $\rho_j(\vec{\mathcal{Z}}) \geq 0$, for $j = 1, \ldots, r$. In this case we say that

the inequalities $\rho(\vec{\mathcal{Z}}) \geq 0$ are weaker than the inequalities $\tilde{\rho}(\vec{\mathcal{Z}}) \geq 0$.

This condition is the same as $\mathcal{M}(\mathcal{G}_{\tilde{\rho}}) \subseteq \mathcal{M}(\mathcal{G}_\rho)$.

While this looks awkward and elaborate, it is in fact the type of "matrix convexity" which fits reasonably into symbolic processing of the type of matrix inequalities which engineers use. We present a few examples in Section 3.5 which make this definition clear and natural. Also matrix convexity is strongly connected with usual notions of geometric convexity, as we now discuss.

A noncommutative rational symmetric function Γ of $\vec{X} = \{X_1, \ldots, X_k\}$ will be called **geometrically matrix convex** provided that whenever the noncommutative variables $\vec{X}$ are taken to be any matrices of compatible dimension, then for all scalars $0 \leq \alpha \leq 1$ we have that

$$\Gamma(\alpha\vec{\mathcal{X}}^1 + (1-\alpha)\vec{\mathcal{X}}^2) \leq \alpha\Gamma(\vec{\mathcal{X}}^1) + (1-\alpha)\Gamma(\vec{\mathcal{X}}^2).$$

Where $\vec{\mathcal{X}}^1 = \{\mathcal{X}_1^1, \ldots, \mathcal{X}_k^1\}$ and $\vec{\mathcal{X}}^2 = \{\mathcal{X}_1^2, \ldots, \mathcal{X}_k^2\}$ are tuples of matrices of compatible dimension. The function Γ is **strictly geometrically matrix convex** if the inequality is strict for $0 < \alpha < 1$. The reverse inequality characterizes **geometrically matrix concave**.

Both the definitions, matrix convex and geometrically matrix convex, are equivalent provided that the domain of the function Γ is a convex set; as stated by the following lemma.

Lemma 3.2.1 *Suppose Γ is a noncommutative rational symmetric function. Then it is geometrically matrix convex (respectively geometrically matrix concave) on a convex region Ω of matrices of fixed sizes if and only if*

$$\mathcal{H}\Gamma(\vec{\mathcal{X}})[\vec{\mathcal{H}}] \geq 0$$

(respectively ≤ 0) for all $\vec{\mathcal{H}}$ and $\vec{\mathcal{X}} \in \Omega$.

Proof. The proof is given in Helton and Merino (1998) where Ω is all matrices of a given size. It extends in a straight forward way to Ω which are convex sets. ∎

Part 3.I

The Algorithm: Its Implementation and Use

3.3 Noncommutative Quadratic Functions

An example of a simple quadratic function in $H = H^T$ and $K = K^T$, where the arguments appear outside the expression, is

$$\mathcal{Q}[H, K] := HAH + KBK + HCK + KC^T H.$$

Or yet, a more complicated function, in the sense that the argument H appears inside the monomial is

$$\mathcal{Q}[H] := HAH + G^T HBH + HB^T HG + G^T HDHG.$$

This function can be written in the form

$$\mathcal{Q}[H] = \begin{pmatrix} H & G^T H \end{pmatrix} \begin{pmatrix} A & B^T \\ B & D \end{pmatrix} \begin{pmatrix} H \\ HG \end{pmatrix}. \tag{3.3}$$

This contrasts with the commutative case where (3.3) takes the form

$$Q[H] = H(A + G^T B + B^T G + G^T DG)H.$$

3.3.1 Representing a quadratic function as a matrix $M_{\mathcal{Q}}$

As suggested by (3.3), a noncommutative quadratic function $\mathcal{Q}$ which is hereditary in $\vec{H} = \{H_1, \ldots, H_k\}$ can be always represented as a product of the form

$$\mathcal{Q} = V[\vec{H}]^T M_{\mathcal{Q}} V[\vec{H}],$$

where $V[\vec{H}]$ is a "vector" with noncommutative entries and $M_{\mathcal{Q}}$ is a symmetric matrix with noncommutative entries. The "vector" $V[\vec{H}]$ is called a **border vector of the quadratic function** $\mathcal{Q}$ and the matrix $M_{\mathcal{Q}}$ is the **coefficient matrix of the quadratic function** $\mathcal{Q}$.

The representation $V^T M_{\mathcal{Q}} V$ for a general hereditary quadratic polynomial in $\vec{H} = \{H, K\}$ is given by $\mathcal{Q}[H, K] :=$

$$\begin{pmatrix} HL_1^1 \\ \vdots \\ HL_{\ell_1}^1 \\ KL_1^2 \\ \vdots \\ KL_{\ell_2}^2 \end{pmatrix}^T \begin{pmatrix} A_{1,1} & \cdots & A_{1,\ell_1} & A_{1,\ell_1+1} & \cdots & A_{1,r} \\ \vdots & & \vdots & \vdots & & \vdots \\ A_{1,\ell_1}^T & \cdots & A_{\ell_1,\ell_1} & A_{\ell_1,\ell_1+1} & \cdots & A_{\ell_1,r} \\ A_{1,\ell_1+1}^T & \cdots & A_{\ell_1,\ell_1+1}^T & A_{\ell_1+1,\ell_1+1} & \cdots & A_{\ell_1+1,r} \\ \vdots & & \vdots & \vdots & & \vdots \\ A_{1,r}^T & \cdots & A_{\ell_1,r}^T & A_{\ell_1+1,r}^T & \cdots & A_{r,r} \end{pmatrix} \begin{pmatrix} HL_1^1 \\ \vdots \\ HL_{\ell_1}^1 \\ KL_1^2 \\ \vdots \\ KL_{\ell_2}^2 \end{pmatrix}$$

where $r = \ell_1 + \ell_2$. The quantity ℓ_1 is the number of times that a monomial of order two in H appears, and the quantity ℓ_2 is the number of times that a monomial of order two in K appears. The L_j^i, $j = 1, \ldots, \ell_i$ are called the **coefficients of the border vector**. The L_j^1 corresponding to H are distinct and only one may be the identity matrix (equivalently for the L_j^2 corresponding to K). The border vector V is the vector composed of H, K and L_j^i. The coefficient matrix $M_{\mathcal{Q}}$ is the one in the middle with entries $A_{s,t}$, for $s, t = 1, \ldots, r$. See appendix A for an algorithm which compute this decomposition. This general notation illustrated by the example in equation (3.3) is:

$$V[H]^T = \begin{pmatrix} H & G^T H \end{pmatrix} \qquad \text{and} \qquad M_{\mathcal{Q}} = \begin{pmatrix} A & B^T \\ B & D \end{pmatrix}.$$

Noncommutative quadratics even though *not hereditary* have a similar representation (which takes much more space to write) for such a quadratic in H, K. For example, the border vector for a quadratic in H, H^T, K, K^T has the form

$$V[H,K]^T = \Big((L_1^1)^T H^T, \ldots, (L_{\ell_1}^1)^T H^T, (L_1^2)^T K^T, \ldots, (L_{\ell_2}^2)^T K^T, (\tilde{L}_1^1)^T H, \ldots, \\ (\tilde{L}_{\tilde{\ell}_1}^1)^T H, (\tilde{L}_1^2)^T K, \cdots, (\tilde{L}_{\tilde{\ell}_2}^2)^T K \Big).$$

As we shall see from the Example 3.5.3 in Section 3.5 the $M_{\mathcal{Q}}$ representation for a quadratic $\mathcal{Q}$ may not be unique. However, this non-uniqueness turns out to produce surprisingly few problems.

We should emphasize that the size of the $M_{\mathcal{Q}}$ representation of a noncommutative quadratic functions $\mathcal{Q}[H_1, \ldots, H_k]$ depends on the particular quadratic and not only on the number of arguments k of the quadratic. For example, there are noncommutative quadratic functions in one variable which have a representation with $M_{\mathcal{Q}}$ a 102×102 matrix.

NCAlgebra Command: NCMatrixOfQuadratic[$\mathcal{Q}$, $\{H_1, \ldots, H_k\}$] generates the list {left border vector, coefficient matrix, right border vector}.

3.3.2 Positivity of noncommutative quadratic functions

Determining positiveness of the Hessian, which is a quadratic function in $\vec{H}$, is the key to determining the convexity of a rational function of matrices. A critical issue is relating $Q[\vec{\mathcal{H}}]$ being a positive semidefinite matrix for all $\vec{\mathcal{H}}$ to the matrix $M_{\mathcal{Q}}$ being positive semidefinite. In this section we roughly summarize our main result which surprisingly says

that under weak hypotheses these two properties are very close to being equivalent. Later, Theorem 3.3.3 gives a definitive test for the positivity of $M_{\mathcal{Q}}$.

Theorem 3.3.1 (Positivity: $\mathcal{Q}$ *versus* $M_{\mathcal{Q}}$) *Suppose that the noncommutative rational function $\mathcal{Q}(\vec{Z})[\vec{H}]$ is quadratic in $\vec{H}$. Represent $\mathcal{Q}(\vec{Z})$ with coefficient matrix $M_{\mathcal{Q}(\vec{Z})}$ and border vector $V[\vec{H}]$, that is $\mathcal{Q}(\vec{Z})[\vec{H}] = V[\vec{H}]^T M_{\mathcal{Q}(\vec{Z})} V[\vec{H}]$. Let $\mathcal{G}$ denote the Symbolic Inequality Domain, based on $M_{\mathcal{Q}(\vec{Z})}$, given by*

$$\mathcal{G} := \left\{ \vec{Z} : M_{\mathcal{Q}(\vec{Z})} \geq 0 \right\}.$$

Then $\mathcal{Q}(\vec{Z})[\vec{H}]$ is a matrix positive quadratic function for each $\vec{Z} \in \mathcal{G}$. Conversely, assume:

i. the $M_{\mathcal{Q}}$ representation of $\mathcal{Q}$ has a border vector $V[\vec{H}]$ with coefficients $L_1^j(\vec{Z})$, ..., $L_{l_j}^j(\vec{Z})$ for H_j which for each j are linearly independent functions of $\vec{Z}$;

ii. the Symbolic Inequality Domain $\mathcal{G}$ is not thin in the sense that the set $\mathcal{M}_\Delta(\mathcal{G})$ is an open set in $\mathcal{M}_\Delta$, provided that the size Δ is large enough (see the Openness Property in Section 3.7.2).

Then the closure of $\mathcal{G}$ in a certain topology is the biggest domain on which $\mathcal{Q}(\vec{Z})[\vec{H}]$ is a matrix positive quadratic function.

Proof. The sufficient side, the symmetric matrix $M_{\mathcal{Q}}$ being positive semidefinite guarantees that the matrix $\mathcal{Q}[\vec{\mathcal{H}}]$ is also positive semidefinite for all tuple of matrices $\vec{\mathcal{H}}$, is trivially proved. To see this, write the quadratic function as

$$\mathcal{Q}[\mathcal{H}_1, \ldots, \mathcal{H}_k] := V[\mathcal{H}_1, \ldots, \mathcal{H}_k]^T M_{\mathcal{Q}} V[\mathcal{H}_1, \ldots, \mathcal{H}_k].$$

Now, let $M_{\mathcal{Q}} \in \mathbb{R}^{r \times r}$ be positive semidefinite. By definition this implies that

$$x^T M_{\mathcal{Q}} x \geq 0 \quad \text{for all vectors} \quad x \in \mathbb{R}^r.$$

So, for any $y \in \mathbb{R}^m$, choose x to be $x = V[\mathcal{H}_1, \ldots, \mathcal{H}_k]y$. Then

$$x^T M_{\mathcal{Q}} x = y^T V[\mathcal{H}_1, \ldots, \mathcal{H}_k]^T M_{\mathcal{Q}} V[\mathcal{H}_1, \ldots, \mathcal{H}_k] y = y^T \mathcal{Q}[\mathcal{H}_1, \ldots, \mathcal{H}_k] y \geq 0.$$

The necessity side requires involved proof which takes up Part II of this chapter. We shall illustrate one of its steps in the simple Example 3.3.1 below. ∎

Note that linear dependence of a small set of matrices in a high dimensional space is a rare event. This intuitively speaking is the type of linear dependence in assumption (i) of Theorem 3.3.1 required to violate the necessity of $M_{\mathcal{Q}}$ being positive. Indeed, this type of linear dependence has never occurred in any experiments we have done, although one could probably make up examples where it occurs.

Even though a quadratic $\mathcal{Q}$ can have two representations $M_{\mathcal{Q}}^1$ and $M_{\mathcal{Q}}^2$ meeting the hypotheses in Theorem 3.3.1, our result implies that $M_{\mathcal{Q}}^1$ will be positive semidefinite if and only if $M_{\mathcal{Q}}^2$ is also positive semidefinite.

Example 3.3.1 Consider the noncommutative quadratic function $\mathcal{Q}[H]$ given by

$$\mathcal{Q}[H] := H^T B H + G^T H^T C H + H^T C^T H G + G^T H^T A H G. \tag{3.4}$$

Here, in distinction to most of Part I, we are not forcing H to be symmetric. This is much easier to analyze than the case where H is symmetric. The border vector $V[H]$ and the coefficient matrix $M_{\mathcal{Q}}$ with noncommutative entries are

$$V[H]^T = \begin{pmatrix} H^T & G^T H^T \end{pmatrix} \qquad \text{and} \qquad M_{\mathcal{Q}} = \begin{pmatrix} B & C^T \\ C & A \end{pmatrix},$$

that is, $\mathcal{Q}[H]$ has the form

$$\mathcal{Q}[H] = V[H]^T M_{\mathcal{Q}} V[H] = \begin{pmatrix} H^T & G^T H^T \end{pmatrix} \begin{pmatrix} B & C^T \\ C & A \end{pmatrix} \begin{pmatrix} H \\ HG \end{pmatrix}.$$

Now, if in equation (3.4) the elements A, B, C, G, H are replaced by matrices in $\mathbb{R}^{n\times n}$, then the noncommutative quadratic function $\mathcal{Q}[H]$ becomes a matrix valued function $\mathcal{Q}[\mathcal{H}]$. The matrix valued function $\mathcal{Q}[\mathcal{H}]$ is positive semidefinite if and only if $x^T\mathcal{Q}[\mathcal{H}]x \geq 0$ for all vectors $x \in \mathbb{R}^n$ and all $\mathcal{H} \in \mathbb{R}^{n\times n}$. Or equivalently, the following inequality must hold

$$\begin{pmatrix} x^T\mathcal{H}^T & x^T\mathcal{G}^T\mathcal{H}^T \end{pmatrix} M_{\mathcal{Q}} \begin{pmatrix} \mathcal{H}x \\ \mathcal{H}\mathcal{G}x \end{pmatrix} \geq 0. \tag{3.5}$$

Let

$$y^T := \begin{pmatrix} x^T\mathcal{H}^T & x^T\mathcal{G}^T\mathcal{H}^T \end{pmatrix}. \tag{3.6}$$

Then (3.5) is equivalent to $y^T M_{\mathcal{Q}}\, y \geq 0$. Now it suffices to prove that all vectors of the form y span $\mathbb{R}^{2n}$.

Suppose for a given x, with $n \geq 2$, the vectors x and $\mathcal{G}x$ are linearly independent. Let $y = \begin{pmatrix} v_1 \\ v_2 \end{pmatrix}$ be any vector in $\mathbb{R}^{2n}$, then we can choose $\mathcal{H} \in \mathbb{R}^{n \times n}$ with the property that $v_1 = \mathcal{H}x$ and $v_2 = \mathcal{H}\mathcal{G}x$. It is clear that vectors of the form

$$\mathcal{R}^x := \left\{ \begin{pmatrix} \mathcal{H}x \\ \mathcal{H}\mathcal{G}x \end{pmatrix} : \text{ for all } \mathcal{H} \right\}$$

is all $\mathbb{R}^{2n}$ as required. Thus we are finished unless for all x the vectors x and $\mathcal{G}x$ are linearly dependent. That is for all x, $\lambda_1(x)x + \lambda_2(x)\mathcal{G}x = 0$ for nonzero $\lambda_1(x)$ and $\lambda_2(x)$. Note $\lambda_2(x) \neq 0$, unless $x = 0$. Set $\tau(x) := \frac{\lambda_1(x)}{\lambda_2(x)}$, then the linear dependence becomes $\tau(x)x + \mathcal{G}x = 0$. This says that every vector x is an eigenvector of $\mathcal{G}$, which implies that $\mathcal{G} = \lambda I$ for some constant λ. This fact can be verified from the Jordan form $\mathcal{G} = \mathcal{M}^{-1}\mathcal{J}\mathcal{M}$ via $\tau(x)\mathcal{M}x + \mathcal{J}\mathcal{M}x = 0$, for all x. Thus the set of all y satisfying (3.6) is all of $\mathbb{R}^{2n}$ unless $\tau I + \mathcal{G} = 0$ for some τ.

Conversely, if $\mathcal{G} = \lambda I$, then the set of y of the form (3.6) is not all of $\mathbb{R}^{2n}$ and has an orthogonal complement $\mathbb{R}^{\perp}$. The function $\mathcal{Q}$ can be positive without $r^T M_{\mathcal{Q}} r$ being positive on vectors $r \in \mathbb{R}^{\perp}$.

Clearly the method used in the proof above to show that $\mathcal{R}^x$ is all of $\mathbb{R}^{2n}$ is very special. Part II of this chapter uses a very different method (there are several parts to this more general proof). In a very vague sense, the main idea behind the proof is that if $\mathcal{R}^x$ is not all of $\mathbb{R}^{2n}$, then the coefficients L_j^i of the border vector form a set of linearly dependent functions. One consequence of this linear dependence property, which is of independent interest, is presented in the following corollary of Theorem 3.10.10 from Part II.

Corollary 3.3.2 (Corollary 3.10.11) *Let $L_1(\vec{Z}), \ldots, L_\ell(\vec{Z})$ be noncommutative rational functions of $\vec{Z} = \{Z_1, \ldots, Z_v\}$. For each vector x, suppose that the vectors $L_1(\vec{\mathcal{Z}})x, \ldots, L_\ell(\vec{\mathcal{Z}})x$ are linearly dependent whenever matrices $\mathcal{Z}_j$ of compatible dimension are substituted for Z_j for all size Δ bigger than some Δ_0. Then there exist real numbers λ_j for $j = 1, \ldots, \ell$ such that*

$$\sum_{j=1}^{\ell} \lambda_j L_j(\vec{Z}) = 0,$$

that is, the functions $L_j(\vec{Z})$ are linearly dependent.

We mention some basic work on positivity of commutative polynomials (not just quadratic polynomials) done in Parrilo (2000); Powers and Wörmann (1998). Our algorithm is

somewhat like theirs, in that both use the LDL$^{\mathrm{T}}$ decomposition. While positivity of commutative quadratic functions is easily checked, noncommutative quadratics cause difficulties reminiscent of what happens with non-quadratic higher order commutative polynomials.

3.3.3 Noncommutative LDU decomposition

In our approach, the LDU factorization of a matrix with noncommutative entries is the key tool for determination of the matrix positivity of a quadratic function, and hence the region of convexity $\mathcal{G}$ of noncommutative functions.

The LDU factorization applied to a symmetric matrix M of size $r \times r$ with noncommutative entries provides the decomposition $M = LDL^T$, where the $r \times r$ matrix D is diagonal[2] or contains 2×2 blocks with zeros on the diagonal, and the $r \times r$ matrix L is lower triangular and normalized so that each diagonal entry equals the identity. To check the positivity of the symmetric matrix M it suffices to check that D is purely diagonal and to check the positivity of the entries of the diagonal matrix D. It is often very useful (sometimes essential) to perform the LDU decomposition not on a given matrix M but on a matrix PMQ obtained from M by permutation matrices P, Q. When M is symmetric, we shall choose $Q = P^T$ so as to obtain $PMP^T = LDL^T$, or equivalently $M = P^T LDL^T P$.

References on LDU decomposition of matrices with commutative entries are Golub and Loan (1983); Horn and Johnson (1996). The LDU decomposition for noncommutative 2×2 matrices is standard and appears in many places. We do not know a reference on the general $r \times r$ case. However, as we shall see its properties are much like the well understood commuting case. Note that at the k^{th} $\{k := 0, \ldots, r-2\}$ step of the process above, one can choose $(r-k)$-factorial permutations. The noncommutative LDL$^{\mathrm{T}}$ decomposition (as implemented in NCAlgebra) is briefly presented here.

Let a symmetric 2×2 matrix with noncommutative entries be given by

$$M = \begin{pmatrix} A & B^T \\ B & C \end{pmatrix}$$

with A and C symmetric elements. Then M has the following LDL$^{\mathrm{T}}$ decomposition

$$M = \mathrm{LDL}^{\mathrm{T}} = \begin{pmatrix} I & 0 \\ BA^{-1} & I \end{pmatrix} \begin{pmatrix} A & 0 \\ 0 & C - BA^{-1}B^T \end{pmatrix} \begin{pmatrix} I & A^{-1}B^T \\ 0 & I \end{pmatrix}, \tag{3.7}$$

[2]This assumes that at each step of our LDU algorithm a matrix entry called **pivot** is invertible. The case where some pivot may not be invertible will be discussed in details in Theorem 3.3.3.

provided that the noncommutative element A is invertible. Our computer algorithm automatically assumes invertibility when it is needed. If the permutation

$$P = \begin{pmatrix} 0 & I \\ I & 0 \end{pmatrix}$$

is applied to both sides of M producing

$$PMP^T = \begin{pmatrix} C & B \\ B^T & A \end{pmatrix},$$

the decomposition is

$$PMP^T = \mathrm{LDL^T} = \begin{pmatrix} I & 0 \\ B^TC^{-1} & I \end{pmatrix} \begin{pmatrix} C & 0 \\ 0 & A - B^TC^{-1}B \end{pmatrix} \begin{pmatrix} I & C^{-1}B \\ 0 & I \end{pmatrix}. \tag{3.8}$$

Note that matrix D in the two decompositions (3.7) and (3.8) above has the classical Schur complements as its main ingredients.

Now we sketch the computer algebra algorithm for noncommutative symmetric matrices of size $r \times r$. Suppose that matrix M has $r \times r$ noncommutative entries. Then M can be always partitioned as

$$M = \begin{pmatrix} A_{11} & \mathbf{B}^T \\ \mathbf{B} & \mathbf{C} \end{pmatrix} \tag{3.9}$$

with $\mathbf{C}$ a matrix of size $(r-1)\times(r-1)$ and B a matrix of size $(r-1)\times 1$ with noncommutative entries. Now apply the 2×2 $\mathrm{LDL^T}$ decomposition as in (3.7) to get

$$\begin{pmatrix} I & 0 \\ \mathbf{B}A_{11}^{-1} & I \end{pmatrix} \begin{pmatrix} A_{11} & 0 \\ 0 & \mathbf{C} - \mathbf{B}A_{11}^{-1}\mathbf{B}^T \end{pmatrix} \begin{pmatrix} I & A_{11}^{-1}\mathbf{B}^T \\ 0 & I \end{pmatrix}.$$

In our symbolic algorithm we assume that if A_{11} is not 0, then it has an inverse denoted A_{11}^{-1}. We call A_{11} the **pivot** for this step of the algorithm. At the next step the $r-1 \times r-1$ matrix $\mathbf{C} - \mathbf{B}A_{11}^{-1}\mathbf{B}^T$ with noncommutative entries, called the **residual matrix**, can also be factored as $\hat{L}\hat{D}\hat{L}^T$ using a partition form analogous to (3.9). In that case M takes the form

$$M = \begin{pmatrix} I & 0 \\ \mathbf{B}A_{11}^{-1} & \hat{L} \end{pmatrix} \begin{pmatrix} A_{11} & 0 \\ 0 & \hat{D} \end{pmatrix} \begin{pmatrix} I & A_{11}^{-1}\mathbf{B}^T \\ 0 & \hat{L}^T \end{pmatrix}.$$

The procedure continues until the residual matrix has size 1×1 (in which case we are finished) or the diagonal entry on which we need to pivot is 0. In the later case we find a non-zero diagonal entry A_{kk} and apply a permutation P from right and left to move this

diagonal entry A_{kk} to the pivot[3] position. Then we proceed as before. This procedure with permutations stops when the residual matrix R has size 1×1 or all diagonal entries of the residual matrix R of size greater than 1 are identically zero (and no pivot is possible).

The key property of the Noncommutative LDU Algorithm is

Theorem 3.3.3 *Suppose M is a symmetric matrix of size $r \times r$ with noncommutative rational function entries. The possibly permuted LDU algorithm outputs a matrix D with noncommutative rational entries.* **Either** D *is* **diagonal**,

i. in which case, whenever $n \times n$ matrices are substituted for the variables in the function D_j, $j = 1, \ldots, r$ in D and produce matrices $\mathcal{D}_j$, which for $j = 1, \ldots, r-1$ are invertible, then

> *each $\mathcal{D}_j$ for $j = 1, \ldots, r$ is a positive definite (resp. positive semidefinite) matrix if and only if the $rn \times rn$ matrix $\mathcal{M}$ resulting from M is positive definite (resp. positive semidefinite).*

or D *can be partitioned as* $D = \mathrm{diag}(\widetilde{D}, R)$,[4] *where $\widetilde{D}$ is a diagonal matrix with noncommutative rational entries $\widetilde{D}_j$, $j = 1, \ldots, d$ with $d < r - 1$, and R is a* **non-diagonal** *matrix of size $(r - d) \times (r - d)$. We need to distinguish two situations:*

ii. **All entries of the matrix R are identically zero**, *in which case D is actually diagonal, and the conclusion of case (i) applies.*

iii. **The off diagonal entries of R are not identically zero**, *in which case some matrices substituted for the variables in M produce $\mathcal{M}$ which is neither a positive semidefinite matrix nor a negative semidefinite matrix.*

Proof. Prove (i,ii): Suppose D is diagonal with entries $D_j, j = 1, \ldots, d$ not identically zero. Our symbolic algorithm used an expression denoting the inverse of each pivot. Note that the pivots used in the algorithm (and assumed invertible) are exactly the diagonal elements D_j, for $j = 1, \ldots, \min(d, r - 1)$. Thus our symbolic formulas are valid when matrices are substituted in, provided that the resulting matrix diagonal entries $\mathcal{D}_j$ for $j = 1, \ldots, \min(d, r-1)$, are invertible. Thus $\mathcal{D}_j$, for $j = 1, \ldots, d$, positive semidefinite (resp.

[3] A appealing way to choose A_{kk} is to observe that each diagonal entry typically will be a rational function of other entries in the matrix. Thus each A_{jj} is given by a formula of some length, and we select A_{kk} to be the nonzero diagonal entry of shortest length. This is a symbolic analog of the common numerical analysis method of picking the pivot of largest size.

[4] $\mathrm{diag}(x_1, \ldots, x_r)$ *means a diagonal matrix with entries* $x_1, \ldots, x_r$.

for $j = 1, \ldots, r$, each $\mathcal{D}_j$ a positive definite matrix) implies that $\mathcal{M}$ is positive semidefinite (resp. positive definite). Conversely, if $\mathcal{M}$ is positive semidefinite (resp. positive definite) the $\mathcal{D}_j$, for $j = 1, \ldots, d$, are positive semidefinite (resp. for $j = 1, \ldots, r$, each $\mathcal{D}_j$ is positive definite) since $\mathcal{L}$ is invertible.

Now we prove (iii): If $n \times n$ matrices of any size n are substituted for the variables in M and in R the resulting symmetric residual matrix $\mathcal{R}$ has block diagonal entries equal to the $n \times n$ zero matrix, which implies that $\mathcal{R}$ has trace 0, which implies $\mathcal{R}$ has some positive and some negative eigenvalues. Thus $\mathcal{R}$ and consequently $\mathcal{M}$ can not be either a positive semidefinite matrix or a negative semidefinite matrix. ∎

While we have presented only enough of the $\mathrm{LDL}^{\mathrm{T}}$ decomposition for noncommutative symmetric matrices to determine positivity, in fact the NCAlgebra program can do more. If the user chooses a certain option, NCAlgebra picks a non zero 2×2 block in R and pivots on it. This procedure combined with permutations when needed, ultimately produces a center matrix D which is block diagonal with blocks of size 1×1 or 2×2. This exactly generalizes the standard behavior of the commutative case.

A further feature of our NCAlgebra implementation is that one can retrieve the sequence of permutations which the algorithm selected. Also one can specify exactly which permutations are to be used and thereby override the algorithm's automatic selection of permutations.

A brief summary of a simplified version of the $\mathrm{LDL}^{\mathrm{T}}$ algorithm code implemented in the NCAlgebra package follows.

Algorithm 3.3.4 (Noncommutative $\mathrm{LDL}^{\mathrm{T}}$ Decomposition)

Set $k = 0$, $M_k = M$
while $k < r$ **do**
    Apply desired permutation on M_k
    Partition M_k *as in (3.9)*
    $L_k D_k L_k^T \leftarrow M_k$; *as in (3.7)*
    Append: $L \leftarrow L_k$; $D \leftarrow D_k$
    Let M_k *be the residual* $C_k - B_k A_k^{-1} B_k^T$
    $k \leftarrow k + 1$
end

NCAlgebra Command: NCLDUDecomposition[M], gives a permuted LDU decomposition of a symmetric M.

3.4 Convexity Algorithm

This section presents our main algorithm that provides the region $\mathcal{G}$ in which a given noncommutative symmetric function $\Gamma(\vec{Z})$ is matrix convex in $\vec{X}$.

1. Compute symbolically $\mathcal{Q}(\vec{Z})[\vec{H}] := \mathcal{H}\Gamma(\vec{X})[\vec{H}]$.

2. As $\mathcal{Q}(\vec{Z})[\vec{H}]$ is second order in $\vec{H}$, it can be expressed as $V[\vec{H}]^T M_{\mathcal{Q}(\vec{Z})} V[\vec{H}]$. Extract the matrix $M_{\mathcal{Q}(\vec{Z})}$ from this quadratic expression.

3. Apply the noncommutative LDL$^{\mathrm{T}}$ decomposition on the matrix $M_{\mathcal{Q}(\vec{Z})}$, i.e., $M_{\mathcal{Q}(\vec{Z})} = LDL^T$, to get matrix D with noncommutative entries.

4. Suppose that matrix D can be partitioned as $D = \mathrm{diag}(\widetilde{D}, R)$, where $\widetilde{D}$ is a diagonal matrix with entries $\rho_j(\vec{Z})$, for $j = 1, \ldots, \tilde{d}$ and R is a non-diagonal matrix of size $(r - \tilde{d}) \times (r - \tilde{d})$ containing zeros on the diagonal or 2×2 blocks

$$R_i = \begin{pmatrix} 0 & \rho_i(\vec{Z}) \\ \rho_i(\vec{Z})^T & 0 \end{pmatrix}$$

for $i = \tilde{d} + 1, \ldots, r$. Thus matrix D has the form

$$D = \begin{pmatrix} \rho_1(\vec{Z}) & & & & & & & \\ & \ddots & & & & & & \\ & & \rho_{\tilde{d}}(\vec{Z}) & & & & & \\ & & & 0 & \rho_{\tilde{d}+1}(\vec{Z}) & & & \\ & & & \rho_{\tilde{d}+1}(\vec{Z})^T & 0 & & & \\ & & & & \ddots & & & \\ & & & & & 0 & \rho_r(\vec{Z}) & \\ & & & & & \rho_r(\vec{Z})^T & 0 & \\ & & & & & & & 0 \end{pmatrix}.$$

5. The Hessian $\mathcal{Q}(\vec{\mathcal{Z}})[\vec{\mathcal{H}}]$ is a positive semidefinite matrix for all $\vec{\mathcal{H}}$ whenever the tuple of matrices $\vec{\mathcal{Z}} = \{\mathcal{Z}_1, \ldots, \mathcal{Z}_v\}$ makes the block diagonal matrix D positive semidefinite.

Thus a set $\mathcal{G}$ where $\Gamma(\vec{Z})$ is matrix convex is given by

$$\mathcal{G} = \left\{ \vec{Z} : \rho_j(\vec{Z}) > 0,\ j = 1, \ldots, \tilde{d} \right\} \bigcap \left\{ \vec{Z} : \rho_i(\vec{Z}) = 0,\ i = \tilde{d}+1, \ldots, r \right\}.$$

6. Note that, if $M_{\mathcal{Q}(\vec{Z})}$ is a matrix of size $r \times r$, then there are $\Pi = r!(r-1)! \cdots 2$ possible $\mathrm{LDL}^{\mathrm{T}}$ decompositions depending on different permutations of the matrix $M_{\mathcal{Q}(\vec{Z})}$. This gives Π different diagonal matrices, D^1, D^2, ..., D^{Π}. Up to the assumptions that the NCLDUDecomposition algorithm makes about invertibility, each D^i must produce a set $\mathcal{G}$. However, the inequalities produced by the diagonal D^i may be much more elegant and useful than those produced by the diagonal D^j, even though they must produce equivalent sets $\mathcal{G}$.

The main difficulty is the fact that there are Π different permutations for doing the $\mathrm{LDL}^{\mathrm{T}}$ decomposition. Checking them all consumes computer time and leaves the user with many choices. In our experience many permutations work to give the same answer (as will be shown in some examples), so finding a satisfactory one appears not to be time consuming.

The set $\mathcal{G}$ produced by the Convexity Algorithm is the biggest possible in a certain sense. This is the content of Theorem 3.3.1 and is described precisely in Theorem 3.8.2 of Part II.

3.5 Examples

In this section we give several examples of the Convexity Algorithm which vary in complication and which illustrate different points. We begin with a simple example.

Example 3.5.1 Define the function $\Gamma(X)$ by

$$\Gamma(X) = G^T X^T A X G + X^T B X + G^T X^T C X + X^T C^T X G,$$

where $B = B^T$ and $A = A^T$. The Hessian of $\Gamma(X)$ is given by

$$\mathcal{H}\Gamma(X)[H] = 2(H^T B H + H^T C^T H G + G^T H^T A H G + G^T H^T C H).$$

Equivalently, this quadratic expression takes the form

$$\mathcal{H}\Gamma(X)[H] = V[H]^T M_{\mathcal{H}\Gamma} V[H] = 2(H^T, G^T H^T) \begin{pmatrix} B & C^T \\ C & A \end{pmatrix} \begin{pmatrix} H \\ HG \end{pmatrix}.$$

The LDL$^{\mathrm{T}}$ decomposition with no permutation applied to $M_{\mathcal{H}\Gamma}$ is

$$\begin{pmatrix} I & 0 \\ CB^{-1} & I \end{pmatrix} \begin{pmatrix} B & 0 \\ 0 & A - CB^{-1}C^T \end{pmatrix} \begin{pmatrix} I & B^{-1}C^T \\ 0 & I \end{pmatrix},$$

provided that B is invertible[5]

Therefore, when B is invertible and $G \neq \alpha I$, for any scalar α, the necessary and sufficient conditions for the Hessian to be positive semidefinite are

$$B > 0 \quad \text{and} \quad A - CB^{-1}C^T \geq 0.$$

On the other hand, if A is invertible and a permutation is applied, the LDL$^{\mathrm{T}}$ decomposition is

$$\begin{pmatrix} I & 0 \\ C^T A^{-1} & I \end{pmatrix} \begin{pmatrix} A & 0 \\ 0 & B - C^T A^{-1} C \end{pmatrix} \begin{pmatrix} I & A^{-1}C \\ 0 & I \end{pmatrix}.$$

For this case, the necessary and sufficient conditions are

$$A > 0 \quad \text{and} \quad B - C^T A^{-1} C \geq 0.$$

3.5.1 NCAlgebra examples

Henceforth our examples will use notation which is standard in Mathematica and NCAlgebra. This adds a level of precision and concreteness to the discussion. Also the notation is quite transparent so it causes little reading difficulty. Sometimes for better visualization, TEX notation is employed. In the course of illustrating the Convexity Algorithm we actually show what is inside the command **NCConvexityRegion**[ ].

Before going through the examples, it is convenient to explain the basic notation used in NCAlgebra. The transpose of an element x is denoted by tp$[x]$. The identity is denoted 1. The inverse of x is inv$[x]$. The product of the noncommutative elements x and y is $x**y$. The product of a matrix A with noncommutative entries by another matrix B is provided by the command MatMult$[A, B]$.

The directional derivative, the Hessian, and the LDU decomposition, were already introduced. They are provided from:

- Hessian$[f(X,Y), \{X,H\}, \{Y,K\}]$,

[5]The list returned by **NCConvexityRegion** is $\{B, \quad A - CB^{-1}C^T\}$.

- DirectionalD[$\Gamma(X,Y)$, $\{X,\ H\}$, $\{Y,\ K\}$],
- NCLDUDecomposition[$Matrix$].

The border vector and the coefficient matrix of a noncommutative quadratic function is given by

NCMatrixOfQuadratic[$\mathcal{Q}$, $\{H,\ K\}$].

The command NCExpand[*expression*] expands out noncommutative multiply's inside an algebraic expression. It is the noncommutative generalization of the Mathematica Expand[ ].

The command NCSimplifyRational[ ], simplifies an expression that includes polynomials and inverses of polynomials. This works by applying a collection of simplifying rules to the expression. The call is

NCSimplifyRational[*expression*]

This is in practice an essential command because the expressions obtained by other commands, such as NCLDUDecomposition[ ], Hessian[ ], etc., usually are not in their simplified form. For more details about simplification of noncommutative expressions and symbolic implementation, the reader is referred to Helton et al. (1998).

The following examples describe the steps for checking the convexity of a noncommutative function.

Example 3.5.2 Let the function Γ be given by

$$F := XA + A^TX - (C1^T - X\ B\ D1^T)(Y - D1\ D1^T)^{-1}(C1 - D1\ B^TX) - XBB^TX$$

with $X = X^T$ and $Y = Y^T$. The definition of this function F in Mathematica is:

```
In[6]:= F := X**A+tp[A]**X - X**B**tp[B]**X
        - (tp[C1]-X**B**tp[D1])**inv[Y-D1**tp[D1]] ** (C1-D1**tp[B]**X);
```

The Hessian of this function is produced by the command

```
In[7]:= hess = 1/2 NCHessian[F, {X, H}, {Y, K}] // NCSimplifyRational;
```

The left (right) border vector and the coefficient matrix Mhess are produced by the command

In[8]:= {LeftBorder, Mhess, RightBorder} = **NCMatrixOfQuadratic**[hess, H, K];

The matrix Mhess from the command above in TEX format is

$$\text{Mhess} = \begin{pmatrix} -BB^T - BD1^T RD1B^T & -BD1^T R & BD1^T R \\ -RD1B^T & -R & R \\ RD1B^T & R & -R \end{pmatrix},$$

where we have made the substitution $R := (Y - D1D1^T)^{-1}$. The LDL$^{\mathrm{T}}$ decomposition of Mhess is obtained by the command

In[9]:= {lu, di, up, P} = **NCLDUDecomposition**[Mhess] // **NCSimplifyRational**;

From the output of this command we obtain the diagonal matrix di, presented below in TEX format

$$\text{di} = \begin{pmatrix} -(Y - D1D1^T)^{-1} & 0 & 0 \\ 0 & -BB^T & 0 \\ 0 & 0 & 0 \end{pmatrix}.$$

The list returned by **NCConvexityRegion** is the entries of the diagonal matrix di:

$$\{-(Y - D1D1^T)^{-1},\ -BB^T,\ 0\}.$$

Therefore we may conclude that the function F is concave on the region $\mathcal{G} := \{Y : Y - D1D1^T > 0\}$.

To determine that closure($\mathcal{G}$) $:= \{Y : Y - D1D1^T \geq 0\}$ is the biggest domain of concavity we need to check if the border vector is linearly independent and if the region $\mathcal{G}$ satisfies the Openness Property[6]. The left border vector "LeftBorder" is

$$\text{LeftBorder} = \{H,\ C1^T(Y - D1D1^T)^{-1}K,\ XBD1^T(Y - D1D1^T)^{-1}K\}.$$

This border vector has linearly independent[7] coefficients for each H and K. To see that, we need to analyze separately the coefficients for the H and K. The H case is trivial as it appears only once. For the K, we need to show that the functions $L_1(Y) := C1^T(Y -$

[6] See the Openness Property in Section 3.7.2 referred to in item (*ii*) of Theorem 3.3.1

[7] A rigorous treatment is given in Definition 3.7.1, where the block linearly independence property is defined.

$D1D1^T)^{-1}$ and $L_2(X,Y) := XBD1^T(Y - D1D1^T)^{-1}$ are linearly independent, which is immediate as L_1 does not depend on X. We remark that the output of the LeftBorder is an option in NCConvexityRegion. Also a sufficient though not necessary test for linear independence of the LeftBorder vector entries is automatically implemented. This test is sketch later in Example 3.5.3.

It is also evident from the strict inequality that for matrices of any compatible dimension the domain $\mathcal{M}(\mathcal{G})$ of matrices is an open set; thus $\mathcal{G}$ satisfy the Openness Property. Therefore we conclude the region closure$(\mathcal{G}) := \{Y : Y - D1D1^T \geq 0\}$ is the biggest domain of concavity for the function F.

An interesting aspect of the next example is that it shows that the $M_{\mathcal{Q}}$ representation may not be unique. This may lead one to conclude that a function is matrix positive instead of being matrix strictly positive.

Example 3.5.3 Let x, y, h and k be symmetric noncommutative elements. Let's define the noncommutative function $F(x,y)$ to be used in the example as

$$F(x,y) := (x - y^{-1})^{-1}.$$

This function F in Mathematica takes the form:

In[10]:= F := inv[x - inv[y]];

Thus, the Hessian $\mathcal{H}\Gamma(x,y)[h,k]$ of this function is produced by the command

In[11]:= hess = 1/2 **NCHessian**[F, {x, h}, {y, k}] // **NCExpand**

```
inv[x - inv[y]] ** h ** inv[x - inv[y]] ** h ** inv[x - inv[y]] + inv[x - inv[y]] **
h ** inv[x - inv[y]] ** inv[y] ** k ** inv[y] ** inv[x - inv[y]] + inv[x - inv[y]] **
inv[y] ** k ** inv[y] ** k ** inv[y] ** inv[x - inv[y]] + inv[x - inv[y]] ** inv[y] **
k ** inv[y] ** inv[x - inv[y]] ** h ** inv[x - inv[y]] + inv[x - inv[y]] ** inv[y] **
k ** inv[y] ** inv[x - inv[y]] ** inv[y] ** k ** inv[y] ** inv[x - inv[y]]
```

The left (right) border vector and the coefficient matrix Mhess are produced by the command

In[12]:= {LeftBorder, Mhess, RightBorder} = **NCMatrixOfQuadratic**[hess, {h, k}];

The Hessian of F, denoted by hess, can be rewritten in TEX format as

$$\text{hess } = V^T \text{Mhess } V,$$

where V^T = LeftBorder is given by

$$V^T = \begin{bmatrix} h(x-y^{-1})^{-1} \\ ky^{-1}(x-y^{-1})^{-1} \end{bmatrix}^T$$

and the matrix Mhess is given by

$$\text{Mhess } = \begin{bmatrix} (x-y^{-1})^{-1} & (x-y^{-1})^{-1}y^{-1} \\ y^{-1}(x-y^{-1})^{-1} & y^{-1}+y^{-1}(x-y^{-1})^{-1}y^{-1} \end{bmatrix}.$$

The LDL$^{\mathrm{T}}$ decomposition of the coefficient matrix Mhess is given by the command

In[13]:= {lu, di, up, P} = **NCLDUDecomposition**[Mhess] // **NCSimplifyRational**;

From the output of this command we obtain the following factorization for P Mhess P^T = lu di up

$$\begin{pmatrix} I & 0 \\ y^{-1} & I \end{pmatrix} \begin{pmatrix} (x-y^{-1})^{-1} & 0 \\ 0 & y^{-1} \end{pmatrix} \begin{pmatrix} I & y^{-1} \\ 0 & I \end{pmatrix},$$

where P is a permutation matrix generated automatically by our LDU algorithm. Finally, the list returned by **NCConvexityRegion** is the entries of the diagonal matrix di, i.e.,

$$\{(x-y^{-1})^{-1}, \quad y^{-1}\}.$$

Therefore the Hessian is matrix strictly positive on the Symbolic Inequality Domain

$$\mathcal{G} := \{(x,y) : y > 0 \quad \text{and} \quad x - y^{-1} > 0\}. \tag{3.10}$$

Now, Let's analyze the effect of a different representation for the Hessian. Where instead of expanding the expression for the Hessian with the command NCExpand, we apply the command NCSimplifyRational.

In[14]:= hess = 1/2 **NCHessian**[F, {x, h}, {y, k}] // **NCSimplifyRational**

k ** h ** inv[x - inv[y]]+inv[x - inv[y]] ** h ** k - k ** x ** inv[x - inv[y]] ** h ** inv[x - inv[y]]+k ** x ** inv[x - inv[y]] ** inv[y] ** k - inv[x - inv[y]] ** h ** k ** x ** inv[x - inv[y]]+inv[x - inv[y]] ** h ** inv[x - inv[y]] ** h ** inv[x - inv[y]] - inv[x - inv[y]] ** h ** inv[x - inv[y]] ** x ** k - inv[x - inv[y]] ** x ** k ** h ** inv[x - inv[y]] - k ** x ** inv[x - inv[y]] ** inv[y] ** k ** x ** inv[x - inv[y]]+inv[x - inv[y]] ** h ** inv[x - inv[y]] ** x ** k ** x ** inv[x - inv[y]]+inv[x - inv[y]] ** x ** k ** x ** inv[x - inv[y]] ** h ** inv[x - inv[y]] - inv[x - inv[y]] ** x ** k ** x ** inv[x - inv[y]] ** inv[y] ** k+ inv[x - inv[y]] ** x ** k ** x ** inv[x - inv[y]] ** inv[y] ** k ** x ** inv[x - inv[y]]

The LeftBorder (RightBorder) vector and the coefficient matrix Mhess are produced by the command

In[15]:= {LeftBorder, Mhess, RightBorder} = **NCMatrixOfQuadratic**[hess, {h, k}];

The Hessian of F can be rewritten in TEX format as hess = V^T Mhess V, where V^T = LeftBorder, given by

$$V^T = \left(k,\ (x - y^{-1})^{-1}h,\ (x - y^{-1})^{-1}xk\right),$$

has linearly independent coefficients, and the matrix Mhess is

$$\text{Mhess} = \begin{pmatrix} x(x-y^{-1})^{-1}y^{-1} & 1 - x(x-y^{-1})^{-1} & -x(x-y^{-1})^{-1}y^{-1} \\ 1 - (x-y^{-1})^{-1}x & (x-y^{-1})^{-1} & -1 + (x-y^{-1})^{-1}x \\ -x(x-y^{-1})^{-1}y^{-1} & -1 + x(x-y^{-1})^{-1} & x(x-y^{-1})^{-1}y^{-1} \end{pmatrix}.$$

The LDL$^{\mathrm{T}}$ decomposition of the coefficient matrix Mhess is given by the command

In[16]:= {lu, di, up, P} = **NCLDUDecomposition**[Mhess];

From the output of this command we obtain the following factorization for P Mhess P^T = lu di up

$$\begin{pmatrix} I & 0 & 0 \\ y^{-1} & I & 0 \\ -y^{-1} & -I & I \end{pmatrix} \begin{pmatrix} (x-y^{-1})^{-1} & 0 & 0 \\ 0 & y^{-1} & 0 \\ 0 & 0 & 0 \end{pmatrix} \begin{pmatrix} I & y^{-1} & -y^{-1} \\ 0 & I & -I \\ 0 & 0 & I \end{pmatrix}. \tag{3.11}$$

Finally, the list returned by **NCConvexityRegion** is

$$\{(x - y^{-1})^{-1},\quad y^{-1},\ 0\}.$$

Thus the region of convexity for F contains

$$\mathcal{G} := \{(x, y) : y > 0 \quad \text{and} \quad x - y^{-1} > 0\}. \tag{3.12}$$

Naturally, this is the same domain that was already determined in (3.10).

To insure that closure($\mathcal{G}$) := $\{(x, y) : y > 0 \quad \text{and} \quad x - y^{-1} \geq 0\}$ contains the biggest region of convexity of F, we must verify hypotheses (i) and (ii) of Theorem 3.3.1. The linear dependence of the coefficients of the border vector states, as in hypothesis (i), that there exist λ_1, λ_2 scalars such that $\lambda_1 I + (x - y^{-1})^{-1}x\lambda_2 = 0$ for all symmetric x, y. It

follows that the coefficients of the border vector are linearly independent. Now we say a few words about a practical test guaranteeing linear independence of the border vector, that is guaranteeing hypotheses (i) of Theorem 3.3.1. This test is implemented in the command NCConvexityRegion. The idea is to declare all variables to commute; then compute a linear combination of the coefficient functions of the border vector which is 0. If the only linear combination is 0, then this insures that condition (i) holds. This is a conservative test and our example passes it.

To check condition (ii) of Theorem 3.3.1, without going into the topology involved, we just say that because the inequalities in 3.12 are strict, the set of $n \times n$ symmetric matrices which satisfy them (for each large n) contains an open set. This suffices to satisfy (ii).

We should emphasize the fact that if we conclude that a function is matrix convex, it could be quite possible that the function actually is matrix "strictly" convex. This happens because we do not have a way to guarantee a unique representation for the matrix $M_{\mathcal{Q}}$. However, the biggest possible domain of convexity of F, the "closure" of $\mathcal{G}$, is uniquely determined whatever representation is used.

Now we discuss permutations. One can observe that for this example (the 3×3 case) there are 12 LDL$^{\mathrm{T}}$ factorizations, related to all possible permutations. We computed them and found that four permutations provide identical decompositions to the one in (3.11), four permutations give division[8] by 0, and the other four give the following diagonal matrix:

$$\begin{pmatrix} -x + x(x - y^{-1})^{-1}x & 0 & 0 \\ 0 & \left\{ \begin{array}{l} -y + x^{-1} - (x - y^{-1})^{-1} + yx(x - y^{-1})^{-1} \\ +(x - y^{-1})^{-1}xy + (x - y^{-1})^{-1}x(x - y^{-1})^{-1} \\ -(x - y^{-1})^{-1}xyx(x - y^{-1})^{-1} \end{array} \right\} & 0 \\ 0 & 0 & 0 \end{pmatrix}. \quad (3.13)$$

Example 3.5.4 Define the function Γ as

$$F := -X + Y - (Y + A^T XB)(R + B^T XB)^{-1}(Y + B^T XA) + A^T XA, \quad (3.14)$$

with $X = X^T$, $Y = Y^T$ and $R = R^T$. In Mathematica it takes the form

```
In[17]:= F := - X + Y - (Y+tp[A]**X**B) ** inv[R+tp[B]**X**B] ** (Y+tp[B]**X**A)
          + tp[A]**X**A;
```

[8] NCLDUDecomposition[ ] contains (automatic) logical rules for permutations to bypass division by 0. Using this automatic permutation, which is the default, the four decompositions provide diagonal matrices identical to the one in (3.13).

For that function the Hessian and the coefficient matrix are obtained from the commands:

In[18]:= hess = **NCHessian**[F, {X,H}, {Y,K}] // **NCSimplifyRational**;

In[19]:= {LeftBorder, Mhess, RightBorder} = **NCMatrixOfQuadratic**[hess, H, K];

The LDL$^{\mathrm{T}}$ decomposition of Mhess is obtained by

In[20]:= {lu, di, up, P} = **NCLDUDecomposition**[Mhess] // **NCSimplifyRational**;

From the output of this command we obtain the diagonal matrix di, presented below

$$\mathrm{di} = \begin{pmatrix} -2\,inv[R + tp[B] * *X * *B] & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{pmatrix}.$$

The list returned by **NCConvexityRegion** is the entries of the diagonal matrix di above. The corresponding lower triangular matrix lu is

$$\mathrm{lu} = \begin{pmatrix} I & 0 & 0 & 0 \\ B & I & 0 & 0 \\ -B & 0 & I & 0 \\ -B & 0 & 0 & I \end{pmatrix}.$$

The coefficient matrix is

$$\mathrm{Mhess} = -2 \begin{pmatrix} I \\ B \\ -B \\ -B \end{pmatrix} inv[R + tp[B] * *X * *B] \begin{pmatrix} I & tp[B] & -tp[B] & -tp[B] \end{pmatrix}.$$

Therefore the condition for negative semi-definiteness of Mhess is $R + tp[B] * *X * *B > 0$. In which, one concludes that the function F in (3.14) is concave on the region $\{X : R + tp[B] * *X * *B > 0\}$.

Part 3.II

Theoretical Results and Proofs

Earlier in Section 3.3.2, we saw that positivity of the matrix $M_{\mathcal{Q}}$ implies matrix positivity of the associated quadratic function $\mathcal{Q}$. Also, Example 3.3.1 in Section 3.3.1 gives a glimpse of the main linear independence idea behind the converse. Part II fully addresses the converse; we know that the quadratic function $\mathcal{Q}$ is matrix positive in some sense and we wish to conclude that the matrix $M_{\mathcal{Q}}$ is also matrix positive. Our main results show a substantial class of cases in which this is true. From these results we obtain under weak hypotheses that our Convexity Algorithm determines exactly the correct Symbolic Inequality Domain up to its "closure".

Part II of this chapter is a bit redundant with Part I, so that it can be read without constantly flipping back to Part I.

3.6 Main Theorem on Sufficient Condition for Convexity

As we now see, it is easy to prove that our Convexity Algorithm in Section 3.4 produces a Symbolic Inequality Domain $\mathcal{G}$ on which a noncommutative symmetric rational function Γ is matrix convex on $\mathcal{G}$.

Remark 3.6.1 *We do not analyze the full Convexity Algorithm, but we shall treat only the case where the residual matrix R in the LDU decomposition is identically zero. The reason we do little work on this case is that matrix D can be partitioned as*

$$D = \begin{pmatrix} \rho_1(\vec{Z}) & & & & & & & & \\ & \ddots & & & & & & & \\ & & \rho_{\tilde{d}}(\vec{Z}) & & & & & & \\ & & & 0 & \rho_{\tilde{d}+1}(\vec{Z}) & & & & \\ & & & \rho_{\tilde{d}+1}(\vec{Z})^T & 0 & & & & \\ & & & & & \ddots & & & \\ & & & & & & 0 & \rho_r(\vec{Z}) & \\ & & & & & & \rho_r(\vec{Z})^T & 0 & \\ & & & & & & & & 0 \end{pmatrix}.$$

This matrix D is positive semidefinite for $\vec{\mathcal{Z}}$ only if $\vec{\mathcal{Z}}$ makes $\rho_j(\vec{\mathcal{Z}}) \geq 0$ for $j = 1, \ldots, \tilde{d}$ and $\rho_i(\vec{\mathcal{Z}}) = 0$ for $i = \tilde{d}+1, \ldots, r$. The constraint $\rho_i(\vec{\mathcal{Z}}) = 0$ is very demanding and typically will force the Symbolic Inequality Domain $\mathcal{G}$ to violate the Openness Property. We have not analyzed this situation carefully, since we felt confident that it would not cause difficulties in our Convexity Algorithm. The NCConvexityRegion command lists the domain of convexity

$\mathcal{G}$ for $\Gamma(\vec{Z})$ as those $\vec{Z}$ such that

$$\mathcal{G} = \left\{ \vec{Z} : \rho_j(\vec{Z}) > 0,\ j = 1, \ldots, \tilde{d} \right\} \bigcap \left\{ \vec{Z} : \rho_i(\vec{Z}) = 0,\ i = \tilde{d}+1, \ldots, r \right\}.$$

The strict inequality $\rho_j(\vec{Z}) > 0$ reflects the fact that the LDU algorithm requires invertibility of the ρ_j for $j = 1, \ldots, \tilde{d}$.

Theorem 3.6.2 (Sufficient Condition for Convexity) *Let $\vec{Z} = \{\vec{A}, \vec{X}\}$ and $\Gamma(\vec{Z})$ be a noncommutative symmetric rational function. The function $\Gamma(\vec{Z})$ may be or may not be hereditary*[9]*. Suppose that the coefficient matrix $M_{\mathcal{H}\Gamma}$ of the Hessian $\mathcal{H}\Gamma(\vec{X})[\vec{H}]$ has a noncommutative $L(\vec{Z})D(\vec{Z})L(\vec{Z})^T$ decomposition with diagonal $D(\vec{Z})$ whose entries are all matrix positive on a Symbolic Inequality Domain*[10] *$\mathcal{G}$. Then $\Gamma(\vec{Z})$ is matrix convex on $\mathcal{G}$.*

Proof. It suffices to prove that the Hessian $\mathcal{H}\Gamma(\vec{X})[\vec{H}]$ is a matrix positive quadratic function for $\vec{Z} = \{\vec{A}, \vec{X}\}$ in the Symbolic Inequality Domain $\mathcal{G}$. Let $\mathcal{H}\Gamma(\vec{X})\ [\vec{H}]$ be in the form $V[\vec{H}]^T M_{\mathcal{H}\Gamma} V[\vec{H}]$, where $M_{\mathcal{H}\Gamma} = L(\vec{Z})D(\vec{Z})L(\vec{Z})^T$. Thus

$$\mathcal{H}\Gamma(\vec{X})[\vec{H}] = V[\vec{H}]^T L(\vec{Z})D(\vec{Z})L(\vec{Z})^T V[\vec{H}]. \tag{3.15}$$

Now, substitute for $\vec{Z}$ and $\vec{H}$ in (3.15) any tuple of matrices $\vec{\mathcal{H}}$ and $\vec{\mathcal{Z}} = \{\vec{\mathcal{A}}, \vec{\mathcal{X}}\}$ in $\mathcal{M}(\mathcal{G})$[11] of compatible dimension. Since $D(\vec{\mathcal{Z}})$ has positive semidefinite entries, formula (3.15) implies that $\mathcal{H}\Gamma(\vec{\mathcal{X}})[\vec{\mathcal{H}}]$ is positive semidefinite. This says that $\Gamma(\vec{Z})$ is matrix convex on $\mathcal{G}$. ∎

3.7 Key Definitions

This section presents the definitions essential for the statement of our most general theorem, which shows that no "bigger" Symbolic Inequality Domain than the $\mathcal{G}$ produced by our Convexity Algorithm yields a function Γ which is matrix convex on $\mathcal{G}$. We start with a simple illustrative case and then we present the general case.

3.7.1 Definitions of linearly dependent functions and borders

To make sure there is no confusion in understanding our results and discussion of borders we include notational discussion which looks at the border of a quadratic function $\mathcal{Q}$ carefully.

[9] *Defined in Section 3.2.1, Part I.*
[10] *Defined in Section 3.2.4, Part I.*
[11] Defined in Section 3.2.4, Part I.

The basic idea

Now we illustrate what we mean by linearly independent border vector. For simplicity of exposition, the hereditary function $\mathcal{Q}$ is limited to be quadratic in two noncommutative variables H_1 and H_2 ($\vec{H} := \{H_1, H_2\}$). In the next section, we will extend the idea to the case of several variables. Let the hereditary quadratic function $\mathcal{Q}(\vec{Z})[\vec{H}]$ take the form

$$\begin{aligned}\mathcal{Q}(\vec{Z})[\vec{H}] =& \sum_{s=1}^{\ell_1}\sum_{t=1}^{\ell_1} {L_s^1}^T(\vec{Z}) H_1^T A_{s,t}(\vec{Z}) H_1 L_t^1(\vec{Z}) \\ &+ sym \sum_{s=1}^{\ell_1}\sum_{t=1}^{\ell_2} {L_s^1}^T(\vec{Z}) H_1^T A_{s,t+\ell_1}(\vec{Z}) H_2 L_t^2(\vec{Z}) \\ &+ \sum_{s=1}^{\ell_2}\sum_{t=1}^{\ell_2} {L_s^2}^T(\vec{Z}) H_2^T A_{s+\ell_1,\, t+\ell_1}(\vec{Z}) H_2 L_t^2(\vec{Z}).\end{aligned}$$

Where each $L_j^i(\vec{Z})$ is a rational function not necessarily distinct; may even be the identity matrix. The quantity ℓ_i is the number of times that the monomial of order two in H_i appears. For the case above, the border of the matrix valued function $\mathcal{Q}(\vec{Z})[\vec{H}]$ has the form

$$V(\vec{Z})[\vec{H}] := \begin{pmatrix} H_1 L_1^1(\vec{Z}) \\ H_1 L_2^1(\vec{Z}) \\ \vdots \\ H_1 L_{\ell_1}^1(\vec{Z}) \\ H_2 L_1^2(\vec{Z}) \\ \vdots \\ H_2 L_{\ell_2}^2(\vec{Z}) \end{pmatrix}. \tag{3.16}$$

In this border, the H_1 and H_2 parts operate independently, so we shall consider separately the polynomials, which are the coefficients of H_1 and H_2:

$$\vec{L}^1(\vec{Z}) := \{L_1^1(\vec{Z}), \ldots, L_{\ell_1}^1(\vec{Z})\} \tag{3.17}$$

and

$$\vec{L}^2(\vec{Z}) := \{L_1^2(\vec{Z}), \ldots, L_{\ell_2}^2(\vec{Z})\}. \tag{3.18}$$

Definition 3.7.1 (Linearly Independent Functions Property)
For a given i, the noncommutative rational functions $L_j^i(\vec{Z})$ for $j = 1, \ldots, \ell_i$ are said to be **linearly independent functions** *if the only scalars λ_j, such that*

$$\sum_{j=1}^{\ell_i} \lambda_j L_j^i(\vec{Z}) = 0$$

are $\lambda_1 = \lambda_2 = \cdots = \lambda_{\ell_i} = 0$. *We emphasize that the scalars* λ_j *do not depend on* $\vec{Z}$. *If there exists such nonzero scalars, the functions* $L^i_j(\vec{Z})$ *are said to be* **linearly dependent functions**.

As we shall see what is critical for our Convexity Algorithm is when either $\vec{L}^1(\vec{Z})$ or $\vec{L}^2(\vec{Z})$ is a **linearly dependent set of functions**. We say that the border vector $V(\vec{Z})[\vec{H}]$ in (3.16) has block linearly independent coefficients, if neither the functions $\vec{L}^1(\vec{Z})$ in (3.17) nor the functions $\vec{L}^2(\vec{Z})$ in (3.18) are linearly dependent. In the next section, we repeat all of these definitions for the most general case.

The general case

In the most general case, the quadratic function $\mathcal{Q}(\vec{Z})[\vec{H}]$ is not constrained to be hereditary. Let's define $\vec{H}$ as

$$\vec{H} := \{H_{-h}, \ldots, H_{-1}, H_1, \ldots, H_h, H_{h+1}, \ldots, H_g, H_{g+1}, \ldots, H_k\}, \tag{3.19}$$

where $\{H_j\}_{j=g+1}^k$ are constrained to be symmetric and $H_j = H_{-j}^T$, for $j = 1, \ldots, h$. That is, we can separate $\vec{H}$ into three different parts as follows: the first part[12] $\{H_j\}_{j=-h}^h$ has the pairwise restriction that $H_{-j} = H_j^T$, for $j = 1, \ldots, h$, the second part $\{H_j\}_{j=h+1}^g$ has no restriction, the third part $\{H_j\}_{j=g+1}^h$ has each H_j constrained to be symmetric. Let $\mathcal{I}$ denote the integers between $-h$ and k except for 0. This is the index set for the H_j which are the entries of $\vec{H}$.

Any noncommutative symmetric quadratic $\mathcal{Q}(\vec{Z})[\vec{H}]$ can be put in the form

$$V(\vec{Z})[\vec{H}]^T M_{\mathcal{Q}(\vec{Z})} V(\vec{Z})[\vec{H}]$$

where the border $V(\vec{Z})[\vec{H}]$ has the form

$$V(\vec{Z})[\vec{H}] := \begin{pmatrix} V^{mix}(\vec{Z})[\vec{H}] \\ V^{pure}(\vec{Z})[\vec{H}] \\ V^{sym}(\vec{Z})[\vec{H}] \end{pmatrix}, \tag{3.20}$$

[12]The integer 0 is not included in the index set $j = -h, \ldots, h$ of the first part, but for simplicity of notation we do not make this explicit, since it is clear from context.

with $V^{mix}(\vec{Z})[\vec{H}]$, $V^{pure}(\vec{Z})[\vec{H}]$, and $V^{sym}(\vec{Z})[\vec{H}]$ defined as follows:

$$V^{mix}(\vec{Z})[\vec{H}] = \begin{pmatrix} H_{-h}L_1^{-h}(\vec{Z}) \\ \vdots \\ H_{-h}L_{\ell_{-h}}^{-h}(\vec{Z}) \\ \vdots \\ H_{-1}L_1^{-1}(\vec{Z}) \\ \vdots \\ H_{-1}L_{\ell_{-1}}^{-1}(\vec{Z}) \\ H_1L_1^1(\vec{Z}) \\ \vdots \\ H_1L_{\ell_1}^1(\vec{Z}) \\ \vdots \\ H_hL_1^h(\vec{Z}) \\ \vdots \\ H_hL_{\ell_h}^h(\vec{Z}) \end{pmatrix} \qquad V^{pure}(\vec{Z})[\vec{H}] = \begin{pmatrix} H_{h+1}L_1^{h+1}(\vec{Z}) \\ \vdots \\ H_{h+1}L_{\ell_{h+1}}^{h+1}(\vec{Z}) \\ \vdots \\ H_gL_1^g(\vec{Z}) \\ \vdots \\ H_gL_{\ell_g}^g(\vec{Z}) \end{pmatrix}$$

$$V^{sym}(\vec{Z})[\vec{H}] = \begin{pmatrix} H_{g+1}L_1^{g+1}(\vec{Z}) \\ \vdots \\ H_{g+1}L_{\ell_{g+1}}^{g+1}(\vec{Z}) \\ \vdots \\ H_kL_1^k(\vec{Z}) \\ \vdots \\ H_kL_{\ell_k}^k(\vec{Z}) \end{pmatrix}$$

In order to illustrate the above definitions, we give a simple example of a quadratic function and its border vector representation. Let the quadratic function $\mathcal{Q}(\vec{Z})[\vec{H}]$ be given by $\mathcal{Q}(\vec{Z})[\vec{H}] = H_1^T * H_1 + H_1 * H_1^T + H_2 * H_2^T + H_3^T * H_3 + H_4 * H_4$, where H_1, H_2, and H_3 are not symmetric and $H_4 = H_4^T$. The symbol $*$ means any expression that does not contain H_i. For this quadratic, the border vector has the following structure:

$$V[\vec{H}] = \begin{pmatrix} H_1 \\ H_1^T \end{pmatrix} \text{ Mixed} \\ \begin{pmatrix} H_2^T \\ H_3 \end{pmatrix} \text{ Pure} \\ \begin{pmatrix} H_4 \end{pmatrix} \text{ Symmetric}$$

Note that this representation of $\mathcal{Q}(\vec{Z})[\vec{H}]$ might require simple relabeling of variables.

For example, if $\mathcal{Q}[\{H,K\}] = H^T AH + KBK^T$, then $H_1 = H$, $H_2 = K^T$ and

$$V[\overrightarrow{H}] = V^{pure}[\overrightarrow{H}] = \begin{pmatrix} H_1 \\ H_2 \end{pmatrix}. \tag{3.21}$$

Indeed, the representations with only $V^{pure}[\overrightarrow{H}]$ give precisely the hereditary[13] $\mathcal{Q}$. Allowing simple relabeling of variables increases the scope of such representations to include all cases like those in example (3.21).

Definition 3.7.2 (Block Linearly Dependent Coefficients)
The border vector $V(\overrightarrow{Z})[\overrightarrow{H}]$ in (3.20) has **block linearly dependent coefficients** *if for some i the functions $L^i_j(\overrightarrow{Z})$ for $j = 1, \ldots, \ell_i$ are linearly dependent, otherwise the border vector $V(\overrightarrow{Z})[\overrightarrow{H}]$ has* **block linearly independent coefficients**.

The "block" nature of the definition above is because we shall often consider separately the set

$$\overrightarrow{L}^i(\overrightarrow{Z}) := \{L^i_1(\overrightarrow{Z}), \ldots, L^i_{\ell_i}(\overrightarrow{Z})\}$$

for each $i \in \mathcal{I}$.

3.7.2 Substituting matrices for indeterminates

In this section we discuss the substitution of matrices for indeterminates and give some definitions. Let $\overrightarrow{Z} = \{Z_1, \ldots, Z_v\}$ be all indeterminates (variables) occurring in whatever noncommutative rational functions $\Gamma(\overrightarrow{Z})$ and constraints $\mathcal{G}$ we are studying. If these indeterminates are replaced by matrices we must be careful to replace them by tuple of matrices $\overrightarrow{\mathcal{Z}} := \{\mathcal{Z}_1, \ldots, \mathcal{Z}_v\}$ of **sizes**

$$\overrightarrow{\mathcal{Z}}^{\#} := \{m_1 \times n_1, \ldots, m_v \times n_v\}$$

compatible with the function $\Gamma(\overrightarrow{Z})$ and the constraints $\mathcal{G}$. Let C^{dim} denote the set of all compatible dimensions. A partial order $\succeq$ on C^{dim}, denoted by $\overrightarrow{\mathcal{Z}}^{\#} \succeq \overrightarrow{\mathcal{Z}}^{a\#}$, is given by

$$\{m_1 \geq m_1^a, n_1 \geq n_1^a, \ldots, m_v \geq m_v^a, n_v \geq n_v^a\},$$

and if strict inequality holds in every entry we write $\overrightarrow{\mathcal{Z}}^{\#} \succ \overrightarrow{\mathcal{Z}}^{a\#}$. Once a size $\Delta \in C^{dim}$ has been selected we let $\mathcal{M}_\Delta$ denote the set of all v tuples of matrices of size Δ. Moreover,

[13] Note that in our definition of hereditary the variables H_j can not be constrained to be symmetric.

if $\mathcal{G}$ is a Symbolic Inequality Domain, then let $\mathcal{M}(\mathcal{G})$ (resp. $\mathcal{M}_\Delta(\mathcal{G})$) denote the set of all matrices meeting the constraints defining $\mathcal{G}$ (resp. and lying in $\mathcal{M}_\Delta$). Often we suppress the subscript Δ because its presence is clear from context.

Definition 3.7.3 (Openness Property)
The domain $\mathcal{G}$ has the Openness Property provided that there is a size Δ_0 in C^{dim} with the property that when indeterminates are replaced by matrices with size $\Delta \succeq \Delta_0$, then the set of matrices $\mathcal{M}_\Delta(\mathcal{G})$ is contained in the closure of the interior of $\mathcal{M}_\Delta(\mathcal{G})$.

3.8 Theorems on Convexity and Positivity

3.8.1 Main result on convexity: Theorem 3.8.2

Theorem 3.8.2, which follows, gives a test which can in fact be implemented with a noncommutative Gröbner basis algorithm (Fröberg (1997); Mora (1986, 1994)). The linear dependence check is purely algebraic and can be performed automatically by computer (software willing). We have not considered seriously the practicality of the Openness Property. However, in all the examples we have done, it is obvious that the set $\mathcal{G}$ obtained satisfy it. Now we set down a class of quadratic functions for which the theory works. The definition also serves as a reminder of Theorem 3.3.3 on LDL^T decompositions.

Definition 3.8.1 (Nice Quadratic on a Symbolic Inequality Domain)
A noncommutative symmetric function $\mathcal{Q}(\vec{Z})[\vec{H}]$, which is rational in $\vec{Z}$ and quadratic in $\vec{H}$, can be always put in the form $V(\vec{Z})[\vec{H}]^T\ M_{\mathcal{Q}}(\vec{Z})\ V(\vec{Z})[\vec{H}]$ with $V(\vec{Z})[\vec{H}]$ as in (3.20). Suppose that the coefficient matrix $M_{\mathcal{Q}}(\vec{Z})$ has a noncommutative $L(\vec{Z})\ D(\vec{Z})\ L(\vec{Z})^T$ decomposition (we may have applied some permutation) with $D(\vec{Z})$ a diagonal matrix (no matrix R in Theorem 3.3.3, unless all entries of the matrix R are identically zero) having entries $D_j(\vec{\mathcal{Z}})$, for $j = 1, \ldots, r-1$, each of which are zero or invertible matrices whenever tuple of matrices $\vec{\mathcal{Z}}$ of compatible dimension in $\mathcal{M}_\Delta(\mathcal{G})$ for large enough Δ are substituted for $\vec{Z}$, then we call $\mathcal{Q}(\vec{Z})[\vec{H}]$ a **nice quadratic**.

Theorem 3.8.2 (A Checkable Necessary and Sufficient Condition for Convexity)

Assumptions: *Define $\vec{Z} = \{\vec{A}, \vec{X}\}$ where X_j may or may not be constrained to be symmetric. Let $\Gamma(\vec{Z})$ be any noncommutative symmetric rational function, whose Hessian $\mathcal{H}\Gamma(\vec{Z})[\vec{H}]$ is a nice quadratic, satisfying the following two conditions:*

i. the function $\Gamma(\overrightarrow{Z})$ is matrix convex for $\overrightarrow{Z}$ on a Symbolic Inequality Domain $\mathcal{G}$ satisfying the Openness Property for some big enough Δ_0;

ii. the border vector $V(\overrightarrow{Z})[\overrightarrow{H}]$ of the Hessian $\mathcal{H}\Gamma(\overrightarrow{Z})[\overrightarrow{H}]$ has block linearly independent coefficients.

Conclusion: *The following statements are equivalent:*

a. when any tuple of matrices $\overrightarrow{\mathcal{Z}}$ in $\mathcal{M}_\Delta(\mathcal{G})$ of compatible dimension $\Delta \succeq \Delta_0$ is substituted into the Hessian $\mathcal{H}\Gamma$, we obtain $\mathcal{H}\Gamma(\overrightarrow{\mathcal{Z}})[\overrightarrow{\mathcal{H}}] \succeq 0$ for all $\overrightarrow{\mathcal{H}}$.

b. for all tuple of matrices $\overrightarrow{\mathcal{Z}}$ in the closure of $\mathcal{M}_\Delta(\mathcal{G})$ the diagonal entries of the $L(\overrightarrow{\mathcal{Z}})$ $D(\overrightarrow{\mathcal{Z}})$ $L(\overrightarrow{\mathcal{Z}})^T$ decomposition are positive semidefinite matrices (that is $D(\overrightarrow{\mathcal{Z}}) \succeq 0$) provided that $D(\overrightarrow{\mathcal{Z}})$ is defined.

Proof. That (b) implies (a) is easy to prove and follows from Theorem 3.6.2. That (a) implies (b) is difficult to prove and follows from:

- the next Theorem 3.8.3 which applies only to quadratic functions and proves under appropriate hypotheses that $\mathcal{H}\Gamma(\overrightarrow{\mathcal{Z}})[\overrightarrow{\mathcal{H}}] \succeq 0$ implies $M_{\mathcal{H}\Gamma}(\overrightarrow{\mathcal{Z}}) \succeq 0$ for $\overrightarrow{\mathcal{Z}}$ defined as in (a) above;
- and that $M_{\mathcal{H}\Gamma}(\overrightarrow{\mathcal{Z}}) \succeq 0$ implies $D(\overrightarrow{\mathcal{Z}}) \succeq 0$, which is true since

$$M_{\mathcal{H}\Gamma}(\overrightarrow{\mathcal{Z}}) = L(\overrightarrow{\mathcal{Z}})D(\overrightarrow{\mathcal{Z}})L(\overrightarrow{\mathcal{Z}})^T$$

  with $L(\overrightarrow{\mathcal{Z}})$ an invertible matrix.[14]

■

3.8.2 Main result on quadratic functions: Theorem 3.8.3

This section gives results about quadratic functions. The main result is Theorem 3.8.3 that concerns positivity of a noncommutative rational function $\mathcal{Q}(\overrightarrow{Z})[\overrightarrow{H}]$ which is quadratic in $\overrightarrow{H}$. The statement of this theorem is presented in this section and its proof is finished in Section 3.10.

[14] $L(\overrightarrow{\mathcal{Z}})$ is an invertible matrix since it is lower triangular with ones on its diagonal.

Theorem 3.8.3 (Main Result on Quadratic Functions)

Assumptions: *Let* $\overrightarrow{H} := \{H_{-h}, \ldots, H_k\}$ *be defined as in (3.19). Consider a noncommutative rational function* $\mathcal{Q}(\overrightarrow{Z})[\overrightarrow{H}]$ *which is a quadratic*[15] *in the variables* $\overrightarrow{H}$ *on a Symbolic Inequality Domain* $\mathcal{G}$. *Write* $\mathcal{Q}(\overrightarrow{Z})[\overrightarrow{H}]$ *in the form*

$$\mathcal{Q}(\overrightarrow{Z})[\overrightarrow{H}] = V(\overrightarrow{Z})[\overrightarrow{H}]^T M_{\mathcal{Q}(\overrightarrow{Z})} V(\overrightarrow{Z})[\overrightarrow{H}].$$

Suppose that the following two conditions hold:

i. the Symbolic Inequality Domain $\mathcal{G}$ *satisfies the Openness Property for some big enough* Δ_0;

ii. the border vector $V(\overrightarrow{Z})[\overrightarrow{H}]$ *of the quadratic function* $\mathcal{Q}(\overrightarrow{Z})[\overrightarrow{H}]$ *has block linearly independent coefficients.*

Conclusion: *The following statements are equivalent:*

a. when any tuple of matrices $\overrightarrow{\mathcal{Z}}$ *in* $\mathcal{M}_\Delta(\mathcal{G})$ *of compatible dimension* $\Delta \succeq \Delta_0$ *is substituted into* $\mathcal{Q}$, *we obtain* $\mathcal{Q}(\overrightarrow{\mathcal{Z}})[\overrightarrow{\mathcal{H}}]$ *is a positive semidefinite matrix for each tuple of matrices* $\overrightarrow{\mathcal{H}}$;

b. we have $M_{\mathcal{Q}(\overrightarrow{\mathcal{Z}})} \geq 0$ *for all* $\overrightarrow{\mathcal{Z}}$ *in the closure of* $\mathcal{M}_\Delta(\mathcal{G})$ *on which* $M_{\mathcal{Q}(\overrightarrow{\mathcal{Z}})}$ *is defined.*

Proof. Clearly (b) implies (a). The hard part is (a) implies (b). The proof of this result consumes the following Section 3.9 and is finalized in Section 3.10. ∎

3.9 Theorems Concerning Quadratic Functions

Before beginning the proof of Theorem 3.8.3 in earnest, we sketch some of the ideas for the simplest type of quadratic functions. Section 3.9, which consist of Section 3.9.1 and Section 3.9.2, concerns primarily a matrix valued quadratic function $\mathcal{Q}[\overrightarrow{\mathcal{H}}]$ of tuple $\overrightarrow{\mathcal{H}}$ of $n \times n$ matrices; there is no dependence on symbolic variables or on variables $\overrightarrow{Z}$. In Section 3.9.1, we treat quadratic functions which are hereditary in the variables $\overrightarrow{\mathcal{H}}$.

Later, in Section 3.10, we begin to combine the matrix results of Section 3.9.1 with symbolic variables, and also we study quadratic functions of $\overrightarrow{H}$ which also depend on $\overrightarrow{Z}$. We reemphasize that the function $\mathcal{Q}(\overrightarrow{Z})[\overrightarrow{H}]$ is quadratic in $\overrightarrow{H}$, but it need not be quadratic in $\overrightarrow{Z}$.

[15] *We emphasize that* $\mathcal{Q}(\overrightarrow{Z})[\overrightarrow{H}]$ *is not restricted to be a* **nice quadratic**.

3.9.1 Some ideas of the proof

This section gives a very special case of Theorem 3.8.3 in order to illustrate a few of the ideas involved and expose the readers to easy cases of the notation. This tutorial proof takes up Section 3.9.1 and then after that the fully general proof begins.

The special case we consider is that of a hereditary quadratic function $\mathcal{Q}[\overrightarrow{\mathcal{H}}]$. To assume that $\mathcal{Q}[\overrightarrow{\mathcal{H}}]$ is a hereditary function is equivalent to imposing that $\overrightarrow{\mathcal{H}}$ has the special form $\overrightarrow{\mathcal{H}} := \{\mathcal{H}_{h+1}, \ldots, \mathcal{H}_g\}$, which in our notation says that $\{\mathcal{H}_i\}_{i=-h}^{h}$ and $\{\mathcal{H}_j\}_{j=g+1}^{k}$ are missing in $\overrightarrow{\mathcal{H}} := \{\mathcal{H}_{-h}, \ldots, \mathcal{H}_{-1}, \mathcal{H}_1, \ldots, \mathcal{H}_h, \mathcal{H}_{h+1}, \ldots, \mathcal{H}_g, \mathcal{H}_{g+1}, \ldots, \mathcal{H}_k\}$. Note that we are treating a purely quadratic function $\mathcal{Q}[\overrightarrow{\mathcal{H}}]$, in other words, $\mathcal{Q}(\overrightarrow{\mathcal{Z}})[\overrightarrow{\mathcal{H}}]$ has no $\overrightarrow{\mathcal{Z}}$ dependence. This special type of $\mathcal{Q}[\overrightarrow{\mathcal{H}}]$ has the following representation

$$\mathcal{Q}[\overrightarrow{\mathcal{H}}] = V^{pure}[\overrightarrow{\mathcal{H}}]^T M_{\mathcal{Q}} V^{pure}[\overrightarrow{\mathcal{H}}],$$

where $V^{pure}[\overrightarrow{\mathcal{H}}]$ is defined as follows

$$V^{pure}[\overrightarrow{\mathcal{H}}] = \begin{pmatrix} \mathcal{H}_{h+1} L_1^{h+1} \\ \vdots \\ \mathcal{H}_{h+1} L_{\ell_{h+1}}^{h+1} \\ \vdots \\ \mathcal{H}_g L_1^g \\ \vdots \\ \mathcal{H}_g L_{\ell_g}^g \end{pmatrix}, \tag{3.22}$$

with each L_j^i being a fixed matrix, that is, they do not depend on matrices $\overrightarrow{\mathcal{Z}}$.

The main result of this section, Proposition 3.9.1, is easy to prove, and serves as an introduction to the ideas of the proof of the main Theorem 3.8.3.

Proposition 3.9.1 (Necessary Condition for Positivity)
Let $\mathcal{Q}[\overrightarrow{\mathcal{H}}]$ be a hereditary quadratic function of tuple $\overrightarrow{\mathcal{H}} = \{\mathcal{H}_j\}_{j=h+1}^{g}$, where each matrix $\mathcal{H}_j$ has dimension $n \times n$. Also assume that this quadratic has a border vector of the type defined in (3.22). Suppose that $\mathcal{Q}[\overrightarrow{\mathcal{H}}]$ is a positive semidefinite matrix for each tuple $\overrightarrow{\mathcal{H}}$, then either

i. the matrix $M_{\mathcal{Q}}$ is positive semidefinite

or

ii. there is an integer $d \in [h+1, g]$ *and real valued functions*

$$\lambda_j : \mathbb{R}^n \to \mathbb{R}, \quad j = 1, \ldots, \ell_d,$$

such that

$$\sum_{j=1}^{\ell_d} \lambda_j(x) L_j^d x = 0, \qquad \text{for } x \in \mathbb{R}^n.$$

We now define some sets that will be used throughout, and especially in the proof of Proposition 3.9.1 above. Let each L_j^i be fixed matrices of dimension $n \times n$. For a given $x \in \mathbb{R}^n$, define the set $\mathcal{R}^{pure,x}_{\vec{L}^i}$ to be

$$\mathcal{R}^{pure,x}_{\vec{L}^i} := \left\{ \begin{pmatrix} \mathcal{H}_i L_1^i x \\ \vdots \\ \mathcal{H}_i L_{\ell_i}^i x \end{pmatrix} : \text{ all } \mathcal{H}_i \in \mathbb{R}^{n \times n} \right\}, \tag{3.23}$$

The Proof of Proposition 3.9.1 follows immediately from Lemma 3.9.2 and Proposition 3.9.3, which we now present.

Lemma 3.9.2 *Let* $\mathcal{Q}[\vec{\mathcal{H}}]$ *be a hereditary quadratic function of tuple* $\vec{\mathcal{H}}$ *of matrices of dimension* $n \times n$*. Also assume that this quadratic has a border vector of the type defined in (3.22). The function* $\mathcal{Q}[\vec{\mathcal{H}}]$ *is positive semidefinite for all* $\vec{\mathcal{H}}$ *implies* $M_{\mathcal{Q}} \geq 0$*, provided that for some* y *the space* $\mathcal{R}^{pure,y}_{\vec{L}^i}$ *fills out the whole space* $\mathbb{R}^{n\ell_i}$ *for all* $i = h+1, \ldots, g$*.*

Proof. Let $\mathcal{Q}[\vec{\mathcal{H}}]$ be positive semidefinite. By definition this implies that $y^T \mathcal{Q}[\vec{\mathcal{H}}] y \geq 0$ for all $y \in \mathbb{R}^n$ and all $\{\mathcal{H}_j\}_{j=h+1}^g \in \mathbb{R}^{n \times n}$. Therefore

$$y^T \mathcal{Q}[\vec{\mathcal{H}}] y = y^T V[\vec{\mathcal{H}}]^T M_{\mathcal{Q}} V[\vec{\mathcal{H}}] y = w^T M_{\mathcal{Q}} w \geq 0$$

for all $w = V[\vec{\mathcal{H}}] y \in \mathbb{R}^{n(\ell_{h+1} + \cdots + \ell_g)}$ and all $\{\mathcal{H}_j\}_{j=h+1}^g \in \mathbb{R}^{n \times n}$. Now it suffices to prove that for some y all vectors of the form w equals $\mathbb{R}^{n(\ell_{h+1} + \cdots + \ell_g)}$. But this condition is directly satisfied from the assumption that the space $\mathcal{R}^{pure,y}_{\vec{L}^i}$ fills out the whole space $\mathbb{R}^{n\ell_i}$ for all $i = h+1, \ldots, g$. ∎

Proposition 3.9.3 *For a given* $x \in \mathbb{R}^n$*, let* $\mathcal{R}^{pure,x}_{\vec{L}^i}$ *be defined as in (3.23). The following holds:*

i. If $\mathcal{R}^{pure,x}_{\vec{L}^i}$ **is** *all of* $\mathbb{R}^{n\ell_i}$*, then* $L_1^i x, L_2^i x, \ldots, L_{\ell_i}^i x$ *are linearly independent vectors.*

ii. If $\mathcal{R}^{pure,x}_{\underset{L}{\rightarrow}^i}$ **is not** *all of* $\mathbb{R}^{n\ell_i}$*, then* L^i_1x*,* L^i_2x*, . . . ,* $L^i_{\ell_i}x$ *are linearly dependent vectors, and consequently there exist nontrivial scalar functions* $\lambda_j(x)$*, that may depend on* x*, such that*

$$\lambda_1(x)L^i_1x + \lambda_2(x)L^i_2x + \cdots + \lambda_{\ell_i}(x)L^i_{\ell_i}x = 0. \tag{3.24}$$

Proof. For a given $x \in \mathbb{R}^n$, let $\mathcal{R}^{pure,x}_{\underset{L}{\rightarrow}^i}$ be all of $\mathbb{R}^{n\ell_i}$. Suppose $L^i_1x, L^i_2x, \ldots, L^i_{\ell_i}x$ are linearly dependent vectors. Without loss of generality, let $L^i_1x = \sum_{j=2}^{\ell_i} \lambda_j(x)\ L^i_jx$, where $\lambda_j(x)$ are scalar functions. Define $s_j = \mathcal{H}_iL^i_jx$, then $\mathcal{R}^{pure,x}_{\underset{L}{\rightarrow}^i}$ becomes

$$\mathcal{R}^{pure,x}_{\underset{L}{\rightarrow}^i} = \left\{ \begin{pmatrix} \lambda_2(x)s_2 + \cdots + \lambda_s(x)s_{\ell_i} \\ s_2 \\ \vdots \\ s_{\ell_i} \end{pmatrix} : \text{ some } s_j \in \mathbb{R}^n \right\}$$

which can not possibly be $\mathbb{R}^{n\ell_i}$. This fact contradicts our assumption on $\mathcal{R}^{pure,x}_{\underset{L}{\rightarrow}^i}$ being all of $\mathbb{R}^{n\ell_i}$, thus $L^i_1x, L^i_2x, \ldots, L^i_{\ell_i}x$ must be a linearly independent set of vectors.

To prove (ii), suppose for a given $x \in \mathbb{R}^n$ the vectors $L^i_1x, L^i_2x, \ldots, L^i_{\ell_i}x$ are linearly independent. Let

$$y = \begin{pmatrix} w_1 \\ \vdots \\ w_{\ell_i} \end{pmatrix}$$

be any vector in $\mathbb{R}^{n\ell_i}$. Then we can choose $\mathcal{H}_i \in \mathbb{R}^{n\times n}$ with the property that $w_1 = \mathcal{H}_iL^i_1x$, $w_2 = \mathcal{H}_iL^i_2x$, . . . , $w_{\ell_i} = \mathcal{H}_iL^i_{\ell_i}x$. Thus $\mathcal{R}^{pure,x}_{\underset{L}{\rightarrow}^i}$ is all of $\mathbb{R}^{n\ell_i}$. ∎

What we have just demonstrated is only the beginning of the proof of Theorem 3.8.3 for a hereditary quadratic function. Next, we must show that the λ_j do not depend on x. For the particular case we have been treating, there are several ways to do this, but they do not all work for the general case of interest. The method we use later to prove that the λ_j are independent of x uses the fact that the quadratic function depends on the variables $\vec{Z}$ (see Theorem 3.10.10 in Section 3.10). Another difficulty is that the sets analogous to $\mathcal{R}^{pure,x}_{\underset{L}{\rightarrow}^i}$ never equal the whole space for the case where $\mathcal{Q}$ is non-hereditary or $\vec{\mathcal{H}}$ contains symmetric elements. Fortunately these sets have co-dimension which depends only on the dimension of the coefficient matrix $M_{\mathcal{Q}}$ and does not depend on the dimension of the matrices contained in the tuple $\vec{\mathcal{Z}}$ substituted for $\vec{Z}$ (See Proposition 3.9.8). We combine this fact about co-dimension with the algebraic dependence of the functions $\mathcal{Q}(\vec{Z})$ and $L^i_j(\vec{Z})$ on $\vec{Z}$ to complete the proof of Theorem 3.8.3 in Section 3.10.

3.9.2 The range of the border vector of a matrix quadratic function

Earlier in Section 3.9.1, a necessary condition for positivity was presented in Proposition 3.9.3 for a particular type of quadratic function. The key was a linear independence property guaranteeing that the space $\mathcal{R}^{pure,x}_{\underset{L}{\rightarrow^i}}$ is all $\mathbb{R}^{n\ell_i}$, that means, the co-dimension of the space $\mathcal{R}^{pure,x}_{\underset{L}{\rightarrow^i}}$ equals zero. Unfortunately, this only characterizes the unconstrained part (the second part) of $\overrightarrow{H}$ defined in (3.19). Section 3.9.2 gives similar conditions on the other two parts of $\overrightarrow{H}$, the pairwise symmetric part (the first part) and the symmetric part (the third part). General quadratic functions are treated in Proposition 3.9.4, and the key property is a uniform bound on certain co-dimensions. Again, as in Section 3.9.1, we study quadratic functions $\mathcal{Q}(\overrightarrow{Z})[\overrightarrow{H}]$ with no $\overrightarrow{Z}$ dependence.

First define $\mathcal{R}^{sym,x}_{\underset{L}{\rightarrow^i}}$ and $\mathcal{R}^{mix,x}_{\underset{L}{\rightarrow^i}}$ to be

$$\mathcal{R}^{sym,x}_{\underset{L}{\rightarrow^i}} := \left\{ \begin{pmatrix} \mathcal{H}_i L_1^i x \\ \vdots \\ \mathcal{H}_i L_{\ell_i}^i x \end{pmatrix} : \text{ all } \mathcal{H}_i = \mathcal{H}_i^T \in \mathbb{R}^{n\times n} \right\}, \tag{3.25}$$

$$\mathcal{R}^{mix,x}_{\underset{L}{\rightarrow^i}} := \left\{ \begin{pmatrix} \mathcal{H}_{-i} L_1^{-i} x \\ \vdots \\ \mathcal{H}_{-i} L_{\ell_{-i}}^{-i} \\ \mathcal{H}_i L_1^i x \\ \vdots \\ \mathcal{H}_i L_{\ell_i}^i x \end{pmatrix} : \text{ all } \mathcal{H}_{-i} = \mathcal{H}_i^T \in \mathbb{R}^{n\times n} \right\}. \tag{3.26}$$

The following Proposition 3.9.4 introduces our main results concerning $\mathcal{R}^{sym,x}_{\underset{L}{\rightarrow^i}}$ and $\mathcal{R}^{mix,x}_{\underset{L}{\rightarrow^i}}$, and also summarizes similar results concerning $\mathcal{R}^{pure,x}_{\underset{L}{\rightarrow^i}}$ given in Proposition 3.9.3.

Proposition 3.9.4 *For a given $x \in \mathbb{R}^n$, let $\mathcal{R}^{pure,x}_{\underset{L}{\rightarrow^i}}$, $\mathcal{R}^{sym,x}_{\underset{L}{\rightarrow^i}}$ and $\mathcal{R}^{mix,x}_{\underset{L}{\rightarrow^i}}$ be defined as in (3.23) and (3.25-3.26). The following holds:*

i. If $\mathcal{R}^{pure,x}_{\underset{L}{\rightarrow^i}}$ **is** *all of $\mathbb{R}^{n\ell_i}$, then $L_1^i x, L_2^i x, \ldots, L_{\ell_i}^i x$ are linearly independent vectors.*

ii. If $\mathcal{R}^{pure,x}_{\underset{L}{\rightarrow^i}}$ **is not** *all of $\mathbb{R}^{n\ell_i}$ (resp. If $\mathcal{R}^{sym,x}_{\underset{L}{\rightarrow^i}}$ has co-dimension in $\mathbb{R}^{n\ell_i}$ greater than $\ell_i[\ell_i - 1]/2$), then $L_1^i x$, $L_2^i x$, …, $L_{\ell_i}^i x$ are linearly dependent vectors, and consequently there exist nontrivial scalar functions $\lambda_j(x)$, that may depend on x, such that*

$$\lambda_1(x) L_1^i x + \lambda_2(x) L_2^i x + \cdots + \lambda_{\ell_i}(x) L_{\ell_i}^i x = 0. \tag{3.27}$$

iii. If $\mathcal{R}^{mix,x}_{\underset{L}{\rightarrow^i}}$ *has co-dimension in* $\mathbb{R}^{n(\ell_i+\ell_{-i})}$ *greater than* $\ell_i\ell_{-i}$*, then either* L^i_1x*,* L^i_2x*, . . . ,* $L^i_{\ell_i}x$ *or* L^{-i}_1x*,* L^{-i}_2x*, . . . ,* $L^{-i}_{\ell_{-i}}x$ *are linearly dependent vectors, and consequently there exist nontrivial scalar functions* $\lambda_j(x)$*, that may depend on* x*, such that either*

$$\lambda_1(x)L^i_1x + \lambda_2(x)L^i_2x + \cdots + \lambda_{\ell_i}(x)L^i_{\ell_i}x = 0 \tag{3.28}$$

or

$$\lambda_1(x)L^{-i}_1x + \lambda_2(x)L^{-i}_2x + \cdots + \lambda_{\ell_{-i}}(x)L^{-i}_{\ell_{-i}}x = 0. \tag{3.29}$$

Proof. The results concerning $\mathcal{R}^{pure,x}_{\underset{L}{\rightarrow^i}}$ were proved in Proposition 3.9.3.

First we treat the case where the $\mathcal{H}_i$ are constrained to be symmetric. If (3.27) fails, then $L^i_1x, \ldots, L^i_{\ell_i}x$ are linearly independent; thus we may use Lemma 3.9.5 below to obtain that $\mathcal{R}^{sym,x}_{\underset{L}{\rightarrow^i}}$ is a space of co-dimension equal to $\ell_i(\ell_i-1)/2$. This contradicts the assumption that $\mathcal{R}^{sym,x}_{\underset{L}{\rightarrow^i}}$ has co-dimension in $\mathbb{R}^{n\ell_i}$ greater than $\ell_i(\ell_i-1)/2$. This proves part (*ii*) of Proposition 3.9.4.

The proof of part (*iii*) follows the same line. If both (3.28) and (3.29) fail, then both $L^i_1x, \ldots, L^i_{\ell_i}x$ and L^{-i}_1x, L^{-i}_2x, . . . , $L^{-i}_{\ell_{-i}}x$ are linearly independent vectors; thus Lemma 3.9.6 below implies that $\mathcal{R}^{mix,x}_{\underset{L}{\rightarrow^i}}$ is a space of co-dimension equal to $\ell_i\ell_{-i}$, contradicting the assumption that $\mathcal{R}^{mix,x}_{\underset{L}{\rightarrow^i}}$ has co-dimension greater than $\ell_i\ell_{-i}$. This completes the proof of Proposition 3.9.4. ∎

Now we present the Lemmas required in the proof of Proposition 3.9.4. We use H instead of $\mathcal{H}$ to stand for a matrix in $\mathbb{R}^{n\times n}$ in Lemma 3.9.5 and Lemma 3.9.6. This makes the rather involved formulas easier to read.

Lemma 3.9.5 *For linearly independent vectors* $v_1, \ldots, v_\ell \in \mathbb{R}^n$ *the space* S *defined by*

$$S = \left\{ \begin{pmatrix} Hv_1 \\ \vdots \\ Hv_\ell \end{pmatrix} : \ \textit{all } H = H^T \in \mathbb{R}^{n\times n} \right\}$$

is a subspace in $\mathbb{R}^{n\ell}$ *with co-dimension* $\ell(\ell-1)/2$.

Proof. Define invertible matrices $P \in \mathbb{R}^{n\times n}$ and $Q \in \mathbb{R}^{\ell\times\ell}$ by

$$\begin{pmatrix} v_1| & \cdots & |v_\ell \end{pmatrix} = P \begin{pmatrix} I \\ 0 \end{pmatrix} Q,$$

where I is the identity matrix with dimension ℓ and $\left(\begin{array}{ccc} v_1| & \cdots & |v_\ell \end{array} \right)$ denotes the matrix whose columns are $v_1, \ldots, v_\ell$. (Note that the hypotheses of this theorem imply $n > \ell$.) The dimension of the space S is

$$\begin{aligned}
\dim(S) &= \dim\left(\left\{ \left(\begin{array}{c} Hv_1 \\ \vdots \\ Hv_\ell \end{array} \right) : \text{ all } H = H^T \in \mathbb{R}^{n\times n} \right\} \right) \\
&= \dim\left(\left\{ H \left(\begin{array}{ccc} v_1| & \ldots & |v_\ell \end{array} \right) : \text{ all } H = H^T \in \mathbb{R}^{n\times n} \right\} \right) \\
&= \dim\left(\left\{ HP \left(\begin{array}{c} I \\ 0 \end{array} \right) Q : \text{ all } H = H^T \in \mathbb{R}^{n\times n} \right\} \right) \\
&= \dim\left(\left\{ HP \left(\begin{array}{c} I \\ 0 \end{array} \right) : \text{ all } H = H^T \in \mathbb{R}^{n\times n} \right\} \right) \\
&= \dim\left(\left\{ P^T HP \left(\begin{array}{c} I \\ 0 \end{array} \right) : \text{ all } H = H^T \in \mathbb{R}^{n\times n} \right\} \right) \\
&= \dim\left(\left\{ \tilde{H} \left(\begin{array}{c} I \\ 0 \end{array} \right) : \text{ all } \tilde{H} = \tilde{H}^T \in \mathbb{R}^{n\times n} \right\} \right) \\
&= n\ell - \ell(\ell-1)/2.
\end{aligned}$$

Thus the co-dimension equals $\ell(\ell-1)/2$. The last step above was a consequence of the following argument. Partition

$$\tilde{H} = \begin{array}{c} \\ \ell \\ n-\ell \end{array} \begin{array}{c} \begin{array}{cc} \ell & n-\ell \end{array} \\ \left(\begin{array}{cc} H_{11} & H_{12} \\ H_{21} & H_{22} \end{array} \right) \end{array}.$$

Then

$$\begin{aligned}
&\dim\left(\left\{ \tilde{H} \left(\begin{array}{c} I \\ 0 \end{array} \right) : \text{ for all } \tilde{H} = \tilde{H}^T \right\} \right) \\
&= \dim\left(\left\{ \left(\begin{array}{c} H_{11} \\ H_{21} \end{array} \right) : \text{ for all } H_{11} = H_{11}^T \in \mathbb{R}^{\ell\times\ell} \text{ and } H_{21} \in \mathbb{R}^{(n-\ell)\times\ell} \right\} \right) \\
&= \dim\left(\left\{ H_{11} : \text{ for all } H_{11} = H_{11}^T \in \mathbb{R}^{\ell\times\ell} \right\} \right) + \\
&\qquad \dim\left(\left\{ H_{21} : \text{ for all } H_{21} \in \mathbb{R}^{(n-\ell)\times\ell} \right\} \right)
\end{aligned}$$

$$= \frac{\ell(\ell+1)}{2} + (n-\ell)\ell$$
$$= n\ell - \ell(\ell-1)/2.$$

■

Lemma 3.9.6 *Suppose that $\{u_i\}_{i=1}^r$ and $\{v_j\}_{j=1}^s$ are two sets of linearly independent vectors in $\mathbb{R}^n$. (The set $\{u_i, v_j\}_{i,j}$ need not consist of linearly independent vectors.) Then the space S defined by*

$$S = \left\{ \begin{pmatrix} Hu_1 \\ \vdots \\ Hu_r \\ H^T v_1 \\ \vdots \\ H^T v_s \end{pmatrix} : \text{ for all } H \in \mathbb{R}^{n\times n} \right\}$$

is a subspace in $\mathbb{R}^{n(r+s)}$ with co-dimension rs.

Proof. Define invertible matrices $P_1 \in \mathbb{R}^{n\times n}$, $Q_1 \in \mathbb{R}^{r\times r}$, $P_2 \in \mathbb{R}^{n\times n}$, and $Q_2 \in \mathbb{R}^{s\times s}$ by

$$\begin{pmatrix} u_1| & \cdots & |u_r \end{pmatrix} = P_1 \begin{pmatrix} I_r \\ 0 \end{pmatrix} Q_1,$$

$$\begin{pmatrix} v_1| & \cdots & |v_s \end{pmatrix} = P_2 \begin{pmatrix} I_s \\ 0 \end{pmatrix} Q_2,$$

where I_r and I_s are the identity matrices with dimension r and s respectively. (Note that the hypotheses of this theorem imply $n > r$ and $n > s$.) The dimension of the space S is

$$\dim(S) = \dim\left(\left\{ \begin{pmatrix} Hu_1 \\ \vdots \\ Hu_r \\ H^T v_1 \\ \vdots \\ H^T v_s \end{pmatrix} : \text{ for all } H \in \mathbb{R}^{n\times n} \right\} \right)$$

$$= \dim\Big(\Big\{ \begin{pmatrix} H \begin{pmatrix} u_1| & \cdots & |u_r \end{pmatrix} & | & H^T \begin{pmatrix} v_1| & \cdots & |v_s \end{pmatrix} \end{pmatrix} : \text{for all } H \in \mathbb{R}^{n\times n} \Big\} \Big)$$

$$
\begin{aligned}
&= \dim\left(\left\{\left(\begin{array}{c} HP_1\begin{pmatrix} I_r \\ 0 \end{pmatrix} Q_1 \end{array} \;\middle|\; H^T P_2 \begin{pmatrix} I_s \\ 0 \end{pmatrix} Q_2 \right) : \text{ for all } H \in \mathbb{R}^{n\times n}\right\}\right) \\
&= \dim\left(\left\{\left(\begin{array}{c} HP_1\begin{pmatrix} I_r \\ 0 \end{pmatrix} Q_1 \end{array} \;\middle|\; H^T P_2 \begin{pmatrix} I_s \\ 0 \end{pmatrix} Q_2 \right)\begin{pmatrix} Q_1^{-1} & 0 \\ 0 & Q_2^{-1} \end{pmatrix} : \right.\right. \\
&\qquad \left.\left. \text{for all } H \in \mathbb{R}^{n\times n}\right\}\right) \\
&= \dim\left(\left\{\left(\begin{array}{c} HP_1\begin{pmatrix} I_r \\ 0 \end{pmatrix} \end{array} \;\middle|\; H^T P_2 \begin{pmatrix} I_s \\ 0 \end{pmatrix} \right) : \text{ for all } H \in \mathbb{R}^{n\times n}\right\}\right) \\
&= \dim\left(\left\{\left(\begin{array}{c} P_2^T HP_1\begin{pmatrix} I_r \\ 0 \end{pmatrix} \end{array} \;\middle|\; P_1^T H^T P_2 \begin{pmatrix} I_s \\ 0 \end{pmatrix} \right) : \right.\right. \\
&\qquad \left.\left. \text{for all } H \in \mathbb{R}^{n\times n}\right\}\right) \\
&= \dim\left(\left\{\left(\begin{array}{c} \tilde{H}\begin{pmatrix} I_r \\ 0 \end{pmatrix} \end{array} \;\middle|\; \tilde{H}^T \begin{pmatrix} I_s \\ 0 \end{pmatrix} \right) : \text{ for all } \tilde{H} \in \mathbb{R}^{n\times n}\right\}\right) \\
&= n(r+s) - rs.
\end{aligned}
$$

Thus the co-dimension equals to rs. The last step above follows from the following argument. Partition (assume $r < s$)

$$
\tilde{H} = \begin{array}{c} \\ r \\ s-r \\ n-s \end{array}\begin{array}{c} \begin{array}{ccc} r & s-r & n-s \end{array} \\ \begin{pmatrix} H_{11} & H_{12} & H_{13} \\ H_{21} & H_{22} & H_{23} \\ H_{31} & H_{32} & H_{33} \end{pmatrix} \end{array}. \tag{3.30}
$$

Then

$$
\begin{aligned}
&\dim\left(\left\{\left(\tilde{H}\begin{pmatrix} I_r \\ 0 \end{pmatrix} \middle| \tilde{H}^T \begin{pmatrix} I_s \\ 0 \end{pmatrix}\right) : \text{ for all } \tilde{H} \in \mathbb{R}^{n\times n}\right\}\right) \\
&\quad = \dim\left(\left\{\begin{pmatrix} H_{11} & H_{11}^T & H_{21}^T \\ H_{21} & H_{12}^T & H_{22}^T \\ H_{31} & H_{13}^T & H_{23}^T \end{pmatrix} : \text{ for all } \tilde{H} \in \mathbb{R}^{n\times n} \text{ as in (3.30)} \right\}\right)
\end{aligned}
$$

$$= \dim(\{\tilde{H} : \text{ for all } \tilde{H} \in \mathbb{R}^{n\times n}\}) - \dim\Big(\Big\{(H_{32} \quad H_{33}) : \text{for all } H_{32} \in \mathbb{R}^{(n-s)\times(s-r)} \text{ and } H_{33} \in \mathbb{R}^{(n-s)\times(n-s)}\Big\}\Big)$$

$$= n^2 - [(n-s)(s-r) + (n-s)(n-s)]$$

$$= n(r+s) - rs$$

■

We now present a lemma concerning co-dimensions, which will be used in the proof of Proposition 3.9.8.

Lemma 3.9.7 *Suppose that each S_i for $i = 1, \ldots, k$ is a subspace in $\mathbb{R}^{n_i}$ with co-dimension m_i, then the space $S = \begin{pmatrix} S_1 \\ \vdots \\ S_k \end{pmatrix}$ is a subspace in $\mathbb{R}^{n_1+\cdots+n_k}$ with co-dimension $m_1+\cdots+m_k$.*

Proof. The space S is the direct sum of the spaces $\begin{pmatrix} 0 \\ \vdots \\ S_i \\ \vdots \\ 0 \end{pmatrix}$, each of which has dimension $n_i - m_i$. The dimension of S equals to the sum of the dimensions of S_i, or equivalently the co-dimension of S equals to the sum of the co-dimensions of S_i, which is $m_1 + \cdots + m_k$. ■

Finally, we present Proposition 3.9.8, which introduces our main result concerning the co-dimension of the range of a border vector.

Proposition 3.9.8 *If there is an $x \in \mathbb{R}^n$ such that $L_1^i x, \ldots, L_{\ell_i}^i x$ are linearly independent vectors for every $i \in \mathcal{I}$, then the following space $\mathcal{R}_{\vec{L}}^{all,x}$ has co-dimension less than or equal to $t := t_1 + \cdots + t_k$, where*

$$t_i = \begin{cases} \ell_{-i}\ell_i & \text{for } i = 1, \ldots, h \\ 0 & \text{for } i = h+1, \ldots, g \\ \ell_i(\ell_i - 1)/2 & \text{for } i = g+1, \ldots, k \end{cases}$$

and $\mathcal{R}^{all,x}_{\overrightarrow{L}}$ *is defined as*

$$\begin{pmatrix} \mathcal{R}^{mix,x}_{\overrightarrow{L}^1} \\ \vdots \\ \mathcal{R}^{mix,x}_{\overrightarrow{L}^h} \\ \mathcal{R}^{pure,x}_{\overrightarrow{L}^{h+1}} \\ \vdots \\ \mathcal{R}^{pure,x}_{\overrightarrow{L}^g} \\ \mathcal{R}^{sym,x}_{\overrightarrow{L}^{g+1}} \\ \vdots \\ \mathcal{R}^{sym,x}_{\overrightarrow{L}^k} \end{pmatrix} = \left\{ \begin{pmatrix} \mathcal{H}_{-h} L_1^{-h} x \\ \vdots \\ \mathcal{H}_h L_{\ell_h}^h x \\ \mathcal{H}_{h+1} L_1^{h+1} x \\ \vdots \\ \mathcal{H}_g L_{\ell_g}^g x \\ \mathcal{H}_{g+1} L_1^{g+1} x \\ \vdots \\ \mathcal{H}_k L_{\ell_k}^k x \end{pmatrix} : \begin{array}{l} \textit{for all } \mathcal{H}_i \in \mathbb{R}^{n\times n} \ (i \in \mathcal{I}), \textit{ satis-} \\ \textit{fying the constraints } \mathcal{H}_{-j} = \mathcal{H}_j^T \\ \textit{for } j = 1, \ldots, h \textit{ and } \mathcal{H}_j = \mathcal{H}_j^T \\ \textit{for } j = g+1, \ldots, k \end{array} \right\}$$

Proof. It follows directly from Lemma 3.9.7, Lemma 3.9.6, and Lemma 3.9.5. ∎

3.10 Linear Dependence of Symbolic Functions

Let Δ_0 be a size sufficiently large that the domain $\mathcal{G}$ possesses the Openness Property[16]. Let $\mathcal{N}_{\Delta_0}(\mathcal{G})$ be the subset of the set of all matrices meeting the inequality constraints $\mathcal{M}(\mathcal{G})$ defined by $\mathcal{N}_{\Delta_0}(\mathcal{G}) := \bigcup_{\Delta \geq \Delta_0} \mathcal{M}_\Delta(\mathcal{G})$. Define also three subsets of $\mathcal{N}_{\Delta_0}(\mathcal{G})$, namely $\mathcal{A}$, $\mathcal{B}$, and $\mathcal{C}$, by

$\mathcal{A} := \{\overrightarrow{\mathcal{Z}} \in \mathcal{N}_{\Delta_0}(\mathcal{G})$: the matrix $M_{\mathcal{Q}(\overrightarrow{\mathcal{Z}})}$ has less than or equal to t negative eigenvalues$\}$, where t is defined in Proposition 3.9.8.

$\mathcal{B} := \{\overrightarrow{\mathcal{Z}} \in \mathcal{N}_{\Delta_0}(\mathcal{G})$: for every x with compatible dimension, there exists $i \in \mathcal{I}$ such that the vectors $L_1^i(\overrightarrow{\mathcal{Z}})x, \ldots, L_{\ell_i}^i(\overrightarrow{\mathcal{Z}})x$ are linearly dependent, that is, for each $\overrightarrow{\mathcal{Z}}$ and x, there exists $\lambda_j(\overrightarrow{\mathcal{Z}}, x)$, such that $\sum_{j=1}^{\ell_i} \lambda_j(\overrightarrow{\mathcal{Z}}, x) L_j^i(\overrightarrow{\mathcal{Z}})x = 0\}$. We emphasize that i also depends on $\overrightarrow{Z}$ and x, that is $i = i(\overrightarrow{\mathcal{Z}}, x)$.

$\mathcal{C} := \mathcal{B} \bigcap \mathcal{A}^c$, where $\mathcal{A}^c$ denotes the set-theoretic complement of set $\mathcal{A}$.

We will show later that the set $\mathcal{N}_{\Delta_0}(\mathcal{G})$ is the disjoint union of the two sets $\mathcal{A}$ and $\mathcal{C}$. Let $\mathcal{A}_\Delta$ be the set of tuples in $\mathcal{A}$ with size Δ. Similarly, $\mathcal{C}_\Delta$ is the set of tuples in $\mathcal{C}$ with size Δ. The next three lemmas give basic properties of the sets $\mathcal{A}$, $\mathcal{B}$, and $\mathcal{C}$.

[16] See definition 3.7.3 in Section 3.7.2.

Lemma 3.10.1 *Let the sets $\mathcal{A}$, $\mathcal{B}$, and $\mathcal{C}$ be defined as above. Suppose that the quadratic function $\mathcal{Q}(\vec{\mathcal{Z}})[\vec{\mathcal{H}}]$ is positive semidefinite for all $\vec{\mathcal{H}}$ provided that the variables $\vec{\mathcal{Z}}$, having compatible dimension, are in $\mathcal{N}_{\Delta_0}(\mathcal{G})$. Then the set $\mathcal{N}_{\Delta_0}(\mathcal{G})$ is the union of the sets $\mathcal{A}$ and $\mathcal{B}$, that is, $\mathcal{N}_{\Delta_0}(\mathcal{G}) = \mathcal{A} \bigcup \mathcal{B}$, furthermore, $\mathcal{N}_{\Delta_0}(\mathcal{G})$ is the disjoint union of the sets $\mathcal{A}$ and $\mathcal{C}$.*

Proof. Observe what happens when we replace $\vec{Z}$ by tuple of matrices $\vec{\mathcal{Z}}$ of compatible dimension. Fix a vector x. Suppose that $x^T \mathcal{Q}(\vec{\mathcal{Z}})[\vec{\mathcal{H}}]x \geq 0$ for all $\vec{\mathcal{H}}$. This implies, that $\vec{w}^T M_{\mathcal{Q}(\vec{\mathcal{Z}})} \vec{w} \geq 0$ for all $\vec{w}$ in $\mathcal{R}^{all,x}_{\vec{L}(\vec{\mathcal{Z}})}$. Thus the number of negative eigenvalues of $M_{\mathcal{Q}(\vec{\mathcal{Z}})}$ is less than or equal to the co-dimension of the space $\mathcal{R}^{all,x}_{\vec{L}(\vec{\mathcal{Z}})}$, which by Proposition 3.9.8 either is bounded by t or there is a $d \in \mathcal{I}$, which depends on $\vec{\mathcal{Z}}$ and x, such that $L^d_1(\vec{\mathcal{Z}})x, \ldots, L^d_{\ell_d}(\vec{\mathcal{Z}})x$ are linearly dependent for every vector x with compatible dimension.

As a consequence of the above result, the set $\mathcal{N}_{\Delta_0}(\mathcal{G})$ is the union of the sets $\mathcal{A}$ and $\mathcal{B}$, and consequently the disjoint union of the sets $\mathcal{A}$ and $\mathcal{C}$. In particular, the set $\mathcal{M}_\Delta(\mathcal{G})$ is the disjoint union of $\mathcal{A}_\Delta$ and $\mathcal{C}_\Delta$ for each $\Delta \succeq \Delta_0$. ∎

Lemma 3.10.2 *For every $\Delta \succeq \Delta_0$, suppose the closure of $\mathcal{A}_\Delta$, denoted by* closure($\mathcal{A}_\Delta$)*, contains $\mathcal{M}_\Delta(\mathcal{G})$, in other words, $\mathcal{A}_\Delta$ is dense in $\mathcal{M}_\Delta(\mathcal{G})$. Then $\mathcal{A}_\Delta$ actually equals the whole set $\mathcal{M}_\Delta(\mathcal{G})$.*

Proof. The lemma follows directly from the fact that the eigenvalues of a symmetric matrix continuously depend on the norm of the matrix, c.f. Appendix D of Golub and Loan (1983). ∎

We present some definitions about direct sum and sets which respect direct sums, since they are important tools for proving linear dependence of the coefficient of the border vector.

Definition 3.10.3 (Direct Sum) *Our definition of the direct sum is the usual one, which for two matrices $\mathcal{Z}_1$ and $\mathcal{Z}_2$ is given by*

$$\mathcal{Z}_1 \oplus \mathcal{Z}_2 := \begin{pmatrix} \mathcal{Z}_1 & 0 \\ 0 & \mathcal{Z}_2 \end{pmatrix}.$$

Now, we extend this definition for v tuples of matrices $\vec{\mathcal{Z}} := \{\mathcal{Z}_1, \ldots, \mathcal{Z}_v\}$. For any positive integer J, we denote by $\vec{\mathcal{Z}}^J$ the direct sum $\vec{\mathcal{Z}} \oplus \cdots \oplus \vec{\mathcal{Z}}$ of J copies of $\vec{\mathcal{Z}}$. For instance, the

direct sum of three v tuples of matrices $\vec{\mathcal{Z}}_1 := \{\mathcal{Z}_{11}, \ldots, \mathcal{Z}_{1v}\}$, $\vec{\mathcal{Z}}_2 := \{\mathcal{Z}_{21}, \ldots, \mathcal{Z}_{2v}\}$, *and* $\vec{\mathcal{Z}}_3 := \{\mathcal{Z}_{31}, \ldots, \mathcal{Z}_{3v}\}$ *is given by*

$$\vec{\mathcal{Z}}_1 \oplus \vec{\mathcal{Z}}_2 \oplus \vec{\mathcal{Z}}_3 := \{\mathcal{Z}_{11} \oplus \mathcal{Z}_{21} \oplus \mathcal{Z}_{31}, \ldots, \mathcal{Z}_{1v} \oplus \mathcal{Z}_{2v} \oplus \mathcal{Z}_{3v}\}.$$

Note that from the above definition, if noncommutative functions L_j^i applied to a v tuples of matrices $\vec{\mathcal{Z}}$ produce matrices $L_j^i(\vec{\mathcal{Z}}) \in \mathbb{R}^{n\times n}$, then these functions L_j^i applied to the direct sum $\vec{\mathcal{Z}}^{J}$ produce matrices $L_j^i(\vec{\mathcal{Z}}^{J}) \in \mathbb{R}^{Jn\times Jn}$.

Definition 3.10.4 (A Set Respects Direct Sums) *A set $\mathcal{P}$ is said to respect direct sums if $\vec{\mathcal{Z}}_i$ for $i = 1, \ldots, \mu$ is contained in the set $\mathcal{P}$ implies that the direct sum $\vec{\mathcal{Z}}_i^{J}$ is also contained in $\mathcal{P}$ for each positive integer J. Furthermore, the direct sum $\vec{\mathcal{Z}}_1^{J} \oplus \cdots \oplus \vec{\mathcal{Z}}_\mu^{J}$ is also contained in $\mathcal{P}$.*

We present Proposition 3.10.5 below because it foreshadow a key idea in the proof of Theorem 3.8.3.

Lemma 3.10.5 *Under the same assumptions as Lemma 3.10.1, the set $\mathcal{C}$ (a subset of $\mathcal{B}$) respects direct sums.*

Proof. The proof is by contradiction. Pick $\vec{\mathcal{Z}}_i \in \mathcal{C}$, thus $\vec{\mathcal{Z}}_i \in \mathcal{A}^c$, which means $M_{\mathcal{Q}(\vec{\mathcal{Z}}_i)}$ has at least $t+1$ negative eigenvalues. Next suppose that $\vec{\mathcal{Z}}_i^{J}$ is not contained in $\mathcal{C}$ for some integer J. Then by Lemma 3.10.1, $\mathcal{Z}_i^J$ is contained in $\mathcal{A}$, which by the definition of the set $\mathcal{A}$ implies that $M_{\mathcal{Q}(\vec{\mathcal{Z}}_i^{J})}$ has less than or equal to t negative eigenvalues. On the other hand, by the property of direct sum, the number of negative eigenvalues of $M_{\mathcal{Q}(\vec{\mathcal{Z}}_i^{J})}$ equals J times the number of the negative eigenvalues of $M_{\mathcal{Q}(\vec{\mathcal{Z}}_i)}$. Thus, $M_{\mathcal{Q}(\vec{\mathcal{Z}}_i)}$ also has less than or equal to t negative eigenvalues, which is a contradiction. Hence, $\vec{\mathcal{Z}}_i^{J}$ is contained in $\mathcal{C}$ for all integers J.

Similarly, we can further prove that the direct sum $\vec{\mathcal{Z}}_1^{J} \oplus \cdots \oplus \vec{\mathcal{Z}}_\mu^{J}$ is also contained in $\mathcal{C}$. ∎

3.10.1 Subsets of $\mathcal{B}$ which respect direct sums

The following few lemmas pertain to a subset $\mathcal{P}$ of $\mathcal{B}$ which respects direct sums. The next lemma shows that for a finite set denoted by $\mathcal{S}$, consisting of different elements in $\mathcal{P}$,

we can find a linear combination of the coefficients of the border vector which equals zero for any $\vec{\mathcal{Z}} \in \mathcal{S}$. We actually prove something a little more general. That is,

Lemma 3.10.6 *Let $\mathcal{P}$ be a subset of $\mathcal{B}$ which respects direct sums. Suppose that $\mathcal{S}$ is a finite subset of $\mathcal{P}$. Then, there are scalars $\lambda_j(\mathcal{S})$ and an integer $d(\mathcal{S}) \in \mathcal{I}$ (which depend upon the choice of the set $\mathcal{S}$) such that*

$$\sum_{j=1}^{\ell_{d(\mathcal{S})}} \lambda_j(\mathcal{S}) L_j^{d(\mathcal{S})}(\vec{\mathcal{Z}}) = 0, \tag{3.31}$$

for every $\vec{\mathcal{Z}} \in \mathcal{S}$.

Proof. The proof relies on taking direct sums of matrices. Write the set $\mathcal{S}$ as $\mathcal{S} = \{\vec{\mathcal{Z}}_1, \ldots, \vec{\mathcal{Z}}_\mu\}$, where each $\vec{\mathcal{Z}}_i \in \mathcal{P}$ for $i = 1, \ldots, \mu$. For this proof, it suffices to take each $L_j^d(\vec{\mathcal{Z}}_i)$ to be in $\mathbb{R}^{n \times n}$. Choose $\vec{\mathcal{Z}}^*$ to be the direct sum $\vec{\mathcal{Z}}_1^n \oplus \cdots \oplus \vec{\mathcal{Z}}_\mu^n$, where each $\vec{\mathcal{Z}}_i^n$ for $i = 1, \ldots, \mu$ is the direct sum of n copies of $\vec{\mathcal{Z}}_i$. Define the vector e^* to be

$$e^* := \begin{pmatrix} e_1 \\ \vdots \\ e_n \end{pmatrix} \in \mathbb{R}^{n^2},$$

where the e_k for $k = 1, \ldots, n$ are the standard basis elements for $\mathbb{R}^n$. Also let x^* be a vector that contains μ copies of e^*, that is,

$$x^* = \begin{pmatrix} e^* \\ \vdots \\ e^* \end{pmatrix} \in \mathbb{R}^{\mu n^2}.$$

Since (by assumption) the set $\mathcal{P}$ respects direct sum, $\vec{\mathcal{Z}}^*$ is also contained in $\mathcal{P}$. Then, by the definition of the set $\mathcal{B}$, there exist scalars $\lambda_j(\vec{\mathcal{Z}}^*, x^*)$ and an integer $d \in \mathcal{I}$ (we reemphasize that $d = d(\vec{\mathcal{Z}}^*, x^*)$), such that

$$\sum_{j=1}^{\ell_d} \lambda_j(\vec{\mathcal{Z}}^*, x^*) L_j^d(\vec{\mathcal{Z}}^*) x^* = 0.$$

It follows that

$$\sum_{j=1}^{\ell_d} \lambda_j(\vec{\mathcal{Z}}^*, x^*) L_j^d(\vec{\mathcal{Z}}_i^n) e^* = 0, \quad \text{for} \quad i = 1, \ldots, \mu.$$

This implies that for $i = 1, \ldots, \mu$,

$$\sum_{j=1}^{\ell_d} \lambda_j(\overrightarrow{\mathcal{Z}}^*, x^*) L_j^d(\overrightarrow{\mathcal{Z}}_i) e_k = 0, \quad \text{for} \quad k = 1, \ldots, n.$$

Since the $\{e_k\}_{k=1}^n$, is a basis for $\mathbb{R}^n$, we obtain that

$$\sum_{j=1}^{\ell_d} \lambda_j(\overrightarrow{\mathcal{Z}}^*, x^*) L_j^d(\overrightarrow{\mathcal{Z}}_i) = 0, \quad \text{for} \quad i = 1, \ldots, \mu.$$

Since $(\overrightarrow{\mathcal{Z}}^*, x^*)$ are determined by the choice of the set $\mathcal{S}$, we conclude that

$$\sum_{j=1}^{\ell_{d(\mathcal{S})}} \lambda_j(\mathcal{S}) L_j^{d(\mathcal{S})}(\overrightarrow{\mathcal{Z}}_i) = 0,$$

for each $\overrightarrow{\mathcal{Z}}_i \in \mathcal{S}$, with $\lambda_j(\mathcal{S}) := \lambda_j(\overrightarrow{\mathcal{Z}}^*, x^*)$ and $d(\mathcal{S}) := d(\overrightarrow{\mathcal{Z}}^*, x^*)$. Thus we obtain equation (3.31) required for the lemma. ∎

The next Lemma 3.10.7 extends this result from the finite set $\mathcal{S}$ to the bigger set $\mathcal{M}_\Delta(\mathcal{G})$.

Lemma 3.10.7 *Let $\mathcal{P}$ be a subset of $\mathcal{B}$ which respects direct sums. For $\Delta \succeq \Delta_0$, if there is an open set $\mathcal{U}_\Delta$ contained in $\mathcal{P}_\Delta := \mathcal{P} \bigcap \mathcal{M}_\Delta(\mathcal{G})$, then there exist scalars $\lambda_j(\Delta)$ and an integer $d(\Delta) \in \mathcal{I}$, such that*

$$\sum_{j=1}^{\ell_{d(\Delta)}} \lambda_j(\Delta) L_j^{d(\Delta)}(\overrightarrow{\mathcal{Z}}) = 0,$$

for every $\overrightarrow{\mathcal{Z}} \in \mathcal{M}_\Delta(\mathcal{G})$.

Proof. Fix a size $\Delta \succeq \Delta_0$. Denote by **vec** the map which sends a tuple of matrices $\overrightarrow{\mathcal{Z}}$ in $\mathcal{P}_\Delta$ to their entries arranged as a vector $(y_1, \ldots, y_K) \in \mathbb{R}^K$ as follows

$$\mathbf{vec} : \mathcal{P}_\Delta \to \mathbb{R}^K,$$

where K is total number of entries in the matrices in $\overrightarrow{\mathcal{Z}}$. The order of the arrangement does not matter, but the same order must be used consistently. Denote $\mathbf{vec}^-$ the inverse map of **vec**. Then each entry of the matrix $L_j^i(\overrightarrow{\mathcal{Z}})$ is a rational function of the elements $y_1, \ldots, y_K$. By multiplying through by some polynomials if necessary, we can assume without loss of generality that each entry of $L_j^i(\overrightarrow{\mathcal{Z}})$ is a polynomial in the K variables $y_1, \ldots, y_K$. Let D_r

be the maximum degree of y_r among all of the polynomials which are entries of $L_j^i(\vec{\mathcal{Z}})$, for all i and j.

Since $\mathcal{P}_\Delta$ contains an open set $\mathcal{U}_\Delta$, we can choose a finite set

$$\tilde{\mathcal{S}} := \{(y_1^{v_1}, \ldots, y_K^{v_K}) \in \mathbb{R}^K, \text{ here } v_r = 1, \ldots, D_r + 1 \text{ for all } r = 1, \ldots, K\},$$

such that for every $r = 1, \ldots, K$, the elements $y_r^1, \ldots, y_r^{D_r+1}$ are distinct. That is, the values in each coordinate of $\tilde{\mathcal{S}}$ are distinct. The set $\tilde{\mathcal{S}}$ is a subset of the space $\mathbb{R}^K$. As a consequence, the cardinality Π of the set $\tilde{\mathcal{S}}$ (the number of elements in $\tilde{\mathcal{S}}$) equals $\Pi = \prod_{r=1}^{K} (D_r + 1)$.

Define $\mathcal{S} = \mathbf{vec}^-(\tilde{\mathcal{S}}) \in \mathcal{P}_\Delta$. By Lemma 3.10.6, for each tuple $\vec{\mathcal{Z}} \in \mathcal{S}$, there are constants $\lambda_j(\mathcal{S})$ and an integer $d(\mathcal{S}) \in \mathcal{I}$, both depending on $\mathcal{S}$ such that

$$\sum_{j=1}^{\ell_{d(\mathcal{S})}} \lambda_j(\mathcal{S}) L_j^{d(\mathcal{S})}(\vec{\mathcal{Z}}) = 0, \tag{3.32}$$

for every tuple of matrices $\vec{\mathcal{Z}} \in \mathcal{S}$.

Now we show that (3.32) actually holds for every $\vec{\mathcal{Z}} \in \mathcal{M}_\Delta(\mathcal{G})$. Note that (3.32) can be equivalently written as

$$\sum_{j=1}^{\ell_{d(\mathcal{S})}} \lambda_j(\mathcal{S}) \left[L_j^{d(\mathcal{S})}(\vec{\mathcal{Z}}) \right]_{(p,q)} = 0, \tag{3.33}$$

for every tuple of matrices $\vec{\mathcal{Z}} \in \mathcal{S}$, where $\left[L_j^{d(\mathcal{S})}(\vec{\mathcal{Z}}) \right]_{(p,q)}$ denotes the (p,q)th entry of $L_j^{d(\mathcal{S})}(\vec{\mathcal{Z}})$. By the previous argument, $\left[L_j^{d(\mathcal{S})}(\vec{\mathcal{Z}}) \right]_{(p,q)}$ is a polynomial in the K variables $y_1, \ldots, y_K$, and also the maximum degree on each indeterminate y_r is no greater than D_r. Clearly all the elements in $\tilde{\mathcal{S}}$ give rise to matrix tuple $\vec{\mathcal{Z}}$ that satisfy the polynomial equation (3.33) for all p and q. By the elementary theorem of algebra which says that every nonzero polynomial in one complex variable with degree D_r has at most D_r zeros, we conclude by the construction (cardinality Π) of the set $\tilde{\mathcal{S}}$ that for every $\vec{\mathcal{Z}} \in \mathcal{M}_\Delta(\mathcal{G})$

$$\sum_{j=1}^{\ell_{d(\mathcal{S})}} \lambda_j(\mathcal{S}) \left[L_j^{d(\mathcal{S})}(\vec{\mathcal{Z}}) \right]_{(p,q)} = 0, \quad \text{for each } p \text{ and } q,$$

Thus it follows that

$$\sum_{j=1}^{\ell_{d(\Delta)}} \lambda_j(\Delta) L_j^{d(\Delta)}(\vec{\mathcal{Z}}) = 0,$$

for every $\vec{\mathcal{Z}} \in \mathcal{M}_\Delta(\mathcal{G})$, by choosing constants $\lambda_j(\Delta) := \lambda_j(\mathcal{S})$ and integer $d(\Delta) = d(\mathcal{S})$. ∎

Now we have obtained the linear combination $\lambda_j(\Delta)$ of $L_j^{d(\Delta)}(\overrightarrow{\mathcal{Z}})$, which is zero for all elements $\overrightarrow{\mathcal{Z}}$ in $\mathcal{M}_\Delta(\mathcal{G})$ for one fixed size Δ. The following lemma connects the coefficients $\lambda_j(\Delta)$ of the linear combinations between different size. It says that if we have an annihilating linear combination for $\mathcal{M}_\Delta(\mathcal{G})$, then this same combination will also be annihilated for all size Δ' with $\Delta \succeq \Delta'$.

Lemma 3.10.8 *Fix a size Δ. Suppose there are scalars $\lambda_j(\Delta)$ and an integer $i(\Delta) \in \mathcal{I}$ such that*

$$\sum_{j=1}^{\ell_i(\Delta)} \lambda_j(\Delta) L_j^{i(\Delta)}(\overrightarrow{\mathcal{Z}}) = 0,$$

for every $\overrightarrow{\mathcal{Z}} \in \mathcal{M}_\Delta(\mathcal{G})$. Then

$$\sum_{j=1}^{\ell_i(\Delta)} \lambda_j(\Delta) L_j^{i(\Delta)}(\overrightarrow{\mathcal{Z}}) = 0,$$

for every $\overrightarrow{\mathcal{Z}} \in \mathcal{M}_{\Delta'}(\mathcal{G})$, with $\Delta \succeq \Delta'$.

Proof. Let $\overrightarrow{\emptyset} = \{\emptyset_1, \ldots, \emptyset_v\}$ be a tuple of zero matrices of compatible dimension. For every $\overrightarrow{\mathcal{Z}}_0 \in \mathcal{M}_{\Delta'}(\mathcal{G})$ let $\overrightarrow{\mathcal{Z}}$ be

$$\overrightarrow{\mathcal{Z}} = \overrightarrow{\mathcal{Z}}_0 \oplus \overrightarrow{\emptyset}$$

to get $\overrightarrow{\mathcal{Z}} \in \mathcal{M}_\Delta(\mathcal{G})$ with $\Delta \succeq \Delta'$. By assumption, there are scalars $\lambda_j(\Delta)$ and an integer $i(\Delta)$ such that

$$\sum_{j=1}^{\ell_i(\Delta)} \lambda_j(\Delta) L_j^{i(\Delta)}(\overrightarrow{\mathcal{Z}}) = 0,$$

for every $\overrightarrow{\mathcal{Z}} \in \mathcal{M}_\Delta(\mathcal{G})$. Then plug in the decomposition of $\overrightarrow{\mathcal{Z}}$ given above, together with the fact that

$$L_j^{i(\Delta)}(\overrightarrow{\mathcal{Z}}_0 \oplus \overrightarrow{\emptyset}) = L_j^{i(\Delta)}(\overrightarrow{\mathcal{Z}}_0) \oplus L_j^{i(\Delta)}(\overrightarrow{\emptyset}) = \begin{pmatrix} L_j^{i(\Delta)}(\overrightarrow{\mathcal{Z}}_0) & 0 \\ 0 & 0 \end{pmatrix},$$

to obtain

$$\sum_{j=1}^{\ell_i(\Delta)} \lambda_j(\Delta) L_j^{i(\Delta)}(\overrightarrow{\mathcal{Z}}_0) = 0$$

for every $\overrightarrow{\mathcal{Z}}_0 \in \mathcal{M}_{\Delta'}(\mathcal{G})$. ∎

So far we have shown that for every fixed size Δ, there exists an annihilating linear combination (that may depend on the size Δ), which also holds for any size Δ' with $\Delta \succeq \Delta'$. Now we show that actually there exists an annihilating linear combination for all $\vec{\mathcal{Z}} \in \mathcal{M}_\Delta(\mathcal{G})$ that does not depend on the size Δ.

Lemma 3.10.9 *Let $\mathcal{P}$ be a subset of $\mathcal{B}$ which respect direct sums. Suppose there is a size $\Delta_1 \succeq \Delta_0$, such that for every size $\Delta \succeq \Delta_1$ there is an open set $\mathcal{U}_\Delta$ contained in $\mathcal{P}_\Delta := \mathcal{P} \bigcap \mathcal{M}_\Delta(\mathcal{G})$. Then, there are constants λ_j and an integer $d \in \mathcal{I}$ (we emphasize that λ_j and the integer d do not depend on the size Δ) such that*

$$\sum_{j=1}^{\ell_d} \lambda_j L_j^d(\vec{\mathcal{Z}}) = 0,$$

for every $\vec{\mathcal{Z}} \in \mathcal{M}(\mathcal{G})$.

Proof. Define the set Λ_*^Δ as

$$\Lambda_*^\Delta := \Big\{ (d(\Delta), \lambda_1(\Delta), \ldots, \lambda_{\ell_{d(\Delta)}}(\Delta)) : \sum_{j=1}^{\ell_{d(\Delta)}} \lambda_j(\Delta) L_j^{d(\Delta)}(\vec{\mathcal{Z}}) = 0, \quad \text{for every} \quad \vec{\mathcal{Z}} \in \mathcal{M}_\Delta(\mathcal{G}) \quad \text{and an integer} \quad d(\Delta) \in \mathcal{I} \Big\}.$$

Since for every $\Delta \succeq \Delta_1$, the set $\mathcal{P}_\Delta$ contains an open set $\mathcal{U}_\Delta$, from Lemma 3.10.7 we have that the set Λ_*^Δ is nonempty. Thus there exists a point

$$(\tilde{d}(\Delta), \tilde{\lambda_1}(\Delta), \ldots, \tilde{\lambda}_{\ell_d(\Delta)}(\Delta)) \in \Lambda_*^\Delta$$

for every $\Delta \succeq \Delta_1$. We can define a collection of sets for every $\Delta \succeq \Delta_1$ and every integer $\tilde{d}(\Delta)$ as

$$\Lambda_*^\Delta(\tilde{d}(\Delta)) := \Big\{ (\lambda_1(\Delta), \ldots, \lambda_{\ell_d(\Delta)}(\Delta)) : (\tilde{d}(\Delta), \lambda_1(\Delta), \ldots, \lambda_{\ell_d(\Delta)}(\Delta)) \in \Lambda_*^\Delta \Big\}.$$

It is clear by the construction that $\Lambda_*^\Delta(d(\Delta))$ is a linear space, which is nontrivial since $(\tilde{\lambda_1}(\Delta), \ldots, \tilde{\lambda}_{\ell_d(\Delta)}(\Delta)) \in \Lambda_*^\Delta(\tilde{d}(\Delta))$ for every $\Delta \succeq \Delta_1$. Since the integer $d(\Delta)$ only has finitely many possibilities in $\mathcal{I}$ there exists an infinite increasing sequence $\{j_i\}_{i=1}^\infty$ and an integer d in $\mathcal{I}$, such that $\Lambda_*^{\Delta_{j_i}}(d)$ is nonempty for any i and such that

$$\Delta_{j_{i_1}} \succ \Delta_{j_{i_2}}, \quad \text{for any} \quad i_1 > i_2.$$

By Lemma 3.10.8, the dimension of the space $\Lambda_*^{\Delta_{j_i}}(d)$ is a non-increasing sequence, which is bounded below by 1. Thus

$$\min_{i\geq 1} \dim(\Lambda_*^{\Delta_{j_i}}(d)) \geq 1.$$

Hence $\bigcap_{i\geq 1} \Lambda_*^{\Delta_{j_i}}(d) \neq \emptyset$, and consequently there is an integer d (that does not depend on Δ) and scalars $\lambda_j(\Delta)$, such that

$$\sum_{j=1}^{\ell_d} \lambda_j(\Delta) L_j^d(\vec{\mathcal{Z}}) = 0,$$

for every $\vec{\mathcal{Z}} \in \bigcup_{i=1}^{\infty} \mathcal{M}_{\Delta_{j_i}}(\mathcal{G})$.

So far we have shown that the integer d does not depend on the size Δ. The next step is to show that the scalars λ_j are also independent of Δ. This is accomplished by applying Lemma 3.10.8 successively. Thus, we conclude that

$$\sum_{j=1}^{\ell_d} \lambda_j L_j^d(\vec{\mathcal{Z}}) = 0,$$

for every $\vec{\mathcal{Z}} \in \mathcal{M}(\mathcal{G})$. ∎

From all of this we obtain the following result.

Theorem 3.10.10 *Let $L_1(\vec{Z}), \ldots, L_\ell(\vec{Z})$ be noncommutative rational functions of $\vec{Z} = \{Z_1, \ldots, Z_v\}$. Let $\mathcal{G}$ be a Symbolic Inequality Domain satisfying the Openness Property. Suppose for all $\Delta \succeq \Delta_0$ we have for each $\vec{\mathcal{Z}} \in \mathcal{M}_\Delta(\mathcal{G})$ of compatible dimension and each vector x that the vectors*

$$L_1(\vec{\mathcal{Z}})x, \ldots, L_\ell(\vec{\mathcal{Z}})x$$

are linearly dependent. Then the functions $L_1(\vec{Z}), \ldots, L_\ell(\vec{Z})$ are linearly dependent, that is, there are scalars λ_j (that do not depend on $\vec{Z}$) such that

$$\sum_{j=1}^{\ell} \lambda_j L_j(\vec{Z}) = 0$$

Proof. Form a subset of $\mathcal{B}$ denoted by $\mathcal{P}$ associated with $L_1(\vec{\mathcal{Z}}), \ldots, L_\ell(\vec{\mathcal{Z}})$ by

$$\mathcal{P} = \Big\{ \vec{\mathcal{Z}} \in \mathcal{N}_{\Delta_0}(\mathcal{G}) : \text{ for each } \vec{\mathcal{Z}}, x \text{ there exist } \lambda(\vec{\mathcal{Z}}, x), \text{ such that } \sum_{j=1}^{\ell} \lambda_j L_j(\vec{\mathcal{Z}})x = 0 \Big\}.$$

Now, we show that this set $\mathcal{P}$ respects direct sums. For $t = 1, \ldots, \mu$ let $\vec{\mathcal{Z}}_t$ be contained in $\mathcal{P}$. By definition of the set $\mathcal{P}$, for each $\vec{\mathcal{Z}}_t, x$ there exist $\lambda(\vec{\mathcal{Z}}_t, x)$ such that

$$\sum_{j=1}^{\ell} \lambda_j(\vec{\mathcal{Z}}_t, x) L_j(\vec{\mathcal{Z}}_t)x = 0, \qquad t = 1, \ldots, \mu.$$

Let x^* be a vector that contains J copies of x. Since $L_j(\vec{\mathcal{Z}}_t^J) = L_j(\vec{\mathcal{Z}}_t) \oplus \cdots \oplus L_j(\vec{\mathcal{Z}}_t)$, we have that, for $t = 1, \ldots, \mu$,

$$\sum_{j=1}^{\ell} \lambda_j(\vec{\mathcal{Z}}_t, x) L_j(\vec{\mathcal{Z}}_t^J)x^* = \begin{pmatrix} \sum_{j=1}^{\ell} \lambda_j(\vec{\mathcal{Z}}_t, x) L_j(\vec{\mathcal{Z}}_t)x & & 0 \\ & \ddots & \\ 0 & & \sum_{j=1}^{\ell} \lambda_j(\vec{\mathcal{Z}}_t, x) L_j(\vec{\mathcal{Z}}_t)x \end{pmatrix} = 0,$$

and consequently $\vec{\mathcal{Z}}_t^J \in \mathcal{P}$ for each $t = 1, \ldots, \mu$.

Thus, Lemma 3.10.6, Lemma 3.10.7, Lemma 3.10.8 and Lemma 3.10.9 apply to $\mathcal{P}$. In particular, Lemma 3.10.9 implies Theorem 3.10.10. ∎

Also Theorem 3.10.10 lays behind Corollary 3.10.11, which is here repeated.

Corollary 3.10.11 *Let $L_1(\vec{Z}), \ldots, L_\ell(\vec{Z})$ be noncommutative rational functions of $\vec{Z} = \{Z_1, \ldots, Z_v\}$. For each vector x, suppose that the vectors $L_1(\vec{\mathcal{Z}})x$, …, $L_\ell(\vec{\mathcal{Z}})x$ are linearly dependent whenever matrices $\mathcal{Z}_j$ of compatible dimension are substituted for Z_j for all size Δ bigger than some Δ_0. Then there exist real numbers λ_j for $j = 1, \ldots, \ell$ such that*

$$\sum_{j=1}^{\ell} \lambda_j L_j(\vec{Z}) = 0,$$

that is, the functions $L_j(\vec{Z})$ are linearly dependent.

Proof. In Theorem 3.10.10 take $\mathcal{G}$ to be everything. That is, $\mathcal{G}$ contains no inequality constraints. Thus $\mathcal{G}$ has the Openness Property, since $\mathcal{M}_\Delta(\mathcal{G}) = \mathcal{M}_\Delta$. ∎

We need the following lemmas to complete the proof of the main Theorem.

Lemma 3.10.12 *Let Δ_0 be any size. Assume that T is a symmetric matrix with noncommutative rational functions $t_{ij}(\vec{Z})$ as entries. Suppose there is an integer r such that whenever tuple of matrices $\vec{\mathcal{Z}} \in \mathcal{N}_{\Delta_0}(\mathcal{G})$ of compatible dimension are substituted for $\vec{Z}$, the resulting matrix $T(\vec{\mathcal{Z}})$ has at most r negative eigenvalues. Then $T(\vec{\mathcal{Z}})$ is positive semidefinite (that is, $r = 0$) for each $\vec{\mathcal{Z}} \in \mathcal{M}(\mathcal{G})$.*

Proof. The key fact is

$$T(\vec{\mathcal{Z}} \oplus \vec{\mathcal{Z}}) = T(\vec{\mathcal{Z}}) \oplus T(\vec{\mathcal{Z}}).$$

This implies that if $T(\vec{\mathcal{Z}})$ has η negative eigenvalues, then T applied to the $2r$-fold direct sum $\vec{\mathcal{Z}} \oplus \cdots \oplus \vec{\mathcal{Z}}$ has $2r\eta$ negative eigenvalues. Consequently the hypothesis $2r\eta \leq r$ implies that $\eta = 0$. ∎

Lemma 3.10.13 *Suppose $M_{\mathcal{Q}(\vec{\mathcal{Z}})}$ is positive semidefinite for every $\vec{\mathcal{Z}} \in \mathcal{M}_\Delta(\mathcal{G})$, then $M_{\mathcal{Q}(\vec{\mathcal{Z}})}$ is also positive semidefinite for every $\vec{\mathcal{Z}} \in \mathcal{M}_{\Delta'}(\mathcal{G})$ with $\Delta \succeq \Delta'$.*

Proof. Use an idea similar to the one in the proof of Lemma 3.10.8. ∎

3.10.2 Proof of Theorem 3.8.3

Proof. For any $\Delta \succeq \Delta_0$, if $\mathcal{A}_\Delta$ is dense in $\mathcal{M}_\Delta(\mathcal{G})$, that is, closure$(\mathcal{A}_\Delta) \supseteq \mathcal{M}_\Delta(\mathcal{G})$, then by Lemma 3.10.2, we have $\mathcal{A}_\Delta = \mathcal{M}_\Delta(\mathcal{G})$. Hence, the number of negative eigenvalues of $M_{\mathcal{Q}(\vec{\mathcal{Z}})}$ is uniformly bounded by t for all $\vec{\mathcal{Z}} \in \mathcal{M}_\Delta(\mathcal{G})$. Now we apply Lemma 3.10.12 with $r = t$ to obtain that, for each tuple of matrices $\vec{\mathcal{Z}} \in \mathcal{M}_\Delta(\mathcal{G})$ substituted for $\vec{Z}$, the matrix $M_{\mathcal{Q}(\vec{\mathcal{Z}})}$ is positive semidefinite. On the other hand, if $\mathcal{A}_\Delta$ is not dense in $\mathcal{M}_\Delta(\mathcal{G})$, then by Lemma 3.10.1 there exists an open set $\mathcal{U}_\Delta$ contained in $\mathcal{C}_\Delta \subset \mathcal{M}_\Delta(\mathcal{G})$.

So far we have shown that for any $\Delta \succeq \Delta_0$ one of the following must be satisfied, either

a. the matrix $M_{\mathcal{Q}(\vec{\mathcal{Z}})}$ is positive semidefinite for each $\vec{\mathcal{Z}} \in \mathcal{M}_\Delta(\mathcal{G})$,

or

b. there exists an open set $\mathcal{U}_\Delta$ contained in $\mathcal{C}_\Delta \subset \mathcal{M}_\Delta(\mathcal{G})$.

The final step is to show that if positivity of $M_{\mathcal{Q}(\vec{\mathcal{Z}})}$ fails, then the block linear independence of the border vector (in assumption (ii) of Theorem 3.8.3) of the quadratic function $\mathcal{Q}$ also fails. Assume there is a size Δ^* such that (a) is not satisfied. Then by Lemma 3.10.13, (a) is not satisfied for every $\Delta \succeq \Delta^*$. Hence (b) is true for every $\Delta \succeq \Delta^*$, which by Lemma 3.10.9 (with $\mathcal{P}_\Delta = \mathcal{C}_\Delta$) and Lemma 3.10.5 implies that there are constants λ_j and an integer d such that $\sum_{j=1}^{\ell_d} \lambda_j L_j^d(\vec{\mathcal{Z}}) = 0$ for every $\vec{\mathcal{Z}} \in \mathcal{M}(\mathcal{G})$. Thus, by Corollary 3.10.11, the noncommutative rational functions $L_j^d(\vec{Z})$ are linearly dependent for $j = 1, \ldots, \ell_d$ and consequently the border $V(\vec{Z})[\vec{H}]$ has block linearly dependent coefficients.

But this contradicts assumption (ii) of Theorem 3.8.3, finalizing in this way the proof of the main Theorem 3.8.3. ■

Remark 3.10.14 *It is enough (a weaker hypotheses) to consider square matrices of dimension $n \times n$ (when substituting matrices for indeterminate) to prove the theorems concerning convexity and matrix positive of noncommutative rational functions.*

Chapter 4

Convex Optimization over Matrix Functions

4.1 Introduction

This chapter provides tools that can solve system and control problems, or any other type of engineering problem that can be posed as Matrix Inequalities (MIs). To use these tools, no knowledge of Linear Matrix Inequalities (LMIs) or how to manipulate matrix inequalities to be expressed as LMIs is required. Furthermore, the tools presented here may have the same advantages as the LMI framework.

To understand the motivation of this task, one must expose some of the advantages and disadvantages of the LMI framework. The wide acceptance of LMIs stems from the following facts: 1) if a control problem is posed as an LMI, then any solution is a global optimum; 2) efficient LMI solvers are readily available; 3) once a control problem is posed as an LMI, any other constraints in the form of LMIs can be added to the problem. On the other hand, the LMI framework has the following disadvantages: 1) there is no systematic way to produce LMIs for general classes of problems; 2) there is no way of knowing whether or not it is possible to reduce a system problem to an LMI without actually doing it; 3) the user must possess the knowledge of manipulating LMIs; 4) transformations via *Schur complements* can lead to a large LMI representation.

If someone has the ability to check if a MI is convex and convertible to an LMI, then the optimization problem can be solved by the many available LMI solvers. Most of the numerical implementation of these solvers are based on the *Semidefinite Programming*

(SDP) machinery. To cite a few of them Gahinet et al. (1995); Sturm (1999); Vandenberghe and Balakrishnan (1997); Vandenberghe and Boyd (1995, 1996) and references therein. Fundamental results in primal-dual interior-point methods are found in Wright (1997), and a collection of many results on convex optimization can be found in Boyd and Vandenberghe (2003). An elegant exposition of interior-point methods in convex optimization is found in Renegar (2001). A large collection of control problems that can be posed as LMIs, and algorithms used to solve them, can be found in Boyd et al. (1994); Colaneri et al. (1997). In particular, the book by Skelton et al. (1998) demonstrates that many linear controller design problems reduce to a single linear algebra problem having the form

$$\Gamma G \Delta + (\Gamma G \Delta)^T + \Theta < 0,$$

for the unknown matrix G.

Unfortunately, if one does not have the ability to deal with LMIs, it is not clear what one should do. An available alternative is to restate the entire optimization problem in the form used by some particular numerical nonlinear optimization solver. In this case, since optimization over matrix functions are inherently not smooth, there is no guarantee of a local minimum. Furthermore, the tedious process of reformulating a matrix optimization problem usually requires a high level of algebraic skills.

There are a few papers on solving matrix inequalities which are not linear in the unknowns (Jarre (2000); Leibfritz and Mostafa (2002) and references therein). In Leibfritz and Mostafa (2002), the authors presented and analyzed a numerical interior point trust region algorithm that can be used for solving a class of nonlinear (nonconvex) semidefinite programming problem, posed as:

$$\min_{F,L} J(F,L) \quad \text{s.t.} \quad h(F,L) = 0, \quad Y(F,L) < 0, \quad L > 0 \tag{4.1}$$

where the functions $h, Y : \mathbb{R}^{p\times r} \times \mathbb{S}^n \to \mathbb{S}^n$ and $J : \mathbb{R}^{p\times r} \times \mathbb{S}^n \to \mathbb{R}$ are assumed to be twice continuously differentiable. The author solved the above problem (4.1) by a barrier method. Their formulas for the update directions naturally depend on the functions J, h, and Y, and its derivatives. They have applied the algorithm to a variety of numerical examples, including static output feedback control designs. Their approach requires the user to compute the derivatives of the matrix functions in order to implement the code, while to use our approach the user does not need to compute derivatives, since this is done automatically. Moreover, this thesis focuses much attention on the efficient use of the formulas for the derivatives (see Section 4.6). Even though in this dissertation, we have

focused on convex optimization problems over matrix inequalities, the extension of our ideas to a nonconvex approach is immediate.

The NCSDP numerical optimization solver to be presented in this chapter can be used to solve optimization problems corresponding to matrix inequalities. This approach does not require any knowledge of LMIs, or how to manipulate MIs to be expressed as LMIs. Consequently, there is no need to determine *Schur complements* in order to express the matrix constraints as LMIs. Moreover, since transformations via Schur complements can lead to an LMI representation with large matrices, the presented solver has the potential to reduce the optimization time significantly when the dimensions of the matrices involved are large (when compared to primal-dual numerical solvers, see Section 4.7).

The NCSDP solver is based on an implementation of the method of centers (see Boyd and El Ghaoui (1993); Colaneri et al. (1997); Huard (1967); Lieu and Huard (1965)). This solver is implemented in Matlab and Mathematica, and it can be split into two parts: a symbolic and a numerical one. Roughly speaking, at the symbolic level, Mathematica computes the gradient map $\mathbb{Q}$ and the Hessian map $\mathbb{H}(\delta_X)$ of an auxiliary potential function that appropriately incorporates the objective and the constraints. Thereby, producing a linear system of equations $\mathbb{H}(\delta_X) = \mathbb{Q}$ in the update direction δ_X. Then, a Matlab code numerically solves this system for δ_X. The method successively iterates, at the numerical level, until the algorithm converges to an optimal solution.

To convey what it is meant for minimization over matrix functions, suppose one is given matrices of compatible dimensions A and S where they need to solve the following problem for symmetric matrices X and $Y > 0$ within the unit ball:

$$\max_{X,Y} \operatorname{Tr}\{X\}$$

subject to

$$\begin{aligned} XA^TY^{-1}AX - AX(XA^TY^{-1}AX - Y)^{-1}XA^T - (Y^{-1}XA^TY^{-1}AXY^{-1} - Y^{-1})^{-1} \\ - AX(I + Y^{-1}XA^TY^{-1}AX)^{-1} - (I + XA^TY^{-1}AXY^{-1})^{-1}XA^T - S < 0 \end{aligned}$$

$$XX < I \quad \text{and} \quad YY < I.$$

Where the matrices A and S are given by

$$A = \begin{bmatrix} 1 & -1 \\ 0 & 2 \end{bmatrix}, \qquad S = \begin{bmatrix} 2 & 0 \\ 0 & 1 \end{bmatrix}.$$

This type of problem is easily handled by our numerical NCSDP solver.

This chapter is organized as follow: Section 4.2 presents the notation used, and some important results that are needed in later sections. Section 4.3 presents the theory behind our methodology, showing how we pose our feasibility optimization problem and the inner product optimization problem. Section 4.4 provides a tutorial example of our methodology using a Riccati feasibility problem. Section 4.5 extends the results presented in Section 4.3 for the more general multivariate case. Section 4.6 shows how simplifications rule can be applied to improve the evaluation time of the numerical solver. Section 4.7 provides numerical results of the NCSDP code implemented, comparing its performance with other available SDP solvers, and analyzing its behavior through a variety of control problems.

4.2 Notation

This section presents the notation used throughout this chapter and some important facts that will be needed. Although the presentation here is a bit redundant with the notation presented in other chapters, this chapter is intended to be self-contained so that it can be read independently of the other chapters.

4.2.1 Linear transformations on an Euclidean space

The n-dimensional Euclidean space, endowed with the usual dot product $\langle x, y\rangle = y^T x$, is denoted by $\mathbb{R}^n$. The standard basis of $\mathbb{R}^n$ is denoted by $\{e_1, e_2, \ldots, e_m\}$, in which e_i has 1 as ith component and 0's elsewhere. The space of $n \times m$ real matrices is denoted by $\mathbb{R}^{n\times m}$. The space of $n \times n$ symmetric matrices with real entries is denoted by $\mathbb{S}^n$. In this space, an ordering $\geq$ can be define as: given two matrices $A, B \in \mathbb{S}^n$, the order $A \geq B$ means that $A-B$ is a **positive semidefinite** matrix, in the sense that the inner product $\langle (A-B)x, x\rangle$ is nonnegative for any vector $x \in \mathbb{R}^n$. This space is denoted by $\mathbb{S}^n_+$. If strictly inequality is used in the definition, we obtain the space $\mathbb{S}^n_{++}$ of **positive definite** matrices.

The **Kronecker product** of two matrices A and B is denoted by $A \otimes B$. To define the **vec operation**, let us associate the vector $\mathrm{vec}(X) \in \mathbb{R}^{nm}$ with each matrix $X \in \mathbb{R}^{n\times m}$ by the rule

$$\mathrm{vec}(X) = [X_{11}, X_{21}, \ldots, X_{n1}, X_{12}, \ldots, X_{n2}, \ldots, X_{1m}, \ldots, X_{nm}]^T .$$

We provide two definitions for the **inner product** of two matrices A and B. The first

definition is given by

$$\begin{aligned}\langle A, B\rangle := \operatorname{Tr}\left\{AB^T\right\} &= \sum_{i,j} A_{ij}B_{ij} \\ &= \operatorname{vec}(B)^T \operatorname{vec}(A) = \langle \operatorname{vec}(A), \operatorname{vec}(B)\rangle.\end{aligned} \tag{4.2}$$

And the second definition, the symmetric case, is given by

$$\langle A, B\rangle_s := \operatorname{Tr}\left\{AB^T + BA^T\right\} \tag{4.3}$$

The corresponding induced norm, the Frobenius norm , is

$$\|X\| = \sqrt{\langle X, X\rangle}. \tag{4.4}$$

Let us define the canonical basis $E = \{E_{11}, E_{21}, \ldots, E_{n1}, E_{12}, \ldots, E_{nm}\}$, where each $E_{ij} \in \mathbb{R}^{n\times m}$ denotes the matrix with i, j entry 1 and all other entries 0. With this ordering, E_ℓ means the ℓth component of the set E. When we refer to the **matrix representation** of a linear transformation $L : \mathbb{R}^{p\times q} \to \mathbb{R}^{n\times m}$, we mean the representation relative to the canonical basis E and $\tilde{E}$ in $\mathbb{R}^{p\times q}$ and $\mathbb{R}^{n\times m}$ respectively.

$$\begin{array}{rccc} L: & \mathbb{R}^{p\times q} & \longrightarrow & \mathbb{R}^{n\times m} \\ & \Big\downarrow \text{vec} & & \Big\downarrow \text{vec} \\ \mathcal{M}: & \mathbb{R}^{pq} & \longrightarrow & \mathbb{R}^{nm} \end{array}$$

Figure 4.1: Isomorphism of the mapping $X \longrightarrow \operatorname{vec}(X)$.

With this notation, the matrix that represents a linear transformation $L : \mathbb{R}^{p\times q} \to \mathbb{R}^{n\times m}$ is the matrix $\mathcal{M} \in \mathbb{R}^{nm\times pq}$, as illustrated by Figure 4.1, such that

$$\mathcal{M} \operatorname{vec}(X) = \operatorname{vec}(L(X)) \tag{4.5}$$

for all $X \in \mathbb{R}^{p\times q}$ (Lemma 4.3.2 in Horn and Johnson (1999)). This representation is unique and depends solely on the map L. The existence and uniqueness is immediate from the fact that any linear operator on a finite dimensional space has a unique matrix representation relative to a given basis (Lemma 4.2.3). This can also be deduced from the fact that: (1) the map vec : $\mathbb{R}^{p\times q} \to \mathbb{R}^{pq}$ is an isomorphism; (2) any linear transformation $T : \mathbb{R}^{pq} \to \mathbb{R}^{nm}$ has a unique matrix representation relative to a given basis.

Suppose that $L : \mathbb{R}^{p\times q} \to \mathbb{R}^{n\times m}$ is a linear operator[1], then the unique **adjoint operator** $L^\star : \mathbb{R}^{n\times m} \to \mathbb{R}^{p\times q}$ satisfies

$$\langle L(X), Y\rangle = \langle X, L^\star(Y)\rangle \tag{4.6}$$

[1]The name operator, transformation and map are used interchangeably.

for all $X \in \mathbb{R}^{p\times q}$ and $Y \in \mathbb{R}^{n\times m}$. A map L is said to be **self-adjoint** if $L = L^\star$. It will be shown that a linear transformation L is self-adjoint if and only if the matrix $\mathcal{M}$ representing L is real and symmetric.

An example of a self-adjoint operator $L(X)$ is the map $X \to AXB + A^TXB^T : \mathbb{R}^{n\times n} \to \mathbb{R}^{n\times n}$, for given $A, B \in \mathbb{R}^{n\times n}$. The matrix representation for L is given by $\mathcal{M} = B^T \otimes A + B \otimes A^T$, where the symbol $\otimes$ stands for the Kronecker product. It should be emphasized that the map $L(X)$ being self-adjoint does not necessarily imply that the matrix[2] $L(X)$ is symmetric for a given matrix X.

To prove that $L(X) = AXB + A^TXB^T$ is a self-adjoint operator, one needs to verify that for any $X, Y \in \mathbb{R}^{n\times n}$, the adjoint relation (4.6) holds with $L(X) = L^\star(X)$. This is accomplished by the following manipulations:

$$\begin{aligned} \langle L(X), Y\rangle &= \mathrm{Tr}\left\{(AXB + A^TXB^T)Y^T\right\} \\ &= \mathrm{Tr}\left\{X(BY^TA + B^TY^TA^T)\right\} \\ &= \mathrm{Tr}\left\{XL^\star(Y)^T\right\} \\ &= \langle X, L^\star(Y)\rangle. \end{aligned} \tag{4.7}$$

The adjoint map $L^\star(Y)$ is therefore given by $Y \to AYB + A^TYB^T$, from which one promptly concludes that $L(X) = L^\star(X)$. Symmetry of the matrix representation $\mathcal{M}$ comes from the fact that the Kronecker product $(B^T \otimes A)^T$ equals $B \otimes A^T$. Many properties of functions on matrices can be found in Horn and Johnson (1999).

After this example, we should state and prove a simple but very useful result, Theorem 4.2.1, which says that if $L : \mathbb{R}^{p\times q} \to \mathbb{R}^{p\times q}$ is a self-adjoint linear transformation, then its matrix representation $\mathcal{M} \in \mathbb{R}^{pq\times pq}$ is real and symmetric. This theorem is a particular case of a more general result which holds for any self-adjoint linear transformation on finite-dimensional inner-product[3] spaces. This is a standard result available in many textbooks like MacLane and Birkhoff (1999) and Michel and Herget (1981).

Theorem 4.2.1 *Let $L : \mathbb{R}^{p\times q} \to \mathbb{R}^{p\times q}$ be a linear transformation. Then L is self-adjoint if and only if its matrix representation $\mathcal{M}$ is real and symmetric.*

Proof. Follows immediately from Proposition 4.2.2. ∎

[2] We do not use different symbols to distinguish a map $L(X)$ from its matrix value $L(X)$ for a given X, as done in the previous chapter 3.

[3] An inner-product space is sometimes called a pre-Hilbert space. If the space is also complete, then it is said to be a Hilbert space.

Proposition 4.2.2 *Let E and $\widetilde{E}$ be canonical basis in $\mathbb{R}^{p\times q}$ and $\mathbb{R}^{n\times m}$ respectively. The matrix representation (relative to E, $\widetilde{E}$) of each linear transformation*

$$L : \mathbb{R}^{p\times q} \to \mathbb{R}^{n\times m}$$

is the transpose of the matrix representation (relative to $\widetilde{E}$, E) of its adjoint map

$$L^\star : \mathbb{R}^{n\times m} \to \mathbb{R}^{p\times q}.$$

Proof. From Lemma 4.2.3, the matrix representation $\mathcal{M}$ of L given by $\mathcal{M}_{ij} = \langle L(E_j), \widetilde{E}_i\rangle$ satisfies Eq. (4.8). By an analogous argument, the matrix $\widetilde{\mathcal{M}}$, for the adjoint $L^\star$, is $\widetilde{\mathcal{M}}_{ji} = \langle L^\star(\widetilde{E}_i), E_j\rangle$. Using the symmetry property of the inner product and the adjoint relation (4.6), we obtain

$$\widetilde{\mathcal{M}}_{ji} = \langle L^\star(\widetilde{E}_i), E_j\rangle = \langle E_j, L^\star(\widetilde{E}_i)\rangle = \langle L(E_j), \widetilde{E}_i\rangle = \mathcal{M}_{ij}.$$

Hence, $\mathcal{M} = \widetilde{\mathcal{M}}^T$, as asserted. ∎

Lemma 4.2.3 *Let $L : \mathbb{R}^{p\times q} \to \mathbb{R}^{n\times m}$ be a linear transformation, and take E and $\widetilde{E}$ to be canonical basis in $\mathbb{R}^{p\times q}$ and $\mathbb{R}^{n\times m}$ respectively. Then the matrix representation $\mathcal{M}$ of L satisfies*

$$\mathcal{M}\,\mathrm{vec}(X) = \mathrm{vec}(L(X)) \tag{4.8}$$

for all $X \in \mathbb{R}^{p\times q}$, and it is uniquely determined by

$$\mathcal{M}_{ij} = \langle L(E_j), \widetilde{E}_i\rangle.$$

Proof. For $X \in \mathbb{R}^{p\times q}$, let $\{\zeta_j\}$ be the coordinates that represent X relative to the basis E, i.e., $X = \sum_j \zeta_j E_j$. By linearity of the vec operator and of the map L, we have

$$\mathrm{vec}(X) = \sum_j \zeta_j \,\mathrm{vec}(E_j) \quad \text{and} \quad \mathrm{vec}(L(X)) = \sum_j \zeta_j \,\mathrm{vec}(L(E_j)). \tag{4.9}$$

To find the matrix representation $\mathcal{M}$, Eq. (4.8) must hold for all $X \in \mathbb{R}^{p\times q}$. Substituting (4.9) in (4.8) gives

$$\begin{aligned}
\mathcal{M}\,\mathrm{vec}(X) &= \mathrm{vec}(L(X)) \\
\sum_j \zeta_j \mathcal{M}\,\mathrm{vec}(E_j) &= \sum_j \zeta_j \,\mathrm{vec}(L(E_j)) \\
\sum_j \zeta_j \,\mathrm{vec}(\widetilde{E}_i)^T \mathcal{M}\,\mathrm{vec}(E_j) &= \sum_j \zeta_j \,\mathrm{vec}(\widetilde{E}_i)^T \,\mathrm{vec}(L(E_j))
\end{aligned}$$

This choice of basis has the property that $E_j \xrightarrow{\text{vec}} e_j$. Thus, the above equations simplify to

$$\begin{aligned}\sum_j \zeta_j \mathcal{M}_{ij} &= \Big\langle \sum_j \zeta_j \operatorname{vec}(L(E_j)), \operatorname{vec}(\widetilde{E}_i) \Big\rangle \\ &= \Big\langle \sum_j \zeta_j L(E_j), \widetilde{E}_i \Big\rangle.\end{aligned}$$

Hence $\mathcal{M}_{ij} = \langle L(E_j), \widetilde{E}_i \rangle$ as asserted. ∎

4.2.2 Noncommutative symmetric rational functions

That our goal is to use symbolic computation to compute the gradient and the Hessian of the functions to be optimized, we abstract the notion of function on matrices to that of function on variables which are symbolic noncommutative elements. The presentation here is from Section 3.2.1 of Chapter 3.

What occurs in practice are functions F which are polynomial or rational in noncommutative variables (often referred to as indeterminate) with coefficient that are real numbers. **Noncommutative rational functions** of X are polynomials in X and in inverses of polynomials in X. Examples of noncommutative symmetric functions are

$$F(A, B, X) = AX + XA^T - \frac{3}{4} XBB^T X, \qquad X = X^T,$$

$$F(A, D, X, Y) = X^T AX + DYD^T + XYX^T, \qquad Y = Y^T \quad \text{and} \quad A = A^T, \tag{4.10}$$

and

$$F(A, D, E, X, Y) = A^T(XDX + X^T DX^T)A + E(XYX^T + X^T YX)E^T, \tag{4.11}$$

with $Y = Y^T$ and $D = D^T$.

It is also assumed that there is an involution on these rational functions which is denoted by the superscript ^{T}, and which will play the role of transpose later when we substitute matrices for the indeterminate.

Often we should think of some indeterminates as knowns and other indeterminates as unknowns and be concerned primarily about a function's properties with respect to unknowns. For example, in function (4.11) when we are mainly concerned about behavior such as convexity of F in X, Y we write $F(A, D, E, X, Y)$ simply as $F(X, Y)$. We call a noncommutative function $F(A, X)$ **symmetric** provided that $F(A, X)^T = F(A, X)$.

First directional derivatives

The first **directional derivative** of a noncommutative rational function $F(A, X)$ with respect to X in the direction δ_X is defined in the usual way

$$DF(X)[\delta_X] := \lim_{t\to 0} \frac{1}{t}\left(F(X + t\delta_X) - F(X)\right) = \frac{d}{dt}\, F(X + t\delta_X)\Big|_{t=0}$$

For example, the first directional derivative of $F(X)$ in (4.10) with respect to X along the direction δ_X is

$$D_X F(X, Y)[\delta_X] = \delta_X^T AX + X^T A\delta_X + \delta_X Y X^T + XY\delta_X^T.$$

It is easy to check that derivatives of symmetric noncommutative rational functions always have the form

$$DF(X)[\delta_X] = \text{sym}\left\{\sum_{i=1}^{k} A_i \delta_X B_i\right\}. \tag{4.12}$$

where the **sym operator**, defined as $\text{sym}[M] = M + M^T$, is used to make an expression symmetric. As an example, the above derivative can be written in the form (4.12), by defining $k = 1$ and

$$\begin{aligned} A_1 &= X^T A, \qquad & B_1 &= 1 \\ A_2 &= 1, & B_2 &= YX^T \end{aligned}$$

Second directional derivatives

The second directional derivative of a noncommutative rational function is obtained from the second order terms of a Taylor expansion of $F(X + t\delta_X)$ about $t = 0 \in \mathbb{R}$:

$$F(X + t\delta_X) = F(X) + DF(X)[\delta_X]\, t + D^2F(X)[\delta_X, \delta_X]\, t^2 + \cdots$$

Thus, the second directional derivative $D^2F(X)[\delta_X, \delta_X]$ of $F(X)$ is defined by

$$D^2F(X)[\delta_X, \delta_X] = \frac{d^2}{dt^2} F(X + t\delta_X)\Big|_{t=0}.$$

An analogous more general expression holds for more variables. For example, the second directional derivative of $F(X, Y)$ in (4.11) with respect to X along the direction δ_X is

$$D^2F(X, Y)[\delta_X, \delta_X] = 2\Big(A^T(\delta_X D\delta_X + \delta_X^T D\delta_X^T)A + E(\delta_X Y\delta_X^T + \delta_X^T Y\delta_X)E^T\Big). \tag{4.13}$$

One can easily show that the second directional derivative of a symmetric noncommutative rational function $F(X)$ with respect to one variable X in the direction δ_X has the form

$$D^2F(X)[\delta_X, \delta_X] = \operatorname{sym}\left\{\sum_{j=1}^{w_1} M_j \delta_X N_j \delta_X T_j + \sum_{j=1+w_1}^{w_2} M_j {\delta_X}^T N_j \delta_X T_j + \sum_{j=1+w_2}^{w_3} M_j \delta_X N_j {\delta_X}^T T_j\right\}. \quad (4.14)$$

As an example, the second directional derivative given in (4.13) can be expressed in the form (4.14) by defining $w_1 = 1$, $w_2 = 2$, $w_3 = 3$, and

$$\begin{aligned} M_1 &= 2A^T, & N_1 &= D, & T_1 &= A \\ M_2 &= 1/2E, & N_2 &= Y, & T_2 &= E^T \\ M_3 &= 1/2E, & N_3 &= Y, & T_3 &= E^T \end{aligned}$$

4.2.3 Equivalence between different notions of derivatives

We now provide very briefly some connections between different notions of derivatives. For a more complete exposition and proof see Graves (1935); Hildebrandt and Graves (1927); Luenberger (1969); Lusternik and Sobolev (1961); Ortega and Rheinboldt (2000).

The definition of the directional derivative of a noncommutative rational function $F(X)$ presented in the previous section did not assume any norm topology. However, in the derivatives to be taken for the barrier, it will be assumed a topology provided by the trace operator. Depending upon the choice of the inner product, one may have different interpretations for the derivatives. To provide the definition for the gradient and the Hessian map, we shall introduce the definitions of *Gateaux and Fréchet differentials*.

Definition 4.2.4 (Gateaux differential) *Let $\mathcal{V}$ and $\mathcal{W}$ be normed spaces. Let $F(X) : \mathcal{C} \subset \mathcal{V} \to \mathcal{W}$. If for some X in the interior of $\mathcal{C}$, and $\delta_X \in \mathcal{V}$, the limit*

$$\lim_{t \to 0} \frac{1}{t}\left[F(X + t\delta_X) - F(X)\right] = DF(X)[\delta_X]$$

exists, then $F(X)$ is said to have a **Gateaux differential** *at X in the direction δ_X, which we denote by $DF(X)[\delta_X]$.*

Definition 4.2.5 (Gateaux derivative) *If the transformation $DF(X)[\delta_X] : \mathcal{V} \times \mathcal{V} \to \mathcal{W}$ exists for all δ_X and is linear in δ_X, then $F(X)$ is said to be Gateaux-differentiable at X, and the* **Gateaux derivative** *$F'(X)$ satisfies*

$$\lim_{t \to 0} \frac{1}{t}\left\| F(X + t\delta_X) - F(X) - tF'(X)\delta_X \right\| = 0.$$

The map $F'(X)$ is a linear operator from $\mathcal{V}$ to $\mathcal{W}$, which depends, in general, on X. This map is unique and it follows that $DF(X)[\delta_X] = F'(X)\delta_X$.

In the most general case, the Gateaux differential of $F(X)$ may exist at X for all $\delta_X \in \mathcal{V}$, and yet $F(X)$ does not have a Gateaux derivative at X.

Definition 4.2.6 (Fréchet differential) *Let $F(X) : \mathcal{C} \subset \mathcal{V} \to \mathcal{W}$ with $\mathcal{V}$ and $\mathcal{W}$ normed spaces. Let X and δ_X be arbitrary elements in $\mathcal{V}$. Then the map $F(X)$ is Fréchet differentiable at X in the interior of $\mathcal{C}$, if there exists a map $DF(X)[\delta_X] : \mathcal{V} \times \mathcal{V} \to \mathcal{W}$ which is linear and continuous with respect to δ_X such that*

$$\lim_{\delta_X \to 0} \frac{1}{\|\delta_X\|} \big\| F(X + \delta_X) - F(X) - DF(X)[\delta_X] \big\| = 0.$$

This transformation can be written as $DF(X)[\delta_X] = F'(X)\delta_X$, where the linear operator $F'(X) : \mathcal{V} \to \mathcal{W}$ is the **Fréchet derivative** *of $F(X)$, which depends, in general, on X.*

Remark 4.2.7 *If the Fréchet differential exists, then the Gateaux differential exists, and they are equal.*

Remark 4.2.8 *If the Gateaux differential exists for all X in an open neighborhood of a point X^0 and if it is uniformly continuous in X and continuous in δ_X, then the Fréchet differential exists in this neighborhood, and they are equal.*

Remark 4.2.9 (Equivalence of the definitions) *For the type of rational matrix valued functions assumed in this thesis, any one of the above definitions of derivatives implies the other one.*

If the transformation $DF(X)[\delta_X]$ is a linear functional (in the direction δ_X), whose range lies in $\mathcal{W} \subset \mathbb{R}$, then it follows from the Riesz representation theorem[4] that given any inner product $\langle \cdot, \cdot \rangle$, this functional can be represented as

$$DF(X)[\delta_X] = \langle \mathbb{Q}(X), \delta_X \rangle, \tag{4.15}$$

with $\mathbb{Q}(X) : \mathcal{V} \to \mathcal{V}$ defined as the **gradient map**. Evidently, this representation will depend on the choice for the inner product.

[4]The Riesz representation theorem says that every continuous linear functional (similarly, every *sesquilinear form*) on a Hilbert space can be represented in terms of an inner product (Kreyszig (1989); Reed and Simon (2000)).

Following similar ideas, we can also define the second derivative $D^2F(X)[\delta_X, \delta_X]$ of the transformation $F(X) : \mathcal{C} \subset \mathcal{V} \to \mathcal{W}$ as the limit

$$\lim_{t\to 0} \frac{1}{t}\left[F'(X + t\delta_X) - F'(X)\right] = D^2F(X)[\delta_X, \delta_X],$$

provided it exists. If the transformation $D^2F(X)[\delta_X, \delta_X] : \mathcal{V} \times \mathcal{V} \times \mathcal{V} \to \mathcal{W}$ is a sesquilinear form in δ_X, whose range lies in $\mathcal{W} \subset \mathbb{R}$, then from the Riesz representation theorem[4], we know that given any inner product $\langle \cdot, \cdot \rangle$ the map $D^2F(X)[\delta_X, \delta_X]$ can be represented as

$$D^2F(X)[\delta_X, \delta_X] = \langle \mathbb{H}(X, \delta_X), \delta_X \rangle, \tag{4.16}$$

with $\mathbb{H}(X, \delta_X) : \mathcal{V} \times \mathcal{V} \to \mathcal{V}$ being the **Hessian map**, which is linear in the direction δ_X. Evidently, this representation depends on the choice of the inner product.

Remark 4.2.10 *Since ultimately one of the main concerns will be producing and solving a linear system of equations of the "form" $\mathbb{H}(\delta_X) = \mathbb{Q}$ for the update direction δ_X, the dependence on X is usually omitted in the notation for the gradient $\mathbb{Q}$ and for the Hessian map $\mathbb{H}(\delta_X)$.*

Lemma 4.2.11 *Let $F(X) : \mathbb{R}^{p\times q} \to \mathbb{S}^n$. Assume that the inner product $\langle \cdot, \cdot \rangle$ is the one given in* (4.2)*, then it follows that*

$$D^2F(X)[\delta_X, \delta_X] = \langle \mathbb{H}(\delta_X), \delta_X \rangle = \operatorname{vec}(\delta_X)^T \mathcal{H} \operatorname{vec}(\delta_X)$$

where $\mathbb{H}(\delta_X) : \mathbb{R}^{p\times q} \to \mathbb{R}^{p\times q}$ is the Hessian map and $\mathcal{H} \in \mathbb{R}^{pq}$ is the unique matrix representation for $\mathbb{H}(\delta_X)$.

Proof. It follows directly from (4.16) and (4.5) (or equivalently Lemma 4.2.3). ∎

Corollary 4.2.12 *An immediate consequence of the above Lemma 4.2.11 is: if*

$$D^2F(X)[\delta_X, \delta_X] \geq 0 \quad \textit{for all} \quad \delta_X,$$

then the Hessian matrix $\mathcal{H}$ is positive semidefinite. If strictly inequality is used, one finds that $\mathcal{H}$ is a positive definite matrix.

4.2.4 Preliminary results about the barrier function

This section presents some important facts concerning the log-det function. Let the barrier function $\Theta(X)$ be defined as:

$$\Theta(X) = \log\det F(X)^{-1} : \mathcal{G} \to \mathbb{R}, \tag{4.17}$$

with the domain $\mathcal{G}$ given by

$$\mathcal{G} = \{X \in \mathcal{V} : F(X) > 0\}$$

and $F(X)$ a self-adjoint noncommutative rational function. Then its directional derivative, along the direction $\delta_X \in \mathcal{V}$, is the linear form in δ_X given by

$$D\Theta(X)[\delta_X] = -\operatorname{Tr}\left\{F(X)^{-1}DF(X)[\delta_X]\right\}.$$

And its second directional derivative is the quadratic form in δ_X given by

$$\begin{aligned} D^2\Theta(X)[\delta_X,\delta_X] = &\operatorname{Tr}\left\{\left(F(X)^{-1}\,DF(X)[\delta_X]\right)^2\right\} \\ &- \operatorname{Tr}\left\{F(X)^{-1}D^2F(X)[\delta_X,\delta_X]\right\}. \end{aligned} \tag{4.18}$$

The proof is quite simple and follows by applying the definition of directional derivative and the following result provided in Horn and Johnson (1999):

$$\frac{d}{dt}\log\det A(t) = \operatorname{Tr}\left\{A(t)^{-1}\frac{d}{dt}A(t)\right\}.$$

In order to obtain the representation for the gradient and the Hessian map of the barrier $\Theta(X)$, using the definitions introduced in the previous Section 4.2.3, we need to specify an inner product. For this purpose, we take the inner product $\langle\cdot,\cdot\rangle_s$ given in (4.3). Thus the gradient $\mathbb{Q}$ is given by

$$D\Theta(X)[\delta_X] = \langle\mathbb{Q},\delta_X\rangle_s = \operatorname{Tr}\left\{\delta_X\mathbb{Q}^T + \mathbb{Q}\delta_X^T\right\},$$

and the Hessian map $\mathbb{H}(\delta_X)$ is obtained from

$$D^2\Theta(X)[\delta_X,\delta_X] = \langle\mathbb{H}(\delta_X),\delta_X\rangle_s = \operatorname{Tr}\left\{\delta_X\mathbb{H}(\delta_X)^T + \mathbb{H}(\delta_X)\delta_X^T\right\}.$$

Obtaining the algebraic linear system of equation

The linear system of equations (which basically has the form $\mathbb{H}(\delta_X) = \mathbb{Q}$) that will provide the update direction δ_X is obtained by setting the directional derivative of a second-order approximation of the barrier function $\Theta(X)$ to zero:

$$0 = D\Big(D\Theta(X)[\delta_X] + \frac{1}{2}D^2\Theta(X)[\delta_X,\delta_X]\Big)[\delta_V], \qquad \forall\delta_V. \tag{4.19}$$

Using the notation just introduced for the gradient $\mathbb{Q}$ and the Hessian $\mathbb{H}(\delta_X)$, the above equation becomes

$$0 = D\Big(\operatorname{Tr}\left\{\delta_X(1/2\mathbb{H}(\delta_X) - \mathbb{Q})^T + (1/2\mathbb{H}(\delta_X) - \mathbb{Q})\delta_X^T\right\}\Big)[\delta_V], \qquad \forall \delta_V.$$

Which, after taking the directional derivative along the direction δ_V, reduces to

$$0 = \frac{1}{2}\operatorname{Tr}\left\{\delta_V(\mathbb{H}(\delta_X) - 2\mathbb{Q})^T + (\mathbb{H}(\delta_X) - 2\mathbb{Q})\delta_V^T + \delta_X\mathbb{H}(\delta_V)^T + \mathbb{H}(\delta_V)\delta_X^T\right\}, \qquad \forall \delta_V.$$

Since the Hessian is a self-adjoint map, it follows that

$$\operatorname{Tr}\left\{\delta_X\mathbb{H}(\delta_V)^T\right\} = \langle \delta_X, \mathbb{H}(\delta_V)\rangle = \langle \mathbb{H}(\delta_X), \delta_V\rangle = \operatorname{Tr}\left\{\mathbb{H}(\delta_X)\delta_V^T\right\}.$$

And consequently the optimality condition reduces to,

$$0 = \operatorname{Tr}\left\{\delta_V(\mathbb{H}(\delta_X) - \mathbb{Q})^T + (\mathbb{H}(\delta_X) - \mathbb{Q})\delta_V^T\right\}, \quad \text{for all } \delta_V.$$

Equivalently

$$0 = \langle \mathbb{H}(\delta_X) - \mathbb{Q}, \delta_V\rangle_s, \quad \text{for all } \delta_V.$$

It is now clear that there are two "different" ways to produce the linear system of equations, i.e., to determine the Hessian map $\mathbb{H}(\delta_X)$ and the gradient term $\mathbb{Q}$:

1. One can take the directional derivatives of the Taylor expansion (4.19), and manipulate the final formula to be expressed in the form

   $$\operatorname{Tr}\left\{\delta_V(\mathbb{H}(\delta_X) - \mathbb{Q})^T + (\mathbb{H}(\delta_X) - \mathbb{Q})\delta_V^T\right\},$$

   determining in this way $\mathbb{Q}$ and $\mathbb{H}(\delta_X)$.

2. Or, one can just determine $\mathbb{Q}$ and $\mathbb{H}(\delta_X)$ respectively from the definitions provided in (4.15) and (4.16).

In this chapter, we have determined $\mathbb{Q}$ and $\mathbb{H}(\delta_X)$ directly by taking all the necessary directional derivatives.

Convexity of the barrier

We now present another important fact concerning this barrier. Let the map $F(X) : \mathbb{R}^{p\times q} \to \mathbb{S}^n$ be self-adjoint, and assume that the set

$$\mathcal{G} = \{X \in \mathbb{R}^{p\times q} : F(X) > 0\}$$

is nonempty and bounded, then if the barrier

$$\Theta(X) = \log\det F(X)^{-1} : \mathcal{G} \to \mathbb{R}$$

is real analytic and strictly convex for all $X \in \mathcal{G}$, it has a unique minimizer

$$X_\Theta^* = \underset{X}{\operatorname{argmin}} \{\Theta(X) : X \in \mathcal{G}\}.$$

We refer to this minimizer X_Θ^* as the *analytic center* of the matrix inequality $F(X) > 0$.

Since the barrier $\Theta(X)$ is the composition of the log-det function with a rational matrix valued function $F(X)$, which is well defined on the domain $\mathcal{G}$, it follows that the barrier $\Theta(X)$ is real analytic (i.e., in each small region it has a convergent power series expansion). It is easy to show that the barrier is convex if $F(X)$ is a concave function. To see this, recall the expression for $D^2\Theta(X)[\delta_X, \delta_X]$ given in (4.18), and note that

$$\operatorname{Tr}\left\{\left(F(X)^{-1}DF(X)[\delta_X]\right)^2\right\} \geq 0 \qquad \text{for all} \quad \delta_X \in R_{p\times q}$$

and that (assuming $F(X)$ concave)

$$-\operatorname{Tr}\left\{F(X)^{-1}D^2F(X)[\delta_X, \delta_X]\right\} \geq 0 \qquad \text{for all} \quad \delta_X \in R_{p\times q}.$$

Thus

$$D^2\Theta(X)[\delta_X, \delta_X] \geq 0 \qquad \text{for all} \quad \delta_X \in R_{p\times q}.$$

Which from Corollary 4.2.12 implies that the Hessian matrix $\mathcal{H}$ of the barrier function $\Theta(X)$ is a positive semidefinite matrix.

For the specific case where $F(X)$ is an LMI, having the representation $F(x) = F_0 + \sum_{i=1}^m F_i x_i$, with $F_0 \in S_n$ and $F_i \in S_n$ for $i = 1, \ldots, m$, a necessary and sufficient condition for the log-det barrier to be strictly convex is that the matrices F_i be linearly independent (Boyd et al. (1994)). However, for the more general case, where $F(X) : \mathbb{R}^{p\times q} \to \mathbb{S}^n$ is a matrix function, the characterization is more elaborate. Naturally, strictly convexity of the barrier is equivalent to the Hessian map $\mathbb{H}(\delta_X)$ being invertible. However, imposing stronger assumptions on the function $F(X)$, such as being strictly concave on its domain or its first directional derivative $DF(X)[\delta_X]$ being an invertible map, the barrier will be strictly convex.

4.3 Convex Optimization over Matrix Functions

The numerical optimization solver for matrix functions is now presented. We shall demonstrate our approach for two classes of problem: the eigenvalue minimization problem and the more general inner product minimization problem.

In this section, the presentation is limited to the univariate case, which only considers functions of a single variable. Later, in Section 4.5, the results are generalized to the multivariate case, which considers functions of several variables.

4.3.1 The eigenvalue minimization problem

Let $\mathcal{C}$ be a bounded convex domain in $\mathbb{R}^{p\times q}$. For each $i = 1, \ldots, m$, let the map $F_i(X) : \mathcal{C} \to \mathbb{S}^{n_i}$ be concave. Then, the *eigenvalue optimization problem* can be posed as:

find α^*, if one exists, such that

$$\alpha^* = \min \{\alpha : (X, \alpha) \in \text{closure}(\mathcal{G})\}, \tag{4.20}$$

where the feasibility set[5] $\mathcal{G}$ is the convex domain given by

$$\mathcal{G} = \Big\{(X, \alpha) \in \mathcal{C} \times \mathbb{R} : \alpha I - F_1(X) > 0,\ F_2(X) > 0\ , \ldots,\ F_m(X) > 0\ \Big\}.$$

In this setup, the maximum eigenvalue of $F_1(X)$ is minimized inside the convex region provided by the set of matrix inequalities $F_j(X) > 0$ for $j = 2, \ldots, m$, as a function of a single variable X. Section 4.5 expands the idea to the multivariate case, where the F_i can be functions of several variables $X_1, \ldots, X_r$.

4.3.2 The inner product minimization problem

We now pose a variation on the previous optimization problem. Let $\mathcal{C}$ be a bounded convex domain in $\mathbb{R}^{p\times q}$. For each $i = 1, \ldots, m$, let the map $F_i(X) : \mathcal{C} \to \mathbb{S}^{n_i}$ be concave. Then, our main problem, the *inner product minimization problem*, can be posed as:

find t^*, if one exists, such that

$$t^* = \min \{\text{Tr}\{X\} : X \in \text{closure}(\mathcal{G})\}, \tag{4.21}$$

where the feasibility set $\mathcal{G}$ is the convex domain given by

$$\mathcal{G} = \Big\{X \in \mathcal{C} : F_i(X) > 0,\ i = 1, \ldots, m\Big\}.$$

This type of problem incorporates the above eigenvalue minimization problem as a particular case, where, instead of a univariate problem in X, we would have an optimization problem in

[5]The notation closure($\mathcal{G}$) means the closure of $\mathcal{G}$.

the unknowns X and $\alpha \in \mathbb{R}$, and we also would have to redefine appropriately the constraint $F_1(\alpha, X) := \alpha I - F_1(X)$; in addition, in place of $\mathrm{Tr}\{X\}$ we would have $\mathrm{Tr}\{\alpha\} = \alpha$.

The inner product problem is solved using the *method of centers* presented in the next Section 4.3.3.

4.3.3 Method of centers

The presentation here has its roots mainly in Boyd and El Ghaoui (1993). We present the method of centers (also called Huard's method of centers) and show how to use it in order to solve the two problems stated above in (4.20) and (4.21). Other important references on the method of centers are Huard (1967); Lieu and Huard (1965); Nesterov and Nemirovskii (1994).

A more general formulation of the minimization problem

Let $\mathcal{C}$ be a bounded convex domain[6] in $\mathbb{R}^{p\times q}$. Let the function $f(X) : \mathcal{C} \to \mathbb{R}$ be convex, and for each $i = 1, \ldots, m$ the map $F_i : \mathcal{C} \to \mathbb{S}^{n_i}$ be concave. Then, the constrained optimization problem (COP) that we are interested in can be posed as:

find f^{opt}*, if one exists, such that*

$$f^{\mathrm{opt}} = \min\{f(X) : X \in \mathrm{closure}(\mathcal{G})\}, \tag{COP}$$

where the feasibility domain $\mathcal{G}$ is given by

$$\mathcal{G} = \Big\{X \in \mathcal{C} : F_i(X) > 0,\ i = 1, \ldots, m\Big\}.$$

It is well known that, without the loss of generality, we can assume $f(X)$ to be linear. Since it is always possible to define a new cost function $f(X, \xi) = \xi$ and add the constraint $\xi - f(X) > 0$ to the domain $\mathcal{G}$.

The method of centers

The idea behind the method is to replace the above *constrained problem* (COP) by a sequence of *unconstrained convex minimization problems* whose solutions eventually tend

[6] The set $\mathcal{C}$ imposes some regularity assumption on the set $\mathcal{G}$. Actually one can relax this assumption and let $\mathcal{C}$ be the entire Euclidean space, provided that the closure of $\{X : F_i(X) > 0,\ i = 1, \ldots, m\}$ is a compact set.

to the set of optimal solutions of (COP). This approach is in the context of *interior penalty methods*, which was described in the classical monograph of Fiacco and McCormick (1990). For a more practical presentation of optimization methods see Gill et al. (1999).

This sequence of unconstrained problems has to incorporate the inequality constraint imposed by the functions $F_i(X)$ in the sense that its solutions have to always be feasible interior points for (COP). In order to accomplish this, one needs to define a *barrier function* for the feasibility domain $\mathcal{G}$. This **barrier function**, denoted by $\Theta(X)$, has to be a smooth strongly[7] convex function such that $\Theta(X) \to \infty$ for points converging to the boundary of the set $\mathcal{G}$. A usual barrier[8] is the one given by

$$\Theta(X) = -\sum_{i=1}^{m} \log\det F_i(X) : \mathcal{G} \to \mathbb{R}.$$

With the barrier $\Theta(X)$ defined in this way, the original problem (COP) could be approximated by a family of unconstrained problems of the form

$$X^*(\gamma) = \operatorname{argmin}\left\{\gamma f(X) + \Theta(X) : X \in \mathcal{G}\right\}, \tag{4.22}$$

where $\gamma > 0$ is a penalty parameter. Under some mild conditions, the solution $X^*(\gamma)$ of (4.22) approaches the set of optimal solutions of (COP) as $\gamma \to \infty$. This technique belongs to the well known class of *barrier methods*. The method we are interested in is not quite yet this one, instead of the above parameterization, the method of centers is based on the following family of unconstrained minimizations

$$\min\ \Upsilon_\gamma(X) + \Theta(X),$$

with $\gamma > f(X)$, and the barrier function Υ_γ given by

$$\begin{gathered}\Upsilon_\gamma(X) = \zeta \log\left(1/(\gamma - f(X)\right) : \mathcal{G}_\gamma \to \mathbb{R}, \qquad \zeta \geq 1 \\ \mathcal{G}_\gamma = \left\{X \in \mathcal{G} : f(X) < \gamma\right\}.\end{gathered} \tag{4.23}$$

It follows therefore, that the original constrained optimization problem (COP) can be approximated by a sequence of convex unconstrained optimization problems of the form

$$X^*(\gamma) = \operatorname{argmin}\left\{\phi_\gamma(X) : X \in \mathcal{G}_\gamma\right\}, \tag{UOP}$$

[7] A function $f(X)$ is said to be strongly convex, provided there exists a constant k such that its Hessian map $\mathbb{H}(\delta_X)$ satisfies $\langle \mathbb{H}(\delta_X), \delta_X \rangle \geq k\|\delta_X\|$ for all δ_X.

[8] Section 4.2.4 presents some important properties about this function.

with the **unconstrained auxiliary potential function** $\phi_\gamma(X)$ given by

$$\phi_\gamma(X) = \zeta \log\left(1/(\gamma - f(X)\right) - \sum_{i=1}^{m} \log\det F_i(X) : \mathcal{G}_\gamma \to \mathbb{R}, \qquad \zeta \geq 1.$$

The decrease of the parameter γ has to be done in such a way that the method maintains feasibility at each iteration and that the sequence $\{\gamma^k\}$ is guaranteed to converge to f^{opt} (the minimum values of the objective function). The formula for updating γ, at some iteration k, is given by

$$\gamma^{k+1} = (1-\theta)f(X^k) + \theta\gamma^k, \qquad 0 < \theta < 1. \tag{4.24}$$

Under mild hypotheses (see Section 4.2.4), for fixed $\gamma > f^{\text{opt}}$, the *analytic center* $X^*(\gamma)$ of (UOP) is well defined and unique. It is evident that for a decreasing sequence of parameters $\{\gamma^k\}$ the corresponding sequence of minimizers $\{X^*(\gamma^k)\}$ form a path, the so called *path of center*. It can be shown that this curve is analytic and has a limit as $\gamma \to f^{\text{opt}}$. Usually the most desired result is the $\lim_{k\to\infty} X^*(\gamma^k) \to X^{\text{opt}}$, a solution of (COP). A weaker result, but one that is frequently useful, is the $\lim_{k\to\infty} f(X^*(\gamma^k)) \to f^{\text{opt}}$. See Boyd and El Ghaoui (1993); Fiacco and McCormick (1990).

Let X^k denote $X^*(\gamma^k)$. Using these facts, one possible algorithm based on the method of centers can be described by

Algorithm 4.3.1 *Method of centers.*
Fix θ such that $0 < \theta < 1$;
Choose feasible X^0 and γ^0 such that $X^0 \in \mathcal{G}_{\gamma^0}$;
$k \leftarrow 0$;
while not converged **do**
    $\gamma^{k+1} \leftarrow (1-\theta)f(X^k) + \theta\gamma^k$;
    Solve $X^{k+1} = \text{argmin}\left\{\phi_{\gamma^{k+1}}(X^k) : X^k \in \mathcal{G}_{\gamma^{k+1}}\right\}$;
    $k \leftarrow k+1$;
end while

There are two important comments concerning this algorithm:

1. The bound γ^{k+1} used in the determination of the analytic center of $\phi_{\gamma^{k+1}}(X^k)$, is given by the formula

$$\gamma^{k+1} = (1-\theta)f(X^k) + \theta\gamma^k,$$

never produces infeasible starting points X^k and γ^{k+1}, since

$$\gamma^{k+1} - f(X^k) = \theta(\gamma^k - f(X^k)) > 0,$$

and consequently $f(X^k) < \gamma^{k+1}$, thereby satisfying the feasibility set $\mathcal{G}_{\gamma^{k+1}}$ given in (4.23).

2. Evidently, the expensive part of the algorithm is the *inner loop*, the part that computes the analytic center using Newton's method:

$$X^{k+1} = \operatorname{argmin}\left\{\phi_{\gamma^{k+1}}(X^k) : X^k \in \mathcal{G}_{\gamma^{k+1}}\right\}.$$

This is the scope of Section 4.3.5. The book by Nesterov and Nemirovskii (1994) provides a complete convergence analysis of Newton's methods for a general class of *self-concordant* barrier function. Another fundamental source is the classical monograph of Ortega and Rheinboldt (2000).

From now on, we take the cost function $f(X)$ to be $\operatorname{Tr}\{X\}$. This leads to the inner product minimization problem (4.21). For this cost function, and assuming that the constraint $F(X)$ is linear in X, we give a simple proof of convergence. This proof is basically a copy of the proof presented in Boyd et al. (1994) for the LMI case (a similar proof for LMIs is also available in Colaneri et al. (1997)). First, let us characterize the analytic center, the point $X^*(\gamma)$. For simplicity of exposition, assume that $m = 1$ and $\zeta = 1$ (the generalization for $m > 1$ is immediate). The unconstrained potential function becomes:

$$\phi_\gamma(X) = \log\left(1/(\gamma - \operatorname{Tr}\{X\}\right) - \log\det F(X).$$

The point $X^*(\gamma)$ is characterized by setting the derivative of $\phi_\gamma(X)$ to zero. Using the derivatives given in Section 4.2.4 and Eq. (4.36), the optimality condition is given by

$$-\operatorname{Tr}\left\{F(X^*(\gamma))^{-1} DF(X^*(\gamma))[\delta_X]\right\} + (\gamma - \operatorname{Tr}\{X^*(\gamma)\})^{-1} \operatorname{Tr}\{\delta_X\} = 0, \quad \forall \delta_X.$$

To proceed, assume the kth iteration and substitute $X^k - X^{\text{opt}}$ for δ_X in the above equation. This gives

$$\operatorname{Tr}\left\{F(X^k)^{-1}\left(F(X^k) - F(X^{\text{opt}})\right)\right\} = \left(\gamma^k - \operatorname{Tr}\left\{X^k\right\}\right)^{-1} \operatorname{Tr}\left\{X^k - X^{\text{opt}}\right\}.$$

Since $\operatorname{Tr}\left\{F(X^k)^{-1} F(X^{\text{opt}})\right\} \geq 0$, we conclude that

$$n \geq \frac{1}{\gamma^k - \operatorname{Tr}\{X^k\}}\left(\operatorname{Tr}\left\{X^k\right\} - f^{\text{opt}}\right), \tag{4.25}$$

where n equals the dimension of the range of $F(X)$, and $f^{\text{opt}} = \text{Tr}\left\{X^{\text{opt}}\right\}$, which hold true at the solution. From (4.24), it follows that

$$f(X^k) := \text{Tr}\left\{X^k\right\} = (\gamma^{k+1} - \theta\gamma^k)/(1-\theta).$$

Replacing the expression for $\text{Tr}\left\{X^k\right\}$ in (4.25) yields

$$\left(\gamma^{k+1} - f^{\text{opt}}\right) \leq \frac{n+\theta}{n+1}\left(\gamma^k - f^{\text{opt}}\right),$$

which naturally is the result in Boyd et al. (1994). This last equation shows that the method converges at least geometrically whenever $F(X)$ is linear in X. Moreover, the stopping criteria

$$\gamma^k - \text{Tr}\left\{X^k\right\} < \epsilon/n$$

guarantees (from (4.25)) that the solution is found within the precision $\epsilon > 0$ imposed by the designer. Suggestions for improving the speed of this method are provided in Boyd et al. (1994); Colaneri et al. (1997); Roubi (2001).

4.3.4 Feasibility problem

It will be necessary to find feasible starting points X^0 and γ^0 to be used in the algorithm 4.3.1. This is a *feasibility problem* that can be solved by the method of centers. The idea is simple, and it suffices to solve the following convex minimization problem

$$\alpha^* = \min\{\alpha : (X,\alpha) \in \text{closure}(\mathcal{G})\}, \tag{FP}$$

where the feasibility set $\mathcal{G}$ is the convex domain given by

$$\mathcal{G} = \left\{(X,\alpha) \in \mathcal{C} \times \mathbb{R} : F_i(X) + \alpha\, I_{n_i} > 0,\ i = 1,\ldots,m\right\},$$

with α being a scalar and each I_{n_i} being the identity matrix of dimension n_i.

To apply the method of centers to solve the above feasibility problem (FP), an initial feasible guess has also to be provided, however, for this type of problem, this guess is trivially obtained by choosing X^0 to be any matrix in $\mathcal{C}$ and by setting $\gamma^0 > \alpha^0 > \max_i \|F_i(X^0)\|_2$. If the solution α^* is negative, then the corresponding minimizer X^* has the property that $F_i(X^*) > 0$ for each $i = 1,\ldots,m$. In practice, the algorithm can stop as soon as the objective α^k is less than zero at some iteration k. However, if the scalar γ^k is positive for all iterations, then the sequence $\{\gamma^k\}$ converges to some scalar $\gamma^* > 0$, and consequently the problem is infeasible and there is no X such that $F_i(X) > 0$ for all $i = 1,\ldots,m$.

Remark 4.3.2 *An example of this technique is presented in Section 4.4, where the goal is to find a feasible solution to the matrix Riccati inequality, that is, to find a symmetric matrix X such that*

$$AX + XA^T - XRX + Q > 0.$$

This is a standard expression that appears frequently in many control applications.

4.3.5 Solving for the analytic center

The original convex optimization problem (COP) has now been replaced by a sequence of unconstrained convex minimization problems of the form (UOP) for a decreasing sequence of scalars $\{\gamma^k\}$ provided by formula (4.24). In other words, the problem reduces to finding update directions which leads toward the central path for fixed values of γ. To find those directions, Newton's method is applied by minimizing an approximation, the second-order Taylor series expansion, of the potential function $\phi_\gamma(X)$. In a vague sense, these procedures can be summarized as follows:

1. Compute the second-order Taylor expansion of $\phi_\gamma(X + \delta_X)$ in some direction δ_X

$$\phi_\gamma(X) + D\phi_\gamma(X)[\delta_X] + \frac{1}{2}D^2\phi_\gamma(X)[\delta_X, \delta_X].$$

2. The Newton step δ_X^* must satisfy the necessary optimality conditions for the following quadratic minimization problem

$$\delta_X^* \in \operatorname{argmin}\left(D\phi_\gamma(X)[\delta_X] + \frac{1}{2}D^2\phi_\gamma(X)[\delta_X, \delta_X]\right).$$

3. This first-order necessary optimality condition is algebraically given by

$$0 = D\Big[D\phi_\gamma(X)[\delta_X] + \frac{1}{2}D^2\phi_\gamma(X)[\delta_X, \delta_X]\Big][\delta_V], \qquad \forall \delta_V. \tag{4.26}$$

4. Finally, find a Newton update δ_X^* satisfying Eq. (4.26) for all δ_V.

Before presenting the main result of this chapter, Theorem 4.3.5, which concerns the determination of the update direction as the solution of the algebraic condition (4.26), given in step 3, some preliminary facts are now introduced.

Even though the goal is to determine the update direction δ_X for the general multivariate case, we first present the univariate case where the barrier

$$\Theta(X) = -\log\det F(X)$$

depends on a single constraint $F(X)$. Later, Section 4.5 will show how to expand the idea to handle multiple constraints $F_i(X)$, and the more general setup where each constraint $F_i(X_1, \ldots, X_r)$ can be a function of several variables $X_1, \ldots, X_r$. Thus, the unconstrained auxiliary potential function $\phi_\gamma(X)$ is given by:

$$\phi_\gamma(X) = \zeta \log\left(1/(\gamma - f(X)\right) - \log\det F(X),$$

where the map $F(X)$ is concave, and $f(X) : \mathbb{R}^{p\times q} \to \mathbb{R}$ is the trace operator, $f(X) = \operatorname{Tr}\{X\}$. Later, in Section 4.4, we will provide the derivation for the eigenvalue minimization problem, by means of a tutorial example.

Since we shall be taking derivatives of the potential function $\phi_\gamma(X)$ using symbolic computation, the $F(X)$ needs to be visualized as being a function on a symbolic unknown X rather than on a matrix of given dimension. Thus, from now on, $F(X)$ is a noncommutative rational function of the unknown X. The same idea would extends accordingly if $F(X_1, \ldots, X_r)$ were a multivariate function on a tuple r of matrices $X_1, \ldots, X_r$. The dependence of γ in the notation is also suppressed, since γ is just a constant and does not play any role in subsequent derivations. We also abbreviate $F(X)^{-1}$ by just F^{-1}.

To obtain the update direction δ_X, we have to take directional derivatives of the potential $\phi_\gamma(X)$, however, to simplify the exposition, it is useful to split this potential into two parts: the cost term $\Upsilon_\gamma(X)$ and the constraint term $\Theta(X)$, given respectively by

$$\Upsilon_\gamma(X) = \zeta \log\left(1/(\gamma - \operatorname{Tr}\{X\}\right), \qquad \Theta(X) = -\log\det F(X). \tag{4.27}$$

Thus, let us first assume that the potential function contains only the constraint term $\Theta(X)$ and is given by

$$\phi(X) = -\log\det F(X). \tag{4.28}$$

For this "simplified" potential, Lemma 4.3.3 below will provide the formulas for the update direction δ_X. After proving Lemma 4.3.3, the next step will be to incorporate the cost term $\Upsilon_\gamma(X)$ into the potential function $\phi(X)$, enabling us to prove a more general result, Proposition 4.3.4.

Lemma 4.3.3 *Let $\mathcal{V}$ be a subspace of $\mathbb{R}^{p\times q}$ and $\mathcal{C}$ be a convex domain in $\mathcal{V}$. Let the map $F(X) : \mathcal{C} \to \mathbb{S}^n$ be concave. Consider the following potential function*

$$\phi(X) = -\log\det F(X) : \mathcal{G} \to \mathbb{R},$$

where the feasibility domain $\mathcal{G}$ is given by

$$\mathcal{G} = \Big\{ X \in \mathcal{C} \subset \mathcal{V} : F(X) > 0 \Big\}.$$

Then the update direction δ_X^* toward the central path for the above potential satisfies the following symbolically computable algebraic linear equation:

$$\mathrm{Tr}\left\{ \delta_V (\mathbb{H}(\delta_X) - \mathbb{Q})^T + (\mathbb{H}(\delta_X) - \mathbb{Q}) \delta_V^T \right\} = 0, \quad \textit{for all } \delta_V \in \mathcal{V}, \tag{4.29}$$

or equivalently

$$\langle (\mathbb{H}(\delta_X) - \mathbb{Q}), \delta_V \rangle_s = 0, \quad \textit{for all } \delta_V \in \mathcal{V},$$

where $\mathbb{H}(\delta_X)$ is linear in δ_X and $\mathbb{Q}$ is an independent term that does not contain δ_X. Moreover, $\mathbb{Q}$ and $\mathbb{H}(\delta_X)$ are given by

$$\mathbb{Q} = \sum_{i=1}^{k} A_i^T F(X)^{-1} B_i^T,$$

and

$$\begin{aligned}
\mathbb{H}(\delta_X) = & \sum_{i=1}^{k} \sum_{j=1}^{k} A_i^T F^{-1} A_j \delta_X B_j F^{-1} B_i^T + \sum_{i=1}^{k} \sum_{j=1}^{k} A_i^T F^{-1} B_j^T \delta_X^T A_j^T F^{-1} B_i^T \\
& - \frac{1}{2} \sum_{j=1}^{w_1} N_j^T \delta_X^T M_j^T F^{-1} T_j^T + M_j^T F^{-1} T_j^T \delta_X^T N_j^T \\
& - \frac{1}{2} \sum_{j=1+w_1}^{w_2} N_j \delta_X T_j F^{-1} M_j + N_j^T \delta_X M_j^T F^{-1} T_j^T \\
& - \frac{1}{2} \sum_{j=1+w_2}^{w_3} T_j F^{-1} M_j \delta_X N_j + M_j^T F^{-1} T_j^T \delta_X N_j^T
\end{aligned}$$

where the terms A, B, M, N, T are obtained from the first and second directional derivatives of $F(X)$ which have the form[9]

$$DF(X)[\delta_X] = \mathrm{sym}\left\{ \sum_{i=1}^{k} A_i \delta_X B_i \right\}, \tag{4.30}$$

and

$$\begin{aligned}
& D^2F(X)[\delta_X, \delta_X] = \\
& \quad \mathrm{sym}\left\{ \sum_{j=1}^{w_1} M_j \delta_X N_j \delta_X T_j + \sum_{j=1+w_1}^{w_2} M_j \delta_X{}^T N_j \delta_X T_j + \sum_{j=1+w_2}^{w_3} M_j \delta_X N_j \delta_X{}^T T_j \right\}.
\end{aligned} \tag{4.31}$$

[9] *Recall the expressions* (4.12) *and* (4.14) *in Section 4.2.2.*

Proof of Lemma 4.3.3

Proof. To start, let us consider the second-order Taylor expansion of $\phi(X)$. To compute the quadratic approximation of $\phi(X)$, we take δ_X to be the update directions for X, so that the series expansion of $\phi(X)$ up to the second term is given by

$$\phi(X) + D\phi(X)[\delta_X] + \frac{1}{2}D^2\phi(X)[\delta_X, \delta_X] + \cdots$$

Now, we are ready to write down the requisites which will provide the update direction toward the analytic centers. Recall that the Newton step δ_X^* has to satisfy the first-order necessary optimality conditions for the following quadratic minimization problem

$$\delta_X^* \in \operatorname{argmin}\left(D\phi(X)[\delta_X] + \frac{1}{2}D^2\phi(X)[\delta_X, \delta_X]\right).$$

Which is equivalently described by the following algebraic equation

$$0 = D\Big(D\phi(X)[\delta_X] + \frac{1}{2}D^2\phi(X)[\delta_X, \delta_X]\Big)[\delta_V], \qquad \forall \delta_V. \tag{4.32}$$

Therefore, we will be taking directional derivatives along the direction δ_V of the potential $\phi(X)$ as a function of δ_X. Recalling Section 4.2.4, the first directional derivative of

$$\phi(X) = -\log\det F(X),$$

with respect to X along the direction δ_X has the linear form in δ_X given by

$$D\phi(X)[\delta_X] = -\operatorname{Tr}\left\{F^{-1}DF(X)[\delta_X]\right\}, \tag{4.33}$$

and the second directional derivative has the quadratic form in δ_X given by

$$\begin{aligned} D^2\phi(X)[\delta_X, \delta_X] = &\operatorname{Tr}\left\{\left(F^{-1}\,DF(X)[\delta_X]\right)^2\right\} \\ &- \operatorname{Tr}\left\{F^{-1}D^2F(X)[\delta_X, \delta_X]\right\}. \end{aligned} \tag{4.34}$$

In order to compute (4.32) let us apply separately the directional derivative in each one of the terms:

$$D\Big(D\phi(X)[\delta_X]\Big)[\delta_V] \quad \text{and} \quad D\Big(\frac{1}{2}D^2\phi(X)[\delta_X, \delta_X]\Big)[\delta_V].$$

Substituting the expressions given in (4.30) and (4.31) into (4.33) and (4.34), the first directional derivative is given by

$$\begin{aligned} D\phi(X)[\delta_X] &= -\operatorname{Tr}\left\{F^{-1}\operatorname{sym}\left\{\sum_{i=1}^{k} A_i\delta_X B_i\right\}\right\} \\ &= -\operatorname{Tr}\left\{\operatorname{sym}\left\{\delta_X\sum_{i=1}^{k} B_iF^{-1}A_i\right\}\right\}, \end{aligned}$$

and the second directional derivative is

$$\begin{aligned} D^2\phi(X)[\delta_X,\delta_X] = \operatorname{Tr}&\left\{\left(F^{-1}\operatorname{sym}\left\{\sum_{i=1}^{k} A_i\delta_X B_i\right\}\right)^2\right\} \\ &- \operatorname{Tr}\left\{F^{-1}\operatorname{sym}\left\{\sum_{j=1}^{w_1} M_j\delta_X N_j\delta_X T_j\right\}\right\} \\ &- \operatorname{Tr}\left\{F^{-1}\operatorname{sym}\left\{\sum_{j=1+w_1}^{w_2} M_j{\delta_X}^T N_j\delta_X T_j\right\}\right\} \\ &- \operatorname{Tr}\left\{F^{-1}\operatorname{sym}\left\{\sum_{j=1+w_2}^{w_3} M_j\delta_X N_j{\delta_X}^T T_j\right\}\right\}. \end{aligned} \tag{4.35}$$

Note that the direction δ_X appears linearly in $D\phi(X)[\delta_X]$ as already expected. Consequently, it is clear that the independent term $\mathbb{Q}$ will be provided from

$$\begin{aligned} D\Big(D\phi(X)[\delta_X]\Big)[\delta_V] &= -\operatorname{Tr}\left\{\operatorname{sym}\left\{\delta_V\sum_{i=1}^{k} B_iF^{-1}A_i\right\}\right\} \\ &= -\operatorname{Tr}\left\{\delta_V\mathbb{Q}^T + \mathbb{Q}\delta_V^T\right\}. \end{aligned}$$

Thus, the gradient term $\mathbb{Q}$ is given by

$$\mathbb{Q} = \sum_{i=1}^{k} A_i^T F^{-1} B_i^T.$$

Now, it remains to deal with the quadratic term $D^2\phi(X)[\delta_X,\delta_X]$. Its directional derivative, regarded as a function of δ_X, along the direction δ_V will provide the Hessian map $\mathbb{H}(\delta_X)$. Since the manipulation to obtain $\mathbb{H}(\delta_X)$ is somewhat long and does not provide any interesting insight, we present the final result here and provide the details of the derivation in Appendix B.1.

Let us split the second directional derivative of the barrier (4.35) in four parts, H_1, H_2, H_3, and H_4, so that the directional derivative in δ_V can be applied to each one of the terms separately:

$$\mathsf{H}_1(\delta_X) = \frac{1}{2}\operatorname{Tr}\left\{\left(F^{-1}\operatorname{sym}\left\{\sum_{i=1}^{k} A_i\delta_X B_i\right\}\right)^2\right\},$$

$$\mathsf{H}_2(\delta_X) = -\operatorname{Tr}\left\{F^{-1}\operatorname{sym}\left\{\sum_{j=1}^{w_1} M_j\delta_X N_j\delta_X T_j\right\}\right\},$$

$$\mathsf{H}_3(\delta_X) = -\operatorname{Tr}\left\{F^{-1}\operatorname{sym}\left\{\sum_{j=1+w_1}^{w_2} M_j {\delta_X}^T N_j \delta_X T_j\right\}\right\},$$
$$\mathsf{H}_4(\delta_X) = -\operatorname{Tr}\left\{F^{-1}\operatorname{sym}\left\{\sum_{j=1+w_2}^{w_3} M_j \delta_X N_j {\delta_X}^T T_j\right\}\right\}.$$

After applying directional derivatives (see Appendix B.1), the first term H_1 provides

$$\mathbb{H}_1(\delta_X) = 2\sum_{i=1}^{k} A_i^T F^{-1}\left(\sum_{j=1}^{k} A_j \delta_X B_j\right) F^{-1} B_i^T + 2\sum_{i=1}^{k} A_i^T F^{-1}\left(\sum_{j=1}^{k} B_j^T \delta_X^T A_j^T\right) F^{-1} B_i^T$$

the second term H_2 provides

$$\mathbb{H}_2(\delta_X) = -\sum_{j=1}^{w_1} N_j^T \delta_X^T M_j^T F^{-1} T_j^T + \sum_{j=1}^{w_1} M_j^T F^{-1} T_j^T \delta_X^T N_j^T$$

the third term H_3 provides

$$\mathbb{H}_3(\delta_X) = -\sum_{j=1+w_1}^{w_2} N_j \delta_X T_j F^{-1} M_j + \sum_{j=1+w_1}^{w_2} N_j^T \delta_X M_j^T F^{-1} T_j^T$$

and the fourth term H_4 provides

$$\mathbb{H}_4(\delta_X) = \sum_{j=1+w_2}^{w_3} T_j F^{-1} M_j \delta_X N_j + \sum_{j=1+w_2}^{w_2} M_j^T F^{-1} T_j^T \delta_X N_j^T$$

The final term is

$$D\left(\frac{1}{2} D^2\phi(X)[\delta_X, \delta_X]\right)[\delta_V] = \frac{1}{2}\sum_{i=1}^{4} D\mathsf{H}_i(\delta_X)[\delta_V] = \operatorname{Tr}\left\{\delta_V \mathbb{H}(\delta_X)^T + \mathbb{H}(\delta_X)\delta_V^T\right\}$$

with

$$\mathbb{H}(\delta_X) = \frac{1}{2}\sum_{i=1}^{4} \mathbb{H}_i(\delta_X).$$

Thus, the Hessian map $\mathbb{H}(\delta_X)$ is given by

$$\begin{aligned}\mathbb{H}(\delta_X) = {} & \sum_{i=1}^{k}\sum_{j=1}^{k} A_i^T F^{-1} A_j \delta_X B_j F^{-1} B_i^T + \sum_{i=1}^{k}\sum_{j=1}^{k} A_i^T F^{-1} B_j^T \delta_X^T A_j^T F^{-1} B_i^T \\ & - \frac{1}{2}\sum_{j=1}^{w_1} N_j^T \delta_X^T M_j^T F^{-1} T_j^T + M_j^T F^{-1} T_j^T \delta_X^T N_j^T \\ & - \frac{1}{2}\sum_{j=1+w_1}^{w_2} N_j \delta_X T_j F^{-1} M_j + N_j^T \delta_X M_j^T F^{-1} T_j^T \\ & - \frac{1}{2}\sum_{j=1+w_2}^{w_3} T_j F^{-1} M_j \delta_X N_j + M_j^T F^{-1} T_j^T \delta_X N_j^T\end{aligned}$$

To complete the proof, just note that the optimality condition (4.32) can now be equivalently written as

$$\operatorname{Tr}\left\{\delta_V(\mathbb{H}(\delta_X)-\mathbb{Q})^T+(\mathbb{H}(\delta_X)-\mathbb{Q})\delta_V^T\right\}=0,\quad \text{for all } \delta_V\in\mathcal{V},$$

with $\mathbb{Q}$ and $\mathbb{H}(\delta_X)$ as given above. ∎

Adding the term $\Upsilon_\gamma(X)$ to the potential

The previous Lemma 4.3.3 did not account for the term

$$\Upsilon_\gamma(X):=\zeta\log\left(1/(\gamma-\operatorname{Tr}\{X\}\right).$$

However, the goal is to determine the update direction δ_X for the general unconstrained auxiliary potential function $\phi_\gamma(X)$ containing the term $\Upsilon_\gamma(X)$. Thus, the potential is given by

$$\phi_\gamma(X)=\zeta\log\left(1/(\gamma-\operatorname{Tr}\{X\}\right)-\log\det F(X).$$

Since we have already taken all the necessary derivatives for the term

$$\Theta(X)=-\log\det F(X),$$

it now remains to determine the directional derivatives of the cost term $\Upsilon_\gamma(X)$. Once those derivatives are computed, the resulting expressions will be accordingly added to the expressions for $\mathbb{H}(\delta_X)$ and $\mathbb{Q}$ provided in Lemma 4.3.3. Thus, we proceed by taking directional derivatives of $\Upsilon_\gamma(X)$ along the direction δ_X. The first derivative is given by

$$D\Upsilon_\gamma(X)[\delta_X]=\zeta\left(\gamma-\operatorname{Tr}\{X\}\right)^{-1}\operatorname{Tr}\{\delta_X\},\tag{4.36}$$

and the second by

$$\frac{1}{2}D^2\Upsilon_\gamma(X)[\delta_X,\delta_X]=\frac{1}{2}\zeta\left(\left(\gamma-\operatorname{Tr}\{X\}\right)^{-1}\operatorname{Tr}\{\delta_X\}\right)^2.\tag{4.37}$$

Now, to determine the expressions to be added to the terms $\mathbb{H}(\delta_X)$ and $\mathbb{Q}$, we must consider (4.36) and (4.37) as a function of δ_X and take their directional derivatives along the direction δ_V. Thus, for (4.36) we have

$$\zeta\left(\gamma-\operatorname{Tr}\{X\}\right)^{-1}\operatorname{Tr}\{\delta_V\}\quad\Longrightarrow\quad\hat{\mathbb{Q}}=-\frac{1}{2}\zeta\left(\gamma-\operatorname{Tr}\{X\}\right)^{-1}I,\tag{4.38}$$

and for (4.37), we have

$$\zeta\left(\gamma-\operatorname{Tr}\{X\}\right)^{-2}\operatorname{Tr}\{\delta_X\}\operatorname{Tr}\{\delta_V\}\quad\Longrightarrow\quad\hat{\mathbb{H}}(\delta_X)=\frac{1}{2}\zeta\left(\gamma-\operatorname{Tr}\{X\}\right)^{-2}\operatorname{Tr}\{\delta_X\}I.\tag{4.39}$$

Since we have just obtained the expressions $\hat{\mathbb{Q}}$ and $\hat{\mathbb{H}}(\delta_X)$ which should be added to the gradient term $\mathbb{Q}$ and the Hessian map $\mathbb{H}(\delta_X)$, the previous Lemma 4.3.3 can now be generalized to Proposition 4.3.4.

Proposition 4.3.4 *Let $\mathcal{V}$ be a subspace of $\mathbb{R}^{p\times q}$ and $\mathcal{C}$ be a convex domain in $\mathcal{V}$. Let the map $F(X) : \mathcal{C} \to \mathbb{S}$ be concave. Consider the following unconstrained auxiliary potential function*

$$\phi_\gamma(X) = \zeta \log\Big(1/(\gamma - \operatorname{Tr}\{X\})\Big) - \log\det F(X) : \mathcal{G}_\gamma \to \mathbb{R}, \qquad \zeta \geq 1,$$

where the feasibility domains $\mathcal{G}$ and $\mathcal{G}_\gamma$ are respectively given by

$$\mathcal{G} = \Big\{X \in \mathcal{C} \subset \mathcal{V} : F(X) > 0\Big\}, \qquad \mathcal{G}_\gamma = \{X \in \mathcal{G} : \operatorname{Tr}\{X\} < \gamma\}.$$

Then the update direction δ_X^* toward the central path for the above potential is the solution of the following symbolically computable algebraic linear equation:

$$\operatorname{Tr}\left\{\delta_V(\mathbb{H}(\delta_X) - \mathbb{Q})^T + (\mathbb{H}(\delta_X) - \mathbb{Q})\delta_V^T\right\} = 0, \quad \textit{for all } \delta_V \in \mathcal{V}, \tag{4.40}$$

or equivalently

$$\langle(\mathbb{H}(\delta_X) - \mathbb{Q}), \delta_V\rangle_s = 0, \quad \textit{for all } \delta_V \in \mathcal{V},$$

where $\mathbb{H}(\delta_X)$ is linear as regarded as a function of δ_X, and $\mathbb{Q}$ is an independent term that does not contain δ_X. Moreover, $\mathbb{Q}$ and $\mathbb{H}(\delta_X)$ are given by

$$\mathbb{Q} = \sum_{i=1}^{k} A_i^T F(X)^{-1} B_i^T - \frac{1}{2}\zeta\left(\gamma - \operatorname{Tr}\{X\}\right)^{-1} I,$$

and

$$\begin{aligned}
\mathbb{H}(\delta_X) = &\sum_{i=1}^{k}\sum_{j=1}^{k} A_i^T F^{-1} A_j \delta_X B_j F^{-1} B_i^T + \sum_{i=1}^{k}\sum_{j=1}^{k} A_i^T F^{-1} B_j^T \delta_X^T A_j^T F^{-1} B_i^T \\
&- \frac{1}{2}\sum_{j=1}^{w_1} N_j^T \delta_X^T M_j^T F^{-1} T_j^T + M_j^T F^{-1} T_j^T \delta_X^T N_j^T \\
&- \frac{1}{2}\sum_{j=1+w_1}^{w_2} N_j \delta_X T_j F^{-1} M_j + N_j^T \delta_X M_j^T F^{-1} T_j^T \\
&- \frac{1}{2}\sum_{j=1+w_2}^{w_3} T_j F^{-1} M_j \delta_X N_j + M_j^T F^{-1} T_j^T \delta_X N_j^T \\
&+ \frac{1}{2}\zeta\left(\gamma - \operatorname{Tr}\{X\}\right)^{-2} \operatorname{Tr}\{\delta_X\} I,
\end{aligned}$$

where the terms A, B, M, N, T are obtained from the first and second directional derivatives of $F(X)$ as given by (4.30) *and* (4.31).

Proof. Follows directly by adding (4.38) and (4.39) to the results provided in the previous Lemma 4.3.3. ∎

In the next section, we shall eliminate the quantifier for all δ_V from the above algebraic linear system (4.40), to produce an explicit conventional linear system of equations, which δ_X must satisfy. This is the result provided in Theorem 4.3.5.

4.3.6 The Structure of the linear subproblem

An important feature inherited from the theory behind noncommutative rational functions is that the Newton direction is obtained as the solution of a "matrix" algebraic linear equation. Basically, this algebraic linear system of equations has the following structure:

$$\sum_{i}^{N} \mathcal{A}_i \delta_X \mathcal{B}_i + \sum_{i}^{N} \mathcal{B}_i^T \delta_X \mathcal{A}_i^T = \mathbb{Q},$$

where the $\mathcal{A}$'s and $\mathcal{B}$'s are expressions obtained by collecting the terms on the left and on the right side of the update direction δ_X that appear inside the Hessian map $\mathbb{H}(\delta_X)$, and $\mathbb{Q}$, an independent term which does not contain δ_X. (The entire Section 4.6 is devoted on the issue of collecting terms on an expression.) Following the definitions given in Konstantinov et al. (2000), the map

$$\delta_X \rightarrow \sum_{i}^{N} \mathcal{A}_i \delta_X \mathcal{B}_i + \sum_{i}^{N} \mathcal{B}_i^T \delta_X \mathcal{A}_i^T$$

has been defined as a **Sylvester operator** and the integer $2N$ has been defined as the **Sylvester index**.

The result of Proposition 4.3.4, the algebraic linear equation (4.40), can be further specialized depending upon the structure of the underlying subspace $\mathcal{V}$; in other words, if there is or is not some restriction imposed on X. For this purpose, let us define $\mathcal{V}^{\perp}$ to be the orthogonal complement of $\mathcal{V}$. Moreover, just by appropriately defining variables $\mathcal{A}_i$ and $\mathcal{B}_i$, the Hessian map $\mathbb{H}(\delta_X)$ from Proposition 4.3.4 can be expressed in the equivalent form

$$\mathbb{H}(\delta_X) = \sum_{i=1}^{c_1} \mathcal{A}_i \delta_X \mathcal{B}_i + \sum_{j=c_1+1}^{c_2} \mathcal{A}_j \delta_X^T \mathcal{B}_j + \mathcal{C} \operatorname{Tr}\{\delta_X\},$$

where the integer c_1 is the Sylvester index associated with δ_X, and the number $c_2 - (c_1 + 1)$ is the Sylvester index associated with δ_X^T. The term $\mathcal{C}$ is the cost term given by $\mathcal{C} = 1/2\zeta(\gamma - \operatorname{Tr}\{X\})^{-2} I$. Note that $\mathcal{C}$ is a scalar multiple of the identity matrix.

To be able to solve the linear system of equations (4.40) for δ_X, the structure of the underlying subspace $\mathcal{V}$ must be specified. We describe three different situations that appear frequently:

1. The subspace $\mathcal{V}$ equals $\mathbb{R}^{p\times q}$, so that the unknown X can be any matrix in $\mathbb{R}^{p\times q}$. Then the orthogonal complement of $\mathcal{V}$ contains only the null vector, and

$$\langle(\mathbb{H}(\delta_X)-\mathbb{Q}),\delta_V\rangle_s = 0 \quad \text{for all } \delta_V \in \mathcal{V} \quad \Longrightarrow \quad \mathbb{H}(\delta_X)-\mathbb{Q}=0. \tag{4.41}$$

   Thus

$$\sum_{i=1}^{c_1}\mathcal{A}_i\delta_X\mathcal{B}_i + \sum_{j=c_1+1}^{c_2}\mathcal{A}_j\delta_X^T\mathcal{B}_j + \mathcal{C}\,\mathrm{Tr}\,\{\delta_X\} = \mathbb{Q}.$$

2. The subspace $\mathcal{V}$ equals $\mathbb{S}^p$, so that the unknown X is restricted to being symmetric. Therefore, the subspace $\mathcal{V}^\perp$ is the set of all skew-symmetric matrices, and

$$\langle(\mathbb{H}(\delta_X)-\mathbb{Q}),\delta_V\rangle_s = 0 \quad \text{for all } \delta_V \in \mathcal{V} \tag{4.42}$$

   implies that

$$\mathbb{H}(\delta_X)+\mathbb{H}(\delta_X)^T-(\mathbb{Q}+\mathbb{Q}^T)=0.$$

   Thus

$$\sum_{i=1}^{c_2}\mathcal{B}_i^T\delta_X\mathcal{A}_i^T + \mathcal{A}_i\delta_X\mathcal{B}_i + \mathcal{C}\,\mathrm{Tr}\,\{\delta_X\} = \mathbb{Q}+\mathbb{Q}^T.$$

3. The unknown X is restricted to being a scalar multiple of the identity, that is, $X=\sigma I$, for some scalar σ. Thus the orthogonal complement of $\mathcal{V}$ is given by

$$\delta_V^\perp = \{X : 0 = \langle X, \sigma I\rangle\} = \{X : \mathrm{Tr}\,\{X\}=0\},$$

   and consequently

$$\langle(\mathbb{H}(\delta_X)-\mathbb{Q}),\delta_V\rangle_s = 0 \quad \text{for all } \delta_V \in \mathcal{V} \quad \Longrightarrow \quad \mathrm{Tr}\,\{\mathbb{H}(\delta_X)-\mathbb{Q}\}=0. \tag{4.43}$$

   Which can be equivalently written as

$$\mathrm{Tr}\left\{\sum_{i=1}^{c_2}\mathcal{A}_i\mathcal{B}_i+\mathcal{C}\right\}\delta_X = \mathrm{Tr}\,\{\mathbb{Q}\}, \qquad \delta_X\in\mathbb{R}.$$

Having thus expressed the structure of the underlying subspace $\mathcal{V}$, the main result of this chapter, Theorem 4.3.5 is now presented.

Theorem 4.3.5 *Let $\mathcal{V}$ be a subspace of $\mathbb{R}^{p\times q}$ and $\mathcal{C}$ be a convex domain in $\mathcal{V}$. Let the map $F(X):\mathcal{C}\to\mathbb{S}$ be concave. Consider the following unconstrained auxiliary potential function*

$$\phi_\gamma(X) = \zeta\log\Big(1/(\gamma-\mathrm{Tr}\,\{X\})\Big) - \log\det F(X) : \mathcal{G}_\gamma \to \mathbb{R}, \qquad \zeta\geq 1,$$

where the feasibility domains $\mathcal{G}$ and $\mathcal{G}_\gamma$ are respectively given by

$$\mathcal{G} = \Big\{ X \in \mathcal{C} \subset \mathcal{V} : F(X) > 0 \Big\}, \qquad \mathcal{G}_\gamma = \{ X \in \mathcal{G} : \operatorname{Tr}\{X\} < \gamma \}.$$

Then, depending upon the structure of the underlying subspace $\mathcal{V}$, the update direction δ_X^* toward the central path for the above potential is the solution of one of the following symbolically computable algebraic linear equation:

1. *The subspace $\mathcal{V}$ equals $\mathbb{R}^{p\times q}$, so that the unknown X can be any matrix in $\mathbb{R}^{p\times q}$:*

$$\sum_{i=1}^{c_1} \mathcal{A}_i \delta_X \mathcal{B}_i + \sum_{j=c_1+1}^{c_2} \mathcal{A}_j \delta_X^T \mathcal{B}_j + \mathcal{C} \operatorname{Tr}\{\delta_X\} = \mathbb{Q}.$$

2. *The subspace $\mathcal{V}$ equals $\mathbb{S}^p$, so that the unknown X is restricted to being symmetric:*

$$\sum_{i=1}^{c_2} \mathcal{B}_i^T \delta_X \mathcal{A}_i^T + \mathcal{A}_i \delta_X \mathcal{B}_i + \mathcal{C} \operatorname{Tr}\{\delta_X\} = \mathbb{Q} + \mathbb{Q}^T.$$

3. *The unknown X is restricted to being a scalar multiple of the identity, that is, $X = \sigma I$, for some scalar σ:*

$$\operatorname{Tr}\left\{ \sum_{i=1}^{c_2} \mathcal{A}_i \mathcal{B}_i + \mathcal{C} \right\} \delta_X = \operatorname{Tr}\{\mathbb{Q}\}, \qquad \delta_X \in \mathbb{R}.$$

For this expressions, $\mathbb{Q}$ is the gradient term, which does not contain δ_X, given by

$$\mathbb{Q} = \sum_{i=1}^{k} A_i^T F^{-1} B_i^T - \frac{1}{2}\zeta \left(\gamma - \operatorname{Tr}\{X\}\right)^{-1} I.$$

The term $\mathcal{C}$ is the cost term given by $\mathcal{C} = 1/2\zeta(\gamma - \operatorname{Tr}\{X\})^{-2} I$. And, by an appropriate relabeling, the terms $\mathcal{A}_i$ and $\mathcal{B}_i$ are obtained from the Hessian map $\mathbb{H}(\delta_X)$ presented in Proposition 4.3.4.

Proof. Follows directly from Proposition 4.3.4 by expressing the linear system of equations (4.40) according to the structure of the underlying subspace $\mathcal{V}$ given by (4.41)–(4.43). ∎

The above results provide the necessary conditions that the update δ_X must satisfy in order to be a Newton direction toward the central path of the unconstrained auxiliary potential function $\phi_\gamma(X)$.

4.3.7 Solving the linear subproblem

An important question, which remains unanswered, is how any one of the above linear system of equations can be solved efficiently. To address this issue, this section presents a basic approach that uses the vec operation. By applying some properties of the vec operation (see Horn and Johnson (1999)), it can be shown that any of the above algebraic linear systems can be transformed into the equivalent vector form:

$$\mathcal{H}v = g, \tag{4.44}$$

where $\mathcal{H}$ is the Hessian matrix, which by Corollary 4.3.7 is a symmetric matrix, g is the gradient vector, and v is the vector of unknowns. Depending upon the restriction on X, these parameters are given by one of the following cases:

1. The subspace $\mathcal{V}$ equals $\mathbb{R}^{p\times q}$, then[10]

$$\mathcal{H} = \sum_{i=1}^{c_1} \mathcal{B}_i^T \otimes \mathcal{A}_i + \left[\sum_{j=c_1+1}^{c_2} \mathcal{B}_i^T \otimes \mathcal{A}_i\right] \Pi + \operatorname{vec}(\mathcal{C})\operatorname{vec}(I)^T \qquad \text{and} \qquad g = \operatorname{vec}(\mathbb{Q}),$$

   where Π is a permutation matrix such that $\Pi \operatorname{vec}(\delta_X) = \operatorname{vec}(\delta_X^T)$.

2. The subspace $\mathcal{V}$ equals $\mathbb{S}^p$, then[10]

$$\mathcal{H} = \sum_{i=1}^{c_2} \mathcal{B}_i^T \otimes \mathcal{A}_i + \sum_{i=1}^{c_2} \mathcal{A}_i \otimes \mathcal{B}_i^T + \operatorname{vec}(\mathcal{C})\operatorname{vec}(I)^T \qquad \text{and} \qquad g = \operatorname{vec}(\mathbb{Q} + \mathbb{Q}^T).$$

3. The subspace $\mathcal{V}$ equals σI, for some scalar σ, then

$$\mathcal{H} = \operatorname{Tr}\left\{\sum_{i=1}^{c_2} \mathcal{A}_i\mathcal{B}_i + \mathcal{C}\right\} \qquad \text{and} \qquad g = \operatorname{Tr}\{\mathbb{Q}\}.$$

The final equation (4.44) is now in the conventional vector form, and can be solved by any conventional linear system solver. However, this "brute force" procedure does not take advantage of the particular structure of $\mathbb{H}(\delta_X)$. Naturally, after applying the vec operation, the linear system (4.44) somehow contains this "nice" structure; however, from the knowledge of the author, there is no known practical algorithm that can solve the linear system (4.44) taking into account the structure of $\mathbb{H}(\delta_X)$ for any arbitrary Sylvester index N.

[10]The definition of inner product, given in Section 4.2, is used to obtain that

$$\operatorname{Tr}\{\delta_X\} = \operatorname{vec}(I)^T \operatorname{vec}(\delta_X).$$

For the simpler case when $N = 1$, so that $\mathbb{H}(\delta_X) = \mathcal{A}\delta_X\mathcal{B} + \mathcal{B}^T\delta_X\mathcal{A}^T$, one has a "Lyapunov" type of algebraic equation, for which there are many available numerical and analytical results (see Golub and Loan (1983)). However, in the most general form where the Sylvester index can be any number, there is no satisfactory numerical algorithm, or even theoretical results, that can take advantage of the structure of the system. Some works in this area are Konstantinov et al. (2000). Since most of the running time of the numerical solver is spent on solving the algebraic linear system, a satisfactory theory and algorithm would be valuable. We leave this major open area of work for others, although Section 4.6 describes some basics ways to improve speed.

It is important to show that the Hessian map $\mathbb{H}(\delta_X) : \mathcal{V} \to \mathcal{V}$ is self-adjoint. This result is presented in the following Lemma 4.3.6.

Lemma 4.3.6 *The Hessian map $\mathbb{H}(\delta_X) : \mathcal{V} \to \mathcal{V}$, in Proposition 4.3.4, is a self-adjoint operator.*

Proof. To prove this Lemma, let us first recall the expression for the Hessian map given by

$$\begin{aligned}\mathbb{H}(\delta_X) = &\sum_{i=1}^{k}\sum_{j=1}^{k}(A_i^T F^{-1} A_j)\delta_X(B_j F^{-1} B_i^T) + (A_i^T F^{-1} B_j^T)\delta_X^T(A_j^T F^{-1} B_i^T) \\ &- \frac{1}{2}\sum_{j=1}^{w_1} N_j^T \delta_X{}^T M_j^T F^{-1} T_j^T + M_j^T F^{-1} T_j^T \delta_X{}^T N_j^T \\ &- \frac{1}{2}\sum_{j=1+w_1}^{w_2} N_j^T \delta_X M_j^T F^{-1} T_j^T + N_j \delta_X T_j F^{-1} M_j \\ &- \frac{1}{2}\sum_{j=1+w_2}^{w_3} M_j^T F^{-1} T_j^T \delta_X N_j^T + T_j F^{-1} M_j \delta_X N_j\end{aligned}$$

The cost term $\mathcal{C}\,\mathrm{Tr}\,\{\delta_X\}$ has been omitted from this expression, given that it is a scalar multiple of identity.

The definition of an adjoint operator presented in Section 4.2 implies that for any $Y \in \mathcal{V}$ the adjoint map $\mathbb{H}^\star(\delta_X)$ must satisfy:

$$\langle \mathbb{H}(\delta_X), Y\rangle = \langle \delta_X, \mathbb{H}^\star(Y)\rangle$$

Thus, in order for the map $\mathbb{H}(\delta_X)$ to be self-adjoint, one must show that $\mathbb{H}(\delta_X) = \mathbb{H}^\star(\delta_X)$. The manipulations are as follows:

$$\begin{aligned}
\langle \mathbb{H}(\delta_X), Y \rangle &= \operatorname{Tr}\left\{ \left(\sum_{i=1}^{k}\sum_{j=1}^{k} (A_i^T F^{-1} A_j)\delta_X (B_j F^{-1} B_i^T) \right) Y^T \right\} \\
&\quad + \operatorname{Tr}\left\{ \left(\sum_{i=1}^{k}\sum_{j=1}^{k} (A_i^T F^{-1} B_j^T)\delta_X^T (A_j^T F^{-1} B_i^T) \right) Y^T \right\} \\
&\quad - \operatorname{Tr}\left\{ \left(\frac{1}{2}\sum_{j=1}^{w_1} N_j^T \delta_X{}^T M_j^T F^{-1} T_j^T + M_j^T F^{-1} T_j^T \delta_X{}^T N_j^T \right) Y^T \right\} \\
&\quad - \operatorname{Tr}\left\{ \left(\frac{1}{2}\sum_{j=1+w_1}^{w_2} N_j^T \delta_X M_j^T F^{-1} T_j^T + N_j \delta_X T_j F^{-1} M_j \right) Y^T \right\} \\
&\quad - \operatorname{Tr}\left\{ \left(\frac{1}{2}\sum_{j=1+w_2}^{w_3} M_j^T F^{-1} T_j^T \delta_X N_j^T + T_j F^{-1} M_j \delta_X N_j \right) Y^T \right\} \\
&= \operatorname{Tr}\left\{ \delta_X \sum_{i=1}^{k}\sum_{j=1}^{k} (B_j F^{-1} B_i^T) Y^T (A_i^T F^{-1} A_j) \right\} \\
&\quad + \operatorname{Tr}\left\{ \delta_X \left(\sum_{i=1}^{k}\sum_{j=1}^{k} (B_i F^{-1} A_j) Y (B_j F^{-1} A_i) \right) \right\} \\
&\quad - \frac{1}{2}\operatorname{Tr}\left\{ \delta_X \left(\sum_{j=1}^{w_1} T_j F^{-1} M_j Y N_j + N_j Y T_j F^{-1} M_j \right) \right\} \\
&\quad - \frac{1}{2}\operatorname{Tr}\left\{ \delta_X \left(\sum_{j=1+w_1}^{w_2} M_j^T F^{-1} T_j^T Y^T N_j^T + T_j F^{-1} M_j Y^T N_j \right) \right\} \\
&\quad - \frac{1}{2}\operatorname{Tr}\left\{ \delta_X \left(\sum_{j=1+w_2}^{w_3} N_j^T Y^T M_j^T F^{-1} T_j^T + N_j Y^T T_j F^{-1} M_j \right) \right\} \\
&= \operatorname{Tr}\left\{ \delta_X \mathbb{H}^\star(Y)^T \right\} \\
&= \langle \delta_X, \mathbb{H}^\star(Y) \rangle
\end{aligned}$$

Therefore

$$\begin{aligned}
\mathbb{H}^\star(Y) &= \sum_{i=1}^{k}\sum_{j=1}^{k} (A_i^T F^{-1} A_j) Y (B_j F^{-1} B_i^T) + (A_i^T F^{-1} B_j^T) Y^T (A_j^T F^{-1} B_i^T) \\
&\quad - \frac{1}{2}\sum_{j=1}^{w_1} N_j^T Y^T M_j^T F^{-1} T_j^T + M_j^T F^{-1} T_j^T Y^T N_j^T
\end{aligned}$$

$$-\frac{1}{2}\sum_{j=1+w_1}^{w_2} N_j^T Y M_j^T F^{-1} T_j^T + N_j Y T_j F^{-1} M_j$$
$$-\frac{1}{2}\sum_{j=1+w_2}^{w_3} M_j^T F^{-1} T_j^T Y N_j^T + T_j F^{-1} M_j Y N_j$$

and consequently $\mathbb{H}^\star(\delta_X) = \mathbb{H}(\delta_X)$ as claimed. ■

An immediate consequence of the above Lemma 4.3.6 is the following Corollary 4.3.7

Corollary 4.3.7 *The matrix representation $\mathcal{H}$ for the Hessian map $\mathbb{H}(\delta_X)$ is symmetric.*

Proof. Follows directly from the above Lemma 4.3.6 and from Theorem 4.2.1 on page 95. ■

4.3.8 An algorithm to solve the inner loop

The previous section has just presented the algebraic condition that the update direction should satisfy in order to be a Newton direction toward the analytic center of the unconstrained auxiliary potential function $\phi_\gamma(X)$. Thus, this section presents an algorithm based on a modified Newton's method which solves the analytic center:

$$X^{k+1} = \operatorname{argmin}\left\{\phi_\gamma(X^k) : X^k \in \mathcal{G}_\gamma\right\}$$

for fix scalar γ. This algorithm is described by the following steps:

Algorithm 4.3.8 *Modified Newton's method.*
Let v^k denote the update $v^k = \mathrm{vec}(\delta_{X^k})$;
Fix γ. Let $X^0 \in \mathcal{G}_\gamma$;
Fix N; $k \leftarrow 0$;
while $k < N$ and **not** converged **do**
  Evaluate g^k and $\mathcal{H}^k$;
  Compute v^k such that $\mathcal{H}^k v^k = g^k$;
  $X^{k+1} \leftarrow X^k + \sigma^k \delta_{X^k}$
    by choosing σ^k such that $F(X^{k+1}) > 0$;
end while

The step length σ is chosen such that $F(X^{k+1}) > 0$. We mention a few choices for σ. In the convex programming context, Nesterov and Nemirovskii (1994) provided a formula

which always gives feasible steps. Defining $\tau = \sqrt{g^T \mathcal{H}^{-1} g}$, their step length σ is given by

$$\sigma = \begin{cases} 1/(1+\tau) & \text{, if } \tau > 1/4 \\ 1 & \text{, otherwise} \end{cases} \tag{4.45}$$

For the LMI context, the work by Boyd et al. (1994) suggested that an exact line search produces faster convergence of the method of centers. However, Colaneri et al. (1997) provided a sub-optimal line search, which is computationally cheaper than the exact line search. This suboptimal line search σ is given by the following formula

$$\begin{aligned} \mu &= \lambda_{max} \left[F(x)^{-1/2} (F(x) - F(0)) F(x)^{-1/2} \right] \\ \sigma &= 1/(1+\mu) \end{aligned} \tag{4.46}$$

Once the line search has been selected, the stopping criteria is given by $\tau < \epsilon_1$ or $\mu < \epsilon_2$, for some small enough positive scalars ϵ_1, ϵ_2.

4.4 A Feasibility Example: Riccati Inequality

This section provides a tutorial presentation of the proposed methodology applied to the problem of finding feasible solutions to the matrix Riccati inequality. Let us consider the mapping $\text{Ric}(X) : \mathbb{S}^n \to \mathbb{S}^n$ given by

$$X \to AX + XA^T - XRX + Q$$

with $R \in \mathbb{S}^n_+$ and $Q \in \mathbb{S}^n$. This expression is present in many control applications, and it is known as the Riccati equation. To cite a few examples, it appears in the: Linear Quadratic Regulator (LQR), $\mathcal{H}_\infty$ central controller, output covariance controller, Kalman filtering, and many others. See Boyd et al. (1994); Colaneri et al. (1997); Skelton et al. (1998); Zhou et al. (1996). The book by Lancaster and Rodman (1995) is a good source on the properties of the algebraic Riccati equation and its solutions.

One standard problem that appears frequently, is the one of finding feasible solutions to the matrix Riccati inequality, that is,

$$\text{find } X \text{ such that } \text{Ric}(X) > 0 \tag{4.47}$$

This is a convex problem. The easiest way to see this, is by restating the problem using the Schur complement in the equivalent LMI form:

$$\text{find } X \text{ such that } \begin{bmatrix} AX + XA^T + Q & X\tilde{R} \\ \tilde{R}X & I \end{bmatrix} > 0 \tag{4.48}$$

with $\tilde{R}$ being the Cholesky factorization of R. There are many numerical algorithms available to solve LMIs. Most implementations are based on the Semidefinite Programming, SDP, machinery. See Gahinet et al. (1995); Sturm (1999); Vandenberghe and Balakrishnan (1997); Vandenberghe and Boyd (1995) and references therein. In the literature, one can find specific implementations which are, in practice, very efficient, to deal with Riccati's. Thus, we do not claim that our code is efficient for this specific class of problem

4.4.1 Solving the feasibility problem

We now attempt to solve the feasibility problem (4.47) in its natural setup: find matrix X such that $\mathrm{Ric}(X) > 0$. For this purpose let us assume that set

$$\mathcal{R} = \{X : \mathrm{Ric}(X) > 0\}$$

is nonempty and bounded, or equivalently, the spectrum of $\mathrm{Ric}(X)$ is bounded[11] on the set $\mathcal{R}$. This fact enforces the property that the larger set $\mathcal{G}$, given in (4.50), is bounded for fixed value of α. It is evident that to compute a solution to the feasibility problem (4.47), it suffices to solve the following convex minimization problem

$$\alpha^* = \min\{\alpha : (X, \alpha) \in \mathrm{closure}(\mathcal{G})\} \tag{4.49}$$

where the feasibility set $\mathcal{G}$ is the convex domain given by

$$\begin{aligned} \mathcal{G} &= \{(X, \alpha) : F(X, \alpha) > 0\} \\ F(X, \alpha) &= AX + XA^T - XRX + Q + \alpha I \end{aligned} \tag{4.50}$$

As stated above, the idea is to maximize the minimum eigenvalue of the $\mathrm{Ric}(X)$ operator. This type of optimization is a special case of the problem of minimizing the maximum generalized eigenvalue of symmetric definite pencils (Boyd and El Ghaoui (1993); Nesterov and Nemirovskii (1994); Overton (1988)). The article by Helton and Merino (1997, 1998) provides second-order optimality conditions for minimizing the largest eigenvalue of nonlinear matrix inequalities.

To use the method of centers to solve the optimization problem given in (4.49), an initial feasible guess is needed. This is trivially obtained by choosing X^0 to be any matrix in $\mathbb{S}^n$ and by setting $\alpha^0 > \|\mathrm{Ric}(X^0)\|_2$. A few comments are in order now: (1) the optimum α^* is bounded from below, since $\mathrm{Ric}(X)$ is bounded; (2) for fix scalar γ, the

[11] The spectrum of $\mathrm{Ric}(X)$ is bounded whenever $R \in \mathbb{S}^n_{++}$. It is always possible to impose convex constraints on X to guarantee boundedness, for example, $X^2 < \sigma I$ for $\sigma > 0$.

set $\{(X, \alpha) \in \mathcal{G} : \alpha < \gamma\}$ is bounded; (3) the minimization problem (4.49) may not have a unique solution, in the sense that, for the attained bound α^*, which is unique, there may exist many minimizers X^* such that $(X^*, \alpha^*) \in \text{closure}(\mathcal{G})$. However, if such solution α^* is negative, then any corresponding minimizer X^* has the property that $\text{Ric}(X^*) > 0$. An algorithm to detect feasibility, should stop in practice, as soon as the objective α^k is less than zero at some iteration k.

4.4.2 Describing the central path

In order to implement an algorithm to solve the feasibility problem (4.49), using the method of centers, one needs to be able to compute Newton steps toward the analytic center of some conveniently chosen potential function. For this purpose, let us define the unconstrained auxiliary potential function $\phi_\gamma(X, \alpha)$ as described in Theorem 4.3.5 from Section 4.3.5 (with $\text{Tr}\{X\}$ replaced by α). Thus, the potential $\phi_\gamma(X, \alpha)$ is given by

$$\phi_\gamma(X, \alpha) = \zeta \log(1/(\gamma - \alpha)) - \log\det F(X, \alpha) : \mathcal{G}_\gamma \to \mathbb{R} \tag{4.51}$$

where

$$\mathcal{G}_\gamma = \{(X, \alpha) \in \mathcal{G} : \alpha < \gamma\}$$

The parameter ζ is taken to be $\zeta = n$ (see comments on Section 2.4.3 of Boyd et al. (1994)). The analytic center for the potential $\phi_\gamma(X, \alpha)$ is the path given by

$$(X^*(\gamma^k), \alpha^*(\gamma^k)) = \text{argmin}\left\{\phi_{\gamma^k}(X, \alpha) : (X, \alpha) \in \mathcal{G}_\gamma\right\} \tag{4.52}$$

Solving for the analytic center

The optimization problem (4.49) has now been replaced by a sequence of unconstrained minimization problems in the form (4.52) for a decreasing sequence of scalars $\{\gamma^k\}$ provided by formula (4.24) from Section 4.3.3. In this way, we are interested in finding update directions which lead toward to the central path of (4.52). To find those directions, Newton's method is applied to minimize an approximation (the second-order Taylor series expansion) of the potential function $\phi(X, \alpha)$. In a vague sense, these procedures can be summarized as follow:

1. Compute the second-order Taylor expansion of $\phi_\gamma(X, \alpha)$ in some direction $\delta = (\delta_X, \delta_\alpha)$

$$\phi_\gamma(X, \alpha) + D\phi_\gamma(X, \alpha)[\delta] + \frac{1}{2}D^2\phi_\gamma(X, \alpha)[\delta, \delta]$$

2. The Newton step $\delta^* = (\delta_X^*, \delta_\alpha^*)$ satisfies the necessary optimality conditions for the following quadratic minimization problem

$$\delta^* \in \operatorname{argmin} \left(D\phi_\gamma(X,\alpha)[\delta] + \frac{1}{2} D^2\phi_\gamma(X,\alpha)[\delta,\delta] \right)$$

3. This first-order necessary optimality condition is algebraically given by

$$\begin{aligned} 0 &= D\Big(D\phi_\gamma(X,\alpha)[\delta] + \frac{1}{2} D^2\phi_\gamma(X,\alpha)[\delta,\delta] \Big)[\delta_{V_1}], \qquad \forall \delta_{V_1}. \\ 0 &= D\Big(D\phi_\gamma(X,\alpha)[\delta] + \frac{1}{2} D^2\phi_\gamma(X,\alpha)[\delta,\delta] \Big)[\delta_{V_2}], \qquad \forall \delta_{V_2}. \end{aligned} \tag{4.53}$$

4. Finally, find a Newton update $\delta^* = (\delta_X^*, \delta_\alpha^*)$ satisfying (4.53) for all $\delta_{V_1}, \delta_{V_2}$.

We shall go through each step precisely. For clarity of notation, let us omit the subscript γ in $\phi_\gamma(X,\alpha)$. So, to compute the quadratic approximation of $\phi(X,\alpha)$, we take δ_X and δ_α to be the update directions for X and α respectively. Thus, assuming $X^* = X + \delta_X$ and $\alpha^* = \alpha + \delta_\alpha$, the series expansion of $\phi(X,\alpha)$ up to the second term is given by

$$\begin{aligned} \phi(X^*,\alpha^*) = \phi(X,\alpha) &+ D\phi(X,\alpha)[\delta_X] + \frac{1}{2} D^2\phi(X,\alpha)[\delta_X,\delta_X] \\ &+ D\phi(X,\alpha)[\delta_\alpha] + \frac{1}{2} D^2\phi(X,\alpha)[\delta_\alpha,\delta_\alpha] + \mathrm{D}^2\,\phi(X,\alpha)[\delta_X,\delta_\alpha] + \cdots \end{aligned} \tag{4.54}$$

Directional derivatives of $F(X,\alpha)$

In order to compute the derivatives in Eq. (4.54), we need to have at hand the first and second directional derivatives of $F(X,\alpha)$. Recalling that X is symmetric, and therefore so is the update direction δ_X, the first directional derivative of $F(X,\alpha)$ in the direction δ_X is given by

$$\begin{aligned} DF(X,\alpha)[\delta_X] &= (A - XR)\delta_X + \delta_X(A^T - RX) \\ &= \operatorname{sym}\{(A - XR)\delta_X\} \end{aligned}$$

and the second directional derivative is

$$D^2F(X,\alpha)[\delta_X,\delta_X] = -\operatorname{sym}\{\delta_X R \delta_X\}$$

For the direction δ_α, the first and second directional derivatives are easily found to be

$$DF(X,\alpha)[\delta_\alpha] = \delta_\alpha\, I \qquad \text{and} \qquad D^2F(X,\alpha)[\delta_\alpha,\delta_\alpha] = 0$$

Having the above derivatives available, we are ready to take the directional derivatives needed by (4.54). However, to clarify the exposition, by identifying the contributions of each term in the expressions for the derivatives, the potential function $\phi(X, \alpha)$ is split term by term as:

$$\phi_1(X, \alpha) = -\log\det F(X, \alpha) \qquad \text{and} \qquad \phi_2(X, \alpha) = -n\log(\gamma - \alpha)$$

We also abbreviate $F(X, \alpha)$ to just F.

Directional derivatives of $\phi_1(X, \alpha) = -\log\det F(X, \alpha)$

The first and second directional derivative of $\phi_1(X, \alpha)$ in the direction δ_X are given by

$$\begin{aligned} D\phi_1(X, \alpha)[\delta_X] &= -\operatorname{Tr}\left\{F^{-1}DF[\delta_X]\right\} \\ &= -\operatorname{Tr}\left\{F^{-1}\operatorname{sym}\{(A - XR)\delta_X\}\right\} \end{aligned}$$

$$\begin{aligned} D^2\phi_1(X, \alpha)[\delta_X, \delta_X] &= \operatorname{Tr}\left\{\left(F^{-1}DF[\delta_X]\right)^2\right\} - \operatorname{Tr}\left\{F^{-1}D^2F[\delta_X, \delta_X]\right\} \\ &= \operatorname{Tr}\left\{\left(F^{-1}\operatorname{sym}\{(A - XR)\delta_X\}\right)^2\right\} \\ &\quad + \operatorname{Tr}\left\{F^{-1}\operatorname{sym}\{\delta_X R\delta_X\}\right\} \end{aligned}$$

The second directional derivative of $\phi_1(X, \alpha)$ taken first in the direction δ_X and after in the direction $\delta\alpha$ is given by

$$\begin{aligned} \mathrm{D}^2\,\phi_1(X, \alpha)[\delta_X, \delta_\alpha] &= \operatorname{Tr}\left\{F^{-1}(DF[\delta_\alpha])F^{-1}(DF[\delta_X])\right\} \\ &= \operatorname{Tr}\left\{F^{-1}\delta_\alpha F^{-1}\operatorname{sym}\{(A - XR)\delta_X\}\right\} \\ &= \mathrm{D}^2\,\phi_1(X, \alpha)[\delta_\alpha, \delta_X] \end{aligned}$$

Finally the first and second directional derivatives of $\phi_1(X, \alpha)$ along δ_α are

$$D\phi_1(X, \alpha)[\delta_\alpha] = -\operatorname{Tr}\left\{F^{-1}\delta\alpha\right\} \qquad \text{and} \qquad D^2\phi_1(X, \alpha)[\delta_\alpha, \delta_\alpha] = \operatorname{Tr}\left\{F^{-2}\delta\alpha^2\right\}$$

Directional derivatives $\phi_2(X, \alpha) = -n\log(\gamma - \alpha)$

We need to find the directional derivatives of $\phi_2(X, \alpha)$ relative to δ_X and $\delta\alpha$. For δ_X we have

$$D\phi_2(X, \alpha)[\delta_X] = 0 \qquad \text{and} \qquad D^2\phi_2(X, \alpha)[\delta_X, \delta_X] = 0$$

and for δ_α we have

$$D\phi_2(X,\alpha)[\delta_\alpha] = \mathrm{Tr}\left\{\frac{\delta_\alpha}{(\gamma-\alpha)}I\right\}$$

and

$$D^2\phi_2(X,\alpha)[\delta_\alpha,\delta_\alpha] = \mathrm{Tr}\left\{\frac{\delta\alpha^2}{(\gamma-\alpha)^2}I\right\}$$

The second derivative of $\phi_2(X,\alpha)$ taken first in the direction δ_X and after in the direction δ_α (and vice-versa) is given by

$$D^2\phi_2(X,\alpha)[\delta_X,\delta_\alpha] = 0 = D^2\phi_2(X,\alpha)[\delta_\alpha,\delta_X]$$

Optimality conditions

Now we are ready to write down the optimality conditions which will provide the update direction. These conditions are the first-order necessary optimality conditions for problem (4.52), obtained by taking directional derivatives of the Taylor expansion $\phi(X+\delta_X,\alpha+\delta_\alpha)$, given by Eq. (4.54), as a function of δ_X and δ_α in the directions δ_{V_1} and δ_{V_2} respectively.

Let us define $\delta_{X_1} = \delta_X$ and $\delta_{X_2} = \delta_\alpha$. Then, from the directional derivatives just computed, the expression for the second-order approximation

$$\tilde{\phi} = \sum_{i=1}^{2} D\phi(X,\alpha)[\delta_{X_i}] + \frac{1}{2}\sum_{i=1}^{2}\sum_{j=1}^{2} \mathrm{D}^2\,\phi(X,\alpha)[\delta_{X_i},\delta_{X_j}] \tag{4.55}$$

becomes

$$\tilde{\phi} = \tilde{\phi}_1(\delta_X) + \tilde{\phi}_2(\delta_X,\delta_\alpha) + \tilde{\phi}_3(\delta_\alpha)$$

with

$$\begin{aligned}\tilde{\phi}_1(\delta_X) = &-\mathrm{Tr}\left\{F^{-1}\,\mathrm{sym}\left\{(A-XR)\delta_X\right\}\right\}\\ &+\frac{1}{2}\,\mathrm{Tr}\left\{\left(F^{-1}\,\mathrm{sym}\left\{(A-XR)\delta_X\right\}\right)^2\right\}\\ &+\frac{1}{2}\,\mathrm{Tr}\left\{F^{-1}\,\mathrm{sym}\left\{\delta_X R\delta_X\right\}\right\}\end{aligned}$$

$$\tilde{\phi}_2(\delta_X,\delta_\alpha) = \mathrm{Tr}\left\{F^{-1}\delta_\alpha F^{-1}\,\mathrm{sym}\left\{(A-XR)\delta_X\right\}\right\}$$

$$\tilde{\phi}_3(\delta_\alpha) = \mathrm{Tr}\left\{\frac{\delta_\alpha}{(\gamma-\alpha)}I\right\} + \frac{1}{2}\,\mathrm{Tr}\left\{\frac{\delta\alpha^2}{(\gamma-\alpha)^2}I\right\}$$

Since there are two independent variables δ_X and δ_α, we have to set to zero the directional derivative of Eq. (4.54), first as a function of δ_X in the direction δ_{V_1}, and second as a function of δ_α in the direction δ_{V_2}.

Optimality conditions along the direction δ_X

To accomplish the first step, we should compute

$$D\left(D\phi(X,\alpha)[\delta_X] + \frac{1}{2}D^2\phi(X,\alpha)[\delta_X,\delta_X] + \mathrm{D}^2\,\phi(X,\alpha)[\delta_X,\delta_\alpha]\right)[\delta_{V_1}] = 0 \tag{4.56}$$

That is, we should set to zero the directional derivative of (4.55) as a function of δ_X along the direction δ_{V_1}:

$$D\Big(\tilde{\phi}_1(\delta_X) + \tilde{\phi}_2(\delta_X,\delta_\alpha)\Big)[\delta_{V_1}] = 0$$

To compute this expression, let us expand the terms $\tilde{\phi}_1(\delta_X)$ and $\tilde{\phi}_2(\delta_X,\delta_\alpha)$. The term $\tilde{\phi}_1(\delta_X)$ becomes

$$\begin{aligned}
\tilde{\phi}_1(\delta_X) = &-\operatorname{Tr}\left\{F^{-1}\Big((A-XR)\delta_X + \delta_X(A-XR)^T\Big)\right\} \\
&+\frac{1}{2}\operatorname{Tr}\Big\{F^{-1}(A-XR)\delta_X F^{-1}(A-XR)\delta_X \\
&+F^{-1}(A-XR)\delta_X F^{-1}\delta_X(A-XR)^T + \delta_X R\delta_X \\
&+\delta_X R\delta_X + F^{-1}\delta_X(A-XR)^T F^{-1}(A-XR)\delta_X \\
&+F^{-1}\delta_X(A-XR)^T F^{-1}\delta_X(A-XR)^T\Big\}
\end{aligned}$$

and the term $\tilde{\phi}_2(\delta_X,\delta_\alpha)$ becomes

$$\tilde{\phi}_2(\delta_X,\delta_\alpha) = \operatorname{Tr}\left\{F^{-1}\delta_\alpha F^{-1}\Big((A-XR)\delta_X + \delta_X(A-XR)^T\Big)\right\}$$

It remains to compute the expressions for $D\tilde{\phi}_1(\delta_X)[\delta_{V_1}]$ and $D\tilde{\phi}_2(\delta_X)[\delta_{V_1}]$. After a few manipulations, the term $D\tilde{\phi}_1(\delta_X)[\delta_{V_1}]$ is given by

$$\begin{aligned}
D\tilde{\phi}_1(\delta_X)[\delta_{V_1}] = \operatorname{Tr}\Big\{&\delta_{V_1}\Big(F^{-1}\delta_X(A-XR)^T F^{-1}(A-XR) - F^{-1}(A-XR) \\
&+\frac{1}{2}R\delta_X F^{-1} + \frac{1}{2}F^{-1}\delta_X R + F^{-1}(A-XR)\delta_X F^{-1}(A-XR)\Big) \\
&+\Big((A-XR)^T F^{-1}(A-XR)\delta_X F^{-1} + \frac{1}{2}F^{-1}\delta_X R + \frac{1}{2}R\delta_X F^{-1} \\
&-(A-XR)^T F^{-1} + (A-XR)^T F^{-1}\delta_X(A-XR)^T F^{-1}\Big)\delta_{V_1}\Big\}
\end{aligned}$$

and the term $D\tilde{\phi}_2(\delta_X)[\delta_{V_1}]$ is given by

$$D\tilde{\phi}_2(\delta_X)[\delta_{V_1}] = \operatorname{Tr}\left\{\delta_{V_1}\Big(F^{-1}\delta_\alpha F^{-1}(A-XR)\Big) + \Big((A-XR)^T F^{-1}\delta_\alpha F^{-1}\Big)\delta_{V_1}\right\}$$

Therefore, the algebraic linear system of equations given by (4.56) becomes:

$$\mathrm{Tr}\left\{\delta_{V_1}(\mathbb{H}_1(\delta_X,\delta_\alpha)-\mathbb{Q}_1)^T+(\mathbb{H}_1(\delta_X,\delta_\alpha)-\mathbb{Q}_1)\delta_{V_1}\right\}=0 \tag{4.57}$$

with $\mathbb{H}_1(\delta_X,\delta_\alpha)$ and $\mathbb{Q}_1$ given by

$$\mathbb{Q}_1=(A-XR)^TF^{-1}$$

$$\begin{aligned}\mathbb{H}_1(\delta_X,\delta_\alpha)&=\mathbb{H}_{11}(\delta_X)+\mathbb{H}_{12}(\delta_\alpha)\\&=\frac{1}{2}F^{-1}\delta_XR+\frac{1}{2}R\delta_XF^{-1}+(A-XR)^TF^{-1}(A-XR)\delta_XF^{-1}\\&\quad+(A-XR)^TF^{-1}\delta_X(A-XR)^TF^{-1}+(A-XR)^TF^{-1}\delta_\alpha F^{-1}\end{aligned}$$

Since the unknown X is restricted to being symmetric (so is δ_X) the subspace $\mathcal{V}_1$ equals $\mathbb{S}$. Consequently, its orthogonal complement $\mathcal{V}_1^\perp$ is the set of all skew symmetric matrices. Therefore,

$$\langle(\mathbb{H}_1(\delta_X,\delta_\alpha)-\mathbb{Q}_1),\delta_{V_1}\rangle_s=0\quad\longrightarrow\quad\mathbb{H}_1(\delta_X,\delta_\alpha)+\mathbb{H}_1(\delta_X,\delta_\alpha)^T-(\mathbb{Q}_1+\mathbb{Q}_1^T)=0$$

Thus, we obtain the following linear system in δ_X and δ_α

$$\begin{aligned}\mathbb{Q}_1+\mathbb{Q}_1^T&=\mathbb{H}_{11}(\delta_X)+\mathbb{H}_{12}(\delta_\alpha)\\\mathrm{sym}\left\{F^{-1}(A-XR)\right\}&=\mathrm{sym}\Big\{F^{-1}\delta_XR+F^{-1}\delta X(A-XR)^TF^{-1}(A-XR)\\&\qquad+F^{-1}(A-XR)\delta_XF^{-1}(A-XR)\Big\}\\&\quad+\mathrm{sym}\left\{F^{-1}\delta_\alpha F^{-1}(A-XR)\right\}\end{aligned}\tag{4.58}$$

Optimality conditions along the direction δ_α

Now we should follow a similar procedure for the direction δ_α. We begin by taking directional derivatives of the second-order expansion of $\phi(X,\alpha)$ given by (4.54) as a function of δ_α in the direction δ_{V_2}:

$$D\left(D\phi(X,\alpha)[\delta_\alpha]+\frac{1}{2}D^2\phi(X,\alpha)[\delta_\alpha,\delta_\alpha]+\mathrm{D}^2\,\phi(X,\alpha)[\delta_X,\delta_\alpha]\right)[\delta_{V_2}]=0$$

Which, after some manipulations, becomes

$$\mathrm{Tr}\left\{\delta_{V_2}(\mathbb{H}_2(\delta_X,\delta_\alpha)-\mathbb{Q}_2)^T+(\mathbb{H}_2(\delta_X,\delta_\alpha)-\mathbb{Q}_2)\delta_{V_2}\right\}=0 \tag{4.59}$$

with $\mathbb{H}_2(\delta_X, \delta_\alpha)$ and $\mathbb{Q}_2$ given by

$$\mathbb{Q}_2 = \frac{1}{2}\left(F^{-1} - \frac{1}{(\gamma-\alpha)}I\right)$$

$$\begin{aligned}\mathbb{H}_2(\delta_X, \delta_\alpha) &= \mathbb{H}_{21}(\delta_X) + \mathbb{H}_{22}(\delta_\alpha) \\ &= F^{-1}\delta_X(A - XR)^T F^{-1} + \frac{1}{2}\left(F^{-2} + \frac{1}{(\gamma-\alpha)^2}I\right)\delta_\alpha\end{aligned}$$

In contrast to the previous case, it should be recognized that Eq. (4.59) does not necessarily imply that the relation $\mathbb{H}_2 + \mathbb{H}_2^T - (\mathbb{Q}_2 + \mathbb{Q}_2^T) = 0$ holds true, since δ_{V_2} is not free to be all matrices in $\mathbb{S}^n$. In this case, the unknown α (and δ_α) is a scalar multiple of the identity. If $\delta_{V_2}^\perp$ is defined as the orthogonal complement of the space of all the matrices having the form $\delta_{V_2} = \sigma I$, for some scalar σ, then

$$\delta_{V_2}^\perp = \{X : \langle X, \sigma I\rangle = 0\} = \{X : \mathrm{Tr}\,\{X\} = 0\}$$

Consequently what holds is that Eq. (4.59) implies $\mathrm{Tr}\left\{\mathbb{H}_2 + \mathbb{H}_2^T - (\mathbb{Q}_2 + \mathbb{Q}_2^T)\right\} = 0$. Thus, we have the following linear system in δ_X and δ_α

$$\mathrm{Tr}\left\{\mathbb{Q}_2 + \mathbb{Q}_2^T\right\} = \mathrm{Tr}\left\{\mathbb{H}_{21}(\delta_X)\right\} + \mathrm{Tr}\left\{\mathbb{H}_{22}(\delta_\alpha)\right\}$$

$$\begin{aligned}\mathrm{Tr}\left\{F^{-1} - \frac{1}{(\gamma-\alpha)}I\right\} = \mathrm{Tr}\left\{\mathrm{sym}\left\{F^{-1}(A - XR)\delta_X F^{-1}\right\}\right\} \\ + \mathrm{Tr}\left\{\left(F^{-2} + \frac{1}{(\gamma-\alpha)^2}I\right)\delta_\alpha\right\}\end{aligned} \tag{4.60}$$

The algebraic linear system of equations

As expected, we have just obtained two equations, (4.58) and (4.60), in two unknowns δ_X and $\delta\alpha$. Therefore, to find the Newton directions δ_X and δ_α, we need to simultaneously solve the linear system of equations:

$$\begin{aligned}\mathbb{H}_{11}(\delta_X) + \mathbb{H}_{12}(\delta_\alpha) &= \mathbb{Q}_1 + \mathbb{Q}_1^T \\ \mathrm{Tr}\left\{\mathbb{H}_{21}(\delta_X)\right\} + \mathrm{Tr}\left\{\mathbb{H}_{22}(\delta_\alpha)\right\} &= \mathrm{Tr}\left\{\mathbb{Q}_2 + \mathbb{Q}_2^T\right\}\end{aligned} \tag{4.61}$$

To solve this system, we need to apply the vec operation, transforming the matrix representation (4.61) into the equivalent vector form:

$$\mathcal{H}v = g$$

Before applying the vec operation to the first equation $\mathbb{H}_{11}(\delta_X)+\mathbb{H}_{12}(\delta_\alpha)=\mathbb{Q}_1+\mathbb{Q}_1^T$, let us rewrite this equation using a suitable choice of variables $\mathcal{A}$ and $\mathcal{B}$ as:

$$\text{sym}\left\{\sum_{i=1}^{4}\left(\mathcal{A}_{11}^i\delta_X\mathcal{B}_{11}^i\right)\right\}+\text{sym}\left\{\mathcal{A}_{12}^1\ \delta_\alpha\ \mathcal{B}_{12}^1\right\}=\mathbb{Q}_1+\mathbb{Q}_1^T \tag{4.62}$$

where

$$\begin{aligned}
&\mathcal{A}_{11}^1=F^{-1}(A-XR) && \mathcal{B}_{11}^1=F^{-1}(A-XR)\\
&\mathcal{A}_{11}^2=F^{-1} && \mathcal{B}_{11}^2=(A-XR)^TF^{-1}(A-XR)\\
&\mathcal{A}_{11}^3=\frac{1}{2}F^{-1} && \mathcal{B}_{11}^3=R\\
&\mathcal{A}_{11}^4=R && \mathcal{B}_{11}^4=\frac{1}{2}F^{-1}\\
&\mathcal{A}_{12}^1=F^{-1} && \mathcal{B}_{12}^1=F^{-1}(A-XR)
\end{aligned}$$

and

$$\mathbb{Q}_1=(A-XR)^TF^{-1}$$

Applying the vec operation to both sides of (4.62), gives

$$\begin{aligned}
\sum_{i=1}^{4}&\left((\mathcal{B}_{11}^i)^T\otimes\mathcal{A}_{11}^i+\mathcal{A}_{11}^i\otimes(\mathcal{B}_{11}^i)^T\right)\text{vec}(\delta_X)\\
&+\left((\mathcal{B}_{12}^1)^T\otimes\mathcal{A}_{12}^1+\mathcal{A}_{12}^1\otimes(\mathcal{B}_{12}^1)^T\right)\text{vec}(I)\delta_\alpha\ =\ \text{vec}(\mathbb{Q}_1+\mathbb{Q}_1^T)
\end{aligned} \tag{4.63}$$

In a similar way, the second equation

$$\text{Tr}\left\{\mathbb{H}_{21}(\delta_X)\right\}+\text{Tr}\left\{\mathbb{H}_{22}(\delta_\alpha)\right\}=\text{Tr}\left\{\mathbb{Q}_2+\mathbb{Q}_2^T\right\}$$

can be equivalently written as

$$\text{Tr}\left\{\left(\mathcal{B}_{21}^1\mathcal{A}_{21}^1+(\mathcal{A}_{21}^1)^T(\mathcal{B}_{21}^1)^T\right)\delta_X\right\}+\text{Tr}\left\{\sum_{i=1}^{2}\left(\mathcal{A}_{22}^i\ \delta_\alpha\ \mathcal{B}_{22}^i\right)\right\}=\text{Tr}\left\{\mathbb{Q}_2+\mathbb{Q}_2^T\right\} \tag{4.64}$$

with

$$\begin{aligned}
&\mathcal{A}_{21}^1=F^{-1}(A-XR) && \mathcal{B}_{21}^1=F^{-1}\\
&\mathcal{A}_{22}^1=F^{-1} && \mathcal{B}_{22}^1=F^{-1}\\
&\mathcal{A}_{22}^2=\frac{1}{(\gamma-\alpha)}I && \mathcal{B}_{22}^2=\frac{1}{(\gamma-\alpha)}I
\end{aligned}$$

and

$$\mathbb{Q}_2=\frac{1}{2}\left(F^{-1}-\frac{1}{(\gamma-\alpha)}I\right)$$

Recalling the inner product rule given in (4.2), the above equation (4.64) becomes

$$\operatorname{vec}(\mathcal{B}_{21}^1\mathcal{A}_{21}^1 + (\mathcal{A}_{21}^1)^T(\mathcal{B}_{21}^1)^T)^T \operatorname{vec}(\delta_X) + \operatorname{Tr}\left\{\sum_{i=1}^{2}\left(\mathcal{A}_{22}^i\mathcal{B}_{22}^i\right)\right\}\delta_\alpha = \operatorname{Tr}\{\mathbb{Q}_2 + \mathbb{Q}_2\} \quad (4.65)$$

The final equations (4.63) and (4.65) are now easily represented in the vector form

$$\mathcal{H}v = g$$

where $\mathcal{H}$ is the Hessian matrix, g is the gradient vector, and v is the vector of unknowns. These parameters are given by

$$\mathcal{H} = \begin{bmatrix}\mathcal{H}_{11} & \mathcal{H}_{12}\\ \mathcal{H}_{21} & \mathcal{H}_{22}\end{bmatrix}, \qquad v = \begin{pmatrix}\operatorname{vec}(\delta_X)\\ \delta_\alpha\end{pmatrix}, \qquad \text{and} \qquad g = \begin{pmatrix}\operatorname{vec}(\mathbb{Q}_1 + \mathbb{Q}_1^T)\\ \operatorname{Tr}\{\mathbb{Q}_2 + \mathbb{Q}_2^T\}\end{pmatrix}$$

with

$$\begin{aligned}\mathcal{H}_{11} &= \sum_{i=1}^{4}\left((\mathcal{B}_{11}^i)^T \otimes \mathcal{A}_{11}^i + \mathcal{A}_{11}^i \otimes (\mathcal{B}_{11}^i)^T\right)\\ &= (A - XR)^T F^{-1} \otimes F^{-1}(A - XR) + F^{-1}(A - XR) \otimes (A - XR)^T F^{-1}\\ &\quad + F^{-1} \otimes R + R \otimes F^{-1} + F^{-1} \otimes (A - XR)^T F^{-1}(A - XR)\\ &\quad + (A - XR)^T F^{-1}(A - XR) \otimes F^{-1}\end{aligned}$$

$$\begin{aligned}\mathcal{H}_{12} &= \left[(\mathcal{B}_{12}^1)^T \otimes \mathcal{A}_{12}^1 + \mathcal{A}_{12}^1 \otimes (\mathcal{B}_{12}^1)^T\right]\operatorname{vec}(I)\\ &= \left[(A - XR)^T F^{-1} \otimes F^{-1} + F^{-1} \otimes (A - XR)^T F^{-1}\right]\operatorname{vec}(I)\end{aligned}$$

$$\begin{aligned}\mathcal{H}_{21} &= \operatorname{vec}(\mathcal{B}_{21}^1\mathcal{A}_{21}^1 + (\mathcal{A}_{21}^1)^T(\mathcal{B}_{21}^1)^T)^T\\ &= \operatorname{vec}\left(F^{-2}(A - XR) + (A - XR)^T F^{-2}\right)^T\\ &= \mathcal{H}_{12}^T\end{aligned}$$

$$\mathcal{H}_{22} = \operatorname{Tr}\left\{\sum_{i=1}^{2}\left(\mathcal{A}_{22}^i\mathcal{B}_{22}^i\right)\right\} = \operatorname{Tr}\left\{F^{-2} + \frac{1}{(\gamma - \alpha)^2}I\right\}$$

The above Hessian matrix $\mathcal{H}$ is symmetric. This follows from Theorem 4.2.1, by noting that $\mathbb{H}_{12}(\delta_\alpha)$ is the adjoint map of $\mathbb{H}_{21}(\delta_X)$, and that $\mathbb{H}_{11}(\delta_X)$ and $\mathbb{H}_{22}(\delta_\alpha)$ are self-adjoint maps.

4.4.3 A feasibility algorithm using the method of centers

This section introduces an algorithm that can be used to solve the feasibility problem just presented.

A

Algorithm 4.4.1 *Feasibility algorithm using the method of centers.*
Fix θ such that $0 < \theta < 1$;
Choose feasible X^0, α^0, and γ^0 such that $(X^0, \alpha^0) \in \mathcal{G}_{\gamma^0}$;
$k \leftarrow 0$;
while $\gamma^k > 0$ and **not** converged **do**
    $\gamma^{k+1} \leftarrow (1-\theta)\alpha^k + \theta\gamma^k$;
    Solve $(X^{k+1}, \alpha^{k+1}) = \operatorname{argmin}\left\{\phi_{\gamma^{k+1}}(X^k, \alpha^k) : (X^k, \alpha^k) \in \mathcal{G}_{\gamma^{k+1}}\right\}$;
    $k \leftarrow k+1$;
end while

To find a feasible starting point $(X^0, \alpha^0, \gamma^0)$, it suffices to choose a symmetric matrix X^0 (as e.g. $X^0 = I$) and to set $\gamma^0 > \alpha^0 > \|\operatorname{Ric}(X^0)\|$. Note that the parameter γ^k, given by $\gamma^{k+1} = (1-\theta)\alpha^k + \theta\gamma^k$, never produces infeasibility, since

$$\gamma^{k+1} - \alpha^k = \theta(\gamma^k - \alpha^k) > 0$$

Thus, the arguments $X^k, \alpha^k, \gamma^{k+1}$ are always feasible starting points for the next iterate. If at some iteration k the scalar γ^k is negative, then the problem is feasible and the respective X^k is such that $\operatorname{Ric}(X^k) > 0$. However, if the scalar γ^k is positive for all k, then the sequence $\{\gamma^k\}$ converges to some scalar $\gamma^* > 0$, and consequently the problem is infeasible and there is no X such that $\operatorname{Ric}(X) > 0$.

Describing the inner loop

This section describes an algorithm based on a modified Newton's method, which solves the analytic center:

$$(X^{k+1}, \alpha^{k+1}) = \operatorname{argmin}\left\{\phi_\gamma(X^k, \alpha^k) : (X^k, \alpha^k) \in \mathcal{G}_\gamma\right\}$$

for fix scalar γ. This algorithm can be described by the following steps:

Algorithm 4.4.2 *Modified Newton's method.*
Let v^k denote the update $(v^k)^T = (\mathrm{vec}(\delta_{X^k})^T,\ \delta_{\alpha^k})^T$;
Fix γ. Assume $(X^0, \alpha^0) \in \mathcal{G}_\gamma$;
Fix N; $k \leftarrow 0$;
while $k < N$ and **not** converged **do**
    Evaluate g^k and $\mathcal{H}^k$;
    Compute v^k such that $\mathcal{H}^k v^k = g^k$;
    $(X^{k+1}, \alpha^{k+1}) \leftarrow (X^k, \alpha^k) + \sigma^k(\delta_{X^k}, \delta_{\alpha^k})$
        by choosing σ^k such that $F(X^{k+1}, \alpha^{k+1}) > 0$;
end while

A few choices for the step length σ were presented in the previous Section 4.3.8. We will not discuss which one is more efficient. In order to run this feasibility problem as a tutorial, we use the formula provided in Nesterov and Nemirovskii (1994). Thus, the step length σ is given by

$$\sigma = \begin{cases} 1/(1+\tau) & \text{, if } \tau > 1/4 \\ 1 & \text{, otherwise} \end{cases} \tag{4.66}$$

with $\tau = \sqrt{g^T \mathcal{H}^{-1} g}$. The stopping criteria is thus given by $\tau < \epsilon$, for some small enough positive scalar ϵ.

4.4.4 Numerical results for the feasibility problem

This section presents the numerical results for the Riccati feasibility problem (4.49), using algorithm 4.4.1 and algorithm 4.4.2. The problem[12] is:

find $\alpha^* = \min \alpha$ *subject to*

$$AX + XA^T - XRX + Q + \alpha I > 0$$
$$X > 0.$$

If α^* is negative, then the Riccati inequality is feasible and a symmetric matrix $X > 0$ exists such that

$$\mathrm{Ric}(X) := AX + XA^T - XRX + Q > 0.$$

[12] A copy of the Matlab code used to solve this problem is provided in Section C.1. This code is very simple and is mainly intended to illustrate the proposed methodology. The major code NCSDP implemented by the author to solve the experiments in this thesis is not provided, since it would take excessively many pages.

However, the feasibility problem as presented in (4.49) did not impose the constraint that X should be a positive definite matrix; in many control applications, this is required. Thus, we now demonstrate how to incorporate the constraint $X > 0$ into the formulation. This is accomplished[13] with no great difficulty by adding the barrier $\Theta_3(X) = -\log\det X$ to the unconstrained auxiliary potential function $\phi_\gamma(X, \alpha)$. The directional derivatives of $\Theta_3(X)$ along the direction δ_X are given by

$$D\theta_3(X)[\delta_X] = -\operatorname{Tr}\left\{X^{-1}\delta_X\right\}$$

and

$$\frac{1}{2}D^2\theta_3(X)[\delta_X, \delta_X] = \frac{1}{2}\operatorname{Tr}\left\{(X^{-1}\delta_X)^2\right\}.$$

Thus, we only need to modify (4.61) by adding the term $\frac{1}{2}X^{-1}$ to the gradient map $\mathbb{Q}_1$ and the term $\frac{1}{2}X^{-1}\delta_X X^{-1}$ to the map $\mathbb{H}_{11}(\delta_X)$.

Trade-off between inner and outer iterations

For this first numerical experiment, the Riccati feasibility code is executed using four different values of the centralization parameter θ given by

$$\theta = \begin{bmatrix} 0.01, & 0.1, & 0.3, & 0.6 \end{bmatrix}.$$

The matrices used in this experiment are given by

$$A = \begin{bmatrix} -0.3508 & -1.1081 & 0.7508 \\ 0.8920 & -0.0259 & 0.5001 \\ 1.5782 & -1.1106 & -0.5172 \end{bmatrix}, \qquad R = \begin{bmatrix} 1.5696 & -0.2331 & 0.0688 \\ -0.2331 & 0.2617 & -0.0276 \\ 0.0688 & -0.0276 & 0.5717 \end{bmatrix},$$

$$Q = \begin{bmatrix} -1.4228 & 0.1976 & -0.1470 \\ 0.1976 & -1.6930 & 1.1355 \\ -0.1470 & 1.1355 & 1.3836 \end{bmatrix}.$$

Table 4.1 presents the result for $\theta = 0.01$, Table 4.2 for $\theta = 0.3$, and finally Table 4.3 for $\theta = 0.6$. The results are presented with six digits of accuracy. Figure 4.2 shows the results for all the values of the parameter θ. In these tables, the first column, Iter, shows the total number of iterations required to achieve the global minimum within an accuracy of 10^{-5}, i.e., The code stops when the error on the upper bound γ between two successive iterations, $\gamma^{k+1} - \gamma^k$, is less than 10^{-5}. The second column, NeNe, shows the total number

[13]The constraint $X > 0$ is included in the feasibility code presented in Section C.1.

of Newton steps required to compute the analytic center using Nesterov-Nemirosvky step length, within an accuracy of 10^{-3}. The third column shows the value of the imposed upper bound γ for the specific iteration, and the last column presents the value of α attained at the end of the current iteration.

Iter	NeNe	γ	α
1	21	5.703920	3.847762
2	17	3.866323	2.775534
5	17	1.822595	1.635819
10	16	1.467689	1.463884
15	16	1.461376	1.461333
17	16	1.461316	1.461309
18	16	1.461309	1.461306

Table 4.1: $\theta = 0.01$

Iter	NeNe	γ	α
1	11	5.993920	4.018843
5	7	2.452164	1.975021
10	6	1.569275	1.509191
15	6	1.469455	1.464607
20	6	1.461847	1.461521
26	6	1.461325	1.461312
27	6	1.461316	1.461309

Table 4.2: $\theta = 0.3$

Iter	NeNe	γ	α
1	9	6.293920	4.196176
5	5	3.680773	2.668827
10	4	2.247156	1.862534
20	4	1.535658	1.493563
25	4	1.481364	1.469567
30	4	1.466495	1.463398
35	4	1.462627	1.461707
48	4	1.461341	1.461319
49	4	1.461332	1.461315

Table 4.3: $\theta = 0.6$

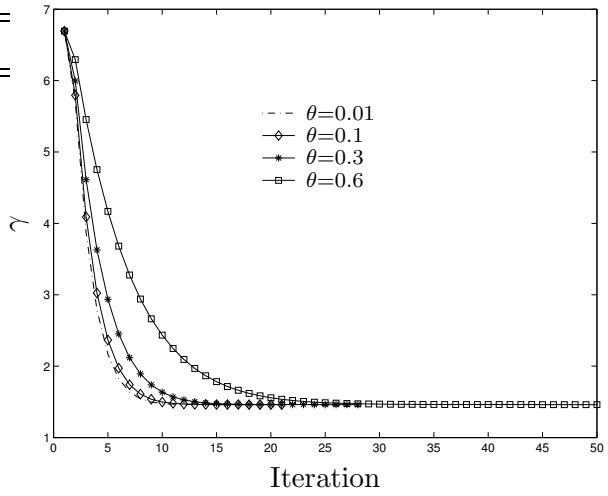

Figure 4.2: Influence of the parameter θ on the method of centers

The initial feasible guess for this experiment was $X^0 = I$, $\alpha^0 = \|\operatorname{Ric}(X^0)\| + 1$, and $\gamma^0 = \alpha^0 + 1$. For all the four centralization parameter θ, the solution converged to $\gamma = 1.4613$. Given that this is a positive number, the problem is infeasible and there is no $X > 0$ such that $\operatorname{Ric}(X) > 0$, for this set of data. The iteration log for the case $\theta = 0.3$ will soon be presented.

As illustrated from the above tables, the trade-off between the number of inner iterations, NeNe, and the number of outer iterations, Iter, is characteristic of Barrier methods,

in particular, the method of centers (Boyd and El Ghaoui (1993)). For $\theta = 0.01$, 18 iterations (297 Newton steps) are required to reduce the objective value within 10^{-5} of the optimal value. For $\theta = 0.1$, the number of iterations is 20 (189 Newton steps). For $\theta = 0.3$, 27 iterations (171 Newton steps) are required. Finally for $\theta = 0.6$, the total number of iterations is 49 (205 Newton steps). Since the cost of numerically computing the update directions (Newton steps) is high, it is desirable to achieve the fewest number of inner iterations as possible. On the other hand, increasing θ may lead to a large number of outer iterations. From Figure 4.2, the parameter $\theta = 0.3$ seems to be a good choice as it provides fast convergence, 27 iterations (few number of outer iterations), while keeping the number of inner iterations (171 Newton steps) low.

Presenting the iteration log for $\theta = 0.3$

This section presents the iteration log for the specific case where the centralization parameter is $\theta = 0.3$. The results are presented below in Table 4.4, where the first column NeNe shows the number of Newton steps required to compute the analytic center using the Nesterov-Nemirosvky step length given by (4.66), within an accuracy of 10^{-3}, i.e., the stopping criteria is $\tau < 10^{-3}$. This computation is performed for each fixed value of the upper bound parameter γ, at each iteration. The second column shows the norm of the gradient vector g, the third column shows τ, the fourth column present the step length σ, the fifth column shows the value of α, and the last two columns presents the minimum and the maximum eigenvalue of the Hessian matrix $\mathcal{H}$. The code stops when the error on the upper bound γ between two successive iterations, $\gamma^{k+1} - \gamma^k$, is less than 10^{-5}. Note that the table does not show every iteration.

Table 4.4: Iteration log for $\theta = 0.3$

NeNe	$\|g\|$	τ	σ	α	$\lambda_{\min}(\mathcal{H})$	$\lambda_{\max}(\mathcal{H})$
Iteration 1				$\gamma = 5.993920$		
1	9.6E+00	1.9E+00	0.3	5.6004145	2.2E+00	3.9E+01
2	7.1E+00	1.8E+00	0.4	5.4780753	1.8E+00	2.5E+01
3	5.2E+00	1.7E+00	0.4	5.3193023	1.5E+00	1.7E+01
⋮		⋮		⋮		
11	2.8E-02	8.0E-03	1.0	4.0198197	1.1E+00	1.6E+01

continued on next page

continued from previous page

NeNe	$\|g\|$	τ	σ	α	$\lambda_{\min}(\mathcal{H})$	$\lambda_{\max}(\mathcal{H})$
12	9.0E-03	2.6E-03	1.0	4.0190821	1.1E+00	1.6E+01
13	2.9E-03	8.6E-04	1.0	4.0188435	1.1E+00	1.6E+01
Iteration 2				$\gamma = 4.6113664$		
1	3.5E+00	1.2E+00	0.5	3.8406879	1.5E+00	1.6E+01
2	2.3E+00	9.4E-01	0.5	3.6384409	1.5E+00	1.8E+01
3	1.3E+00	6.6E-01	0.6	3.4404751	1.6E+00	2.0E+01
⋮		⋮		⋮		
7	3.2E-02	8.4E-03	1.0	3.2081156	1.8E+00	2.6E+01
8	1.1E-02	2.7E-03	1.0	3.2075432	1.8E+00	2.6E+01
9	3.6E-03	8.8E-04	1.0	3.2073622	1.8E+00	2.6E+01
		⋮		⋮		
Iteration 26				$\gamma = 1.4613251$		
1	3.3E+05	1.1E+00	0.5	1.4613168	1.3E+02	8.4E+10
2	2.0E+05	8.5E-01	0.5	1.4613149	1.3E+02	5.9E+10
3	1.0E+05	4.8E-01	0.7	1.4613134	1.3E+02	5.3E+10
4	3.5E+04	1.6E-01	1.0	1.4613127	1.3E+02	6.6E+10
5	6.7E+02	3.0E-03	1.0	1.4613127	1.3E+02	7.8E+10
6	3.7E-01	2.0E-06	1.0	1.4613127	1.3E+02	7.8E+10
Iteration 27				$\gamma = 1.4613164$		
1	5.6E+05	1.1E+00	0.5	1.4613116	1.3E+02	2.5E+11
2	3.5E+05	8.5E-01	0.5	1.4613105	1.3E+02	1.8E+11
3	1.8E+05	4.8E-01	0.7	1.4613096	1.3E+02	1.6E+11
4	6.0E+04	1.6E-01	1.0	1.4613092	1.3E+02	2.0E+11
5	1.2E+03	3.0E-03	1.0	1.4613092	1.3E+02	2.3E+11
6	6.4E-01	1.8E-06	1.0	1.4613092	1.3E+02	2.3E+11

As seen from Table 4.4 above, the prescribed accuracy is reached within 27 iterations, with $\gamma = 1.4613164$ and $\alpha = 1.4613092$. Since α is a positive number, the problem is infeasible and there is no $X > 0$ such that $AX + XA^T - XRX + Q > 0$. The value of the symmetric matrix $X > 0$ corresponding to the achieved α is

$$X = \begin{bmatrix} 0.1429 & 0.0062 & 0.1300 \\ 0.0062 & 0.7850 & 0.6378 \\ 0.1300 & 0.6378 & 0.8539 \end{bmatrix}$$

with its eigenvalues given by

$$\lambda(X) = \Big(0.07050, \quad 0.24573, \quad 1.46567\Big)$$

For this X, the eigenvalues of $\mathrm{Ric}(X) = AX + XA^T - XRX + Q$ are

$$\lambda\left(\mathrm{Ric}(X)\right) = \Big(-1.46130, \quad -1.46130, \quad -0.91451\Big)$$

which show that there are only two eigenvalues binding at the optimal value of α. From Table 4.4, one finds that the condition number of the Hessian is large when α approaches the optimal solution. This ill-conditioning in the Hessian is a well known fact with respect to barrier methods (see Murray (1971); Wright (1992a,b); Wright and Orban (2001) and reference therein). This behavior is highly influenced by the set of constraints that are or are not active (binding) at the solution. It is not an immediate task to determine the set of active constraints in the semidefinite programming formulation (the relation between the binding eigenvalues and the unknown X). This thesis does not investigate this ill-conditioning problem.

An inner product minimization problem

This section presents one more experiment, in which an inner product optimization problem is solved instead of the feasibility problem. We take the same Riccati inequality as before, and solve

find $X^* = \mathrm{argmin}\,\mathrm{Tr}\,\{X\}$, *such that*

$$AX + XA^T - XRX + Q > 0 \tag{4.67}$$

Note that this problem does not impose the constraint $X > 0$.

Considering the development presented in the previous sections, the modifications now proposed with regards to the feasibility code are minimal. Basically, the difference is that the inner product case above is a function of one single variable X. There is no α dependence. Thus, the equation (4.57) used in the feasibility problem now reduces to:

$$\mathbb{Q} = (A - XR)^T F^{-1}$$

$$\begin{aligned}\mathbb{H}(\delta_X) = &\frac{1}{2}F^{-1}\delta_X R + \frac{1}{2}R\delta_X F^{-1} + (A - XR)^T F^{-1}(A - XR)\delta_X F^{-1} \\ &+ (A - XR)^T F^{-1}\delta_X (A - XR)^T F^{-1}.\end{aligned}$$

Since there is no inner product part included in the feasibility problem, we shall add the terms relating to the cost function, $\mathrm{Tr}\{X\}$, to the above equations. As shown from Proposition 4.3.4, this is a trivial step and it suffice to include an extra term to the gradient $\mathbb{Q}$ and another one to the Hessian $\mathbb{H}(\delta_X)$. Thus, the final expressions become:

$$\mathbb{Q} = (A - XR)^T F^{-1} - \frac{1}{2}(\gamma - \mathrm{Tr}\{X\})^{-1} I$$

$$\begin{aligned}\mathbb{H}(\delta_X) = &\frac{1}{2}F^{-1}\delta_X R + \frac{1}{2}R\delta_X F^{-1} + (A - XR)^T F^{-1}(A - XR)\delta_X F^{-1} \\ &+ (A - XR)^T F^{-1}\delta_X (A - XR)^T F^{-1} + \frac{1}{2}(\gamma - \mathrm{Tr}\{X\})^{-2} I.\end{aligned}$$

The example-code that runs this inner product minimization problem is available in Section C.2. The iteration log for this experiment is presented in Table 4.5, where the notation is the same as the one used in previous tables. To run this experiment, a feasible starting point is needed; this is easily obtained by invoking the feasibility code[14]. The numerical data used for this experiment are given by

$$A = \begin{bmatrix} 1.4789 & -0.6841 \\ 1.1380 & -1.2919 \end{bmatrix}, \quad R = \begin{bmatrix} 0.8585 & -0.1979 \\ -0.1979 & 0.6785 \end{bmatrix}$$

$$Q = \begin{bmatrix} 1.4884 & -0.6966 \\ -0.6966 & -0.2463 \end{bmatrix}.$$

The centralization parameter was taken to be $\theta = 0.3$. Not all the iterations are presented in Table 4.5. The stopping criteria was again 10^{-5} for the objective and 10^{-3} for the analytic center.

Table 4.5: An inner product minimization problem

NeNe	$\|g\|$	τ	σ	$\mathrm{Tr}\{X\}$	$\lambda_{\min}(\mathcal{H})$	$\lambda_{\max}(\mathcal{H})$
Iteration 1				$\gamma = 0.2459247$		
1	4.7E+01	1.0E+00	0.5	0.2011414	6.0E-01	2.2E+03
2	3.2E+01	1.0E+00	0.5	0.1788383	6.0E-01	1.0E+03
3	2.1E+01	1.0E+00	0.5	0.1453416	6.0E-01	4.5E+02
4	1.4E+01	1.0E-00	0.5	0.0951365	6.0E-01	2.0E+02
⋮		⋮			⋮	
11	5.4E-01	4.3E-01	0.7	-1.5339932	8.0E-01	2.4E+00

continued on next page

[14]The feasibility code stops as soon as the objective is negative.

continued from previous page						
NeNe	$\|g\|$	τ	σ	$\mathrm{Tr}\{X\}$	$\lambda_{\min}(\mathcal{H})$	$\lambda_{\max}(\mathcal{H})$
12	2.0E-01	1.7E-01	1.0	-1.7350971	8.3E-01	2.4E+00
13	3.6E-03	2.3E-03	1.0	-1.7369562	8.7E-01	2.5E+00
14	3.0E-06	2.0E-06	1.0	-1.7369554	8.7E-01	2.5E+00
Iteration 2				γ =-1.1420914		
1	1.7E+00	6.5E-01	0.6	-1.9563699	1.1E+00	7.0E+00
2	8.8E-01	4.4E-01	0.7	-2.1720246	1.3E+00	4.7E+00
3	3.5E-01	2.0E-01	1.0	-2.3312987	1.5E+00	4.0E+00
4	1.8E-02	1.0E-02	1.0	-2.3395406	1.8E+00	3.9E+00
5	1.9E-05	1.4E-05	1.0	-2.3395506	1.8E+00	3.9E+00
			⋮	⋮		
Iteration 43				γ = -4.0397834		
1	1.8E+05	6.8E-01	0.6	-4.0397910	3.4E+09	7.1E+10
2	1.0E+05	5.3E-01	0.7	-4.0397935	3.8E+09	3.8E+10
3	5.0E+04	3.3E-01	0.8	-4.0397958	4.5E+09	2.4E+10
4	1.8E+04	1.3E-01	1.0	-4.0397972	5.2E+09	1.8E+10
5	1.1E+03	8.8E-03	1.0	-4.0397973	5.7E+09	1.6E+10
6	4.0E+00	3.2E-05	1.0	-4.0397973	5.8E+09	1.6E+10
Iteration 44				γ = -4.0397931		
1	2.4E+05	6.8E-01	0.6	-4.0397990	5.8E+09	1.2E+11
2	1.3E+05	5.3E-01	0.7	-4.0398009	6.5E+09	6.5E+10
3	6.6E+04	3.3E-01	0.8	-4.0398026	7.6E+09	4.0E+10
4	2.3E+04	1.3E-01	1.0	-4.0398037	8.8E+09	3.0E+10
5	1.4E+03	8.8E-03	1.0	-4.0398038	9.8E+09	2.7E+10
6	5.2E+00	3.2E-05	1.0	-4.0398038	9.8E+09	2.7E+10

From the above Table 4.5, the prescribed accuracy is reached within 44 iterations, with the upper bound given by $\gamma = -4.0397931$ and the final values for the objective given by $\mathrm{Tr}\{X\} = -4.0398038$. The corresponding symmetric matrix X is

$$X = \begin{bmatrix} -0.60215 & -0.46865 \\ -0.46865 & -3.43764 \end{bmatrix}.$$

For this X, the eigenvalues of $\mathrm{Ric}(X) = AX + XA^T - XRX + Q$ are given by

$$\lambda(\mathrm{Ric}(X)) = \begin{pmatrix} 0.00004, & 0.00002 \end{pmatrix}.$$

Thus, the constraint are active (the eigenvalues are binding) at the solution. The condition number of the Hessian for this example remains bounded.

4.5 Extending the Results to the Multivariate Case

The previous Section 4.3 limited the presentation to the univariate case, by considering uniquely a function $F(X)$ of one variable X. This current section extends the results to the multivariate case by considering functions on several variables $X_1, \ldots, X_r$. However, we will not attempt to provide a detailed presentation of the derivations, as done in the previous section, but rather, the exposition will be concise. Thus, the derivations will be based on the results provided in Section 4.3 for the univariate case.

To introduce the multivariate case, let us start by posing a more general form of our convex optimization problem for matrix functions. For $j = 1, \ldots, r$, let $\mathcal{C}_j$ be a bounded convex domain in $\mathbb{R}^{p_i,q_i}$. Denote by $\mathcal{C}$ the usual Cartesian product $\mathcal{C} = \mathcal{C}_1 \times \cdots \times \mathcal{C}_r$. For each $i = 1, \ldots, m$, let the map $F_i(X_1, \ldots, X_r) : \mathcal{C} \to \mathbb{S}^{n_i}$ be concave. Then, the inner product minimization problem is posed as:

find t^*, if one exists, such that

$$t^* = \min \{\mathrm{Tr}\,\{X_1\} : X_1 \in \mathrm{closure}(\mathcal{G})\}$$

where the feasibility set $\mathcal{G}$ is the convex domain given by

$$\mathcal{G} = \Big\{(X_1, \ldots, X_r) \in \mathcal{C} : F_i(X_1, \ldots, X_r) > 0\Big\}.$$

4.5.1 Unconstrained auxiliary potential function

The idea behind the method of centers is to approach the above constrained optimization problem with a sequence of unconstrained optimization problems which minimize a potential function. Thus, the auxiliary potential function has to be generalized to the multivariate case:

$$\phi_\gamma(X_1, \ldots, X_r) = \log\left(1/(\gamma - \mathrm{Tr}\,\{X_1\})\right) - \sum_{i=1}^{m} \log\det F_i(X_1, \ldots, X_r).$$

The aim is to determine the algebraic linear system of equations which will provide the update directions toward to the central path given by $\phi_\gamma(X_1, \ldots, X_r)$. For the multivariate

case, there will be r update directions $\delta_{X_1}, \ldots, \delta_{X_r}$, since there are r unknowns $X_1, \ldots, X_r$. Therefore, the linear system will have dimension $r \times r$. Basically, it will have the following form:

$$\begin{array}{ccccccccc} \mathbb{H}_{11}(\delta_{X_1}) & + & \mathbb{H}_{12}(\delta_{X_2}) & + & \cdots & + & \mathbb{H}_{1r}(\delta_{X_r}) & = & \mathbb{Q}_1 \\ \mathbb{H}_{21}(\delta_{X_1}) & + & \mathbb{H}_{22}(\delta_{X_2}) & + & \cdots & + & \mathbb{H}_{2r}(\delta_{X_r}) & = & \mathbb{Q}_2 \\ \vdots & & \vdots & & & & \vdots & & \vdots \\ \mathbb{H}_{r1}(\delta_{X_1}) & + & \mathbb{H}_{r2}(\delta_{X_2}) & + & \cdots & + & \mathbb{H}_{rr}(\delta_{X_r}) & = & \mathbb{Q}_r \end{array}$$

where each one of the Hessian $\mathbb{H}_{ij}(\delta_{X_j})$ is a Sylvester operator (defined in Section(4.3.6)), a linear map on the update δ_{X_j}, which has the form[15]:

$$\mathbb{H}_{ij}(\delta_{X_j}) = \sum_{\ell}^{N} \mathcal{A}_{\ell}^{i,j} \delta_{X_j} \mathcal{B}_{\ell}^{i,j} + \sum_{\ell}^{N} (\mathcal{B}_{\ell}^{i,j})^T \delta_{X_j}^T (\mathcal{A}_{\ell}^{i,j})^T$$

and the gradient term $\mathbb{Q}_i$ is an independent expression that does not contain the updates δ_{X_j}.

To build this system, one needs to take directional derivatives of a second-order Taylor expansion of the auxiliary potential function $\phi_\gamma(X_1, \ldots, X_r)$. For clarity of notation, let us omit the subscript γ in $\phi_\gamma(X_1, \ldots, X_r)$. Also, let us use an arrow over a variable to indicate that the variable is a list of elements, e.g., $\overrightarrow{X} = \{X_1, \ldots, X_r\}$. So, to compute the quadratic approximation of $\phi(\overrightarrow{X})$, let us denote by δ_{X_i} the update directions for each X_i. Thus, assuming $X_i^* = X_i + \delta_{X_i}$, the series expansion of $\phi(\overrightarrow{X})$ up to the second-order term is given by

$$\phi(X_1^*, \ldots, X_r^*) = \phi(\overrightarrow{X}) + \sum_{i=1}^{r} D\phi(\overrightarrow{X})[\delta_{X_i}] + \frac{1}{2} \sum_{i=1}^{r} \sum_{j=1}^{r} \mathrm{D}^2 \, \phi(\overrightarrow{X})[\delta_{X_i}, \delta_{X_j}]. \tag{4.68}$$

Recall that the first-order optimality conditions, which will provide the update directions $\delta_{X_1}, \ldots, \delta_{X_r}$, are obtained by taking directional derivatives of the Taylor expansion $\phi(X_1 + \delta_{X_1}, \ldots, X_r + \delta_{X_r})$, given by (4.68), as a function of $\delta_{X_1}, \ldots, \delta_{X_r}$ in the directions $\delta_{V_1}, \ldots, \delta_{V_r}$. Since there are r independent variables $\delta_{X_1}, \ldots, \delta_{X_r}$, we first set to zero the directional derivative of (4.68) as a function of δ_{X_1} in the direction δ_{V_1}, providing the first equation

$$\mathbb{H}_{11}(\delta_{X_1}) + \mathbb{H}_{12}(\delta_{X_2}) + \cdots + \mathbb{H}_{1r}(\delta_{X_r}) = \mathbb{Q}_1.$$

Second, we set to zero the derivative of (4.68) as a function of δ_{X_2} in the direction δ_{V_2}, providing the linear system

$$\mathbb{H}_{21}(\delta_{X_1}) + \mathbb{H}_{22}(\delta_{X_2}) + \cdots + \mathbb{H}_{2r}(\delta_{X_r}) = \mathbb{Q}_2.$$

[15]The notation presented in this formula for $\mathbb{H}_{ij}(\delta_{X_j})$ is over simplified. The actual formula is presented in Proposition 4.5.1. However, the emphases here is on the linear form (the Sylvester form) of the expressions.

And so on.

4.5.2 Notation for the multivariate case

In Section 4.2, it was shown that the directional derivative of a noncommutative rational function has the representation:

$$DF(X)[\delta_X] = \text{sym}\left\{\sum_{i=1}^{k} A_i \delta_X B_i\right\}.$$

However, for the multivariate case, the notation becomes a bit more complex, since it must be clear which function and which variables are being considered. Thus, the notation for the first directional derivative of $F_i(X_1, \ldots, X_r)$ in the direction δ_{X_t} is:

$$DF_i(\vec{X})[\delta_{X_t}] = \text{sym}\left\{\sum_{\ell=1}^{k(i,t)} A_\ell^{i,t} \delta_{X_t} B_\ell^{i,t}\right\} \tag{4.69}$$

where the subscript i corresponds to the function and the subscript t to the direction. For example, the notation for the first directional derivative of $F_3(X_1, \ldots, X_r)$ in the direction δ_{X_2} is represented by:

$$DF_3(\vec{X})[\delta_{X_2}] = \text{sym}\left\{\sum_{\ell=1}^{k(3,2)} A_\ell^{3,2} \delta_{X_2} B_\ell^{3,2}\right\}.$$

For the second directional derivative, the notation is more involved, since there are two different directions. Thus, the second directional derivative of $F_i(X_1, \ldots, X_r)$ taken first[16] in the direction δ_{X_t} and second in the direction δ_{X_s} is represented by

$$\begin{aligned}
\mathrm{D}^2 F_i(\vec{X})[\delta_{X_t}, \delta_{X_s}] = \\
\text{sym}\Bigg\{ &\sum_{\ell=1}^{w_1(i,ts)} M_\ell^{i,ts} \delta_{X_t} N_\ell^{i,ts} \delta_{X_s} T_\ell^{i,ts} + \sum_{\ell=1+w_1(i,ts)}^{w_2(i,ts)} M_\ell^{i,ts} \delta_{X_t}^T N_\ell^{i,ts} \delta_{X_s} T_\ell^{i,ts} \\
&+ \sum_{\ell=1+w_2(i,ts)}^{w_3(i,ts)} M_\ell^{i,ts} \delta_{X_t} N_\ell^{i,ts} \delta_{X_s}^T T_\ell^{i,ts} + \sum_{\ell=1+w_3(i,ts)}^{w_4(i,ts)} M_\ell^{i,ts} \delta_{X_t}^T N_\ell^{i,ts} \delta_{X_s}^T T_\ell^{i,ts} \Bigg\}
\end{aligned} \tag{4.70}$$

[16] The subscript ts denotes: first in the direction t and second in the direction s. However, it is immaterial the order in which the derivatives are taken, since $\mathrm{D}^2 F_i(\vec{X})[\delta_{X_t}, \delta_{X_s}] = \mathrm{D}^2 F_i(\vec{X})[\delta_{X_s}, \delta_{X_t}]$.

However, for the specific case where the directional derivatives are taken along the same direction δ_{X_t}, the formulas reduce to

$$\mathrm{D}^2 F_i(\overrightarrow{X})[\delta_{X_t}, \delta_{X_t}] = \operatorname{sym}\left\{ \sum_{\ell=1}^{w_1(i,t)} M_\ell^{i,t} \delta_{X_t} N_\ell^{i,t} \delta_{X_t} T_\ell^{i,t} + \sum_{\ell=1+w_1(i,t)}^{w_2(i,t)} M_\ell^{i,t} \delta_{X_t}^T N_\ell^{i,t} \delta_{X_t} T_\ell^{i,t} + \sum_{\ell=1+w_2(i,t)}^{w_3(i,t)} M_\ell^{i,t} \delta_{X_t} N_\ell^{i,t} \delta_{X_t}^T T_\ell^{i,t} \right\} \quad (4.71)$$

Which is equivalent to the univariate case given by expression (4.14) from Section 4.2.

4.5.3 Deriving the optimality condition

We should now proceed with the derivatives, however, since the cost term

$$\log\left(1/(\gamma - \operatorname{Tr}\{X_1\})\right)$$

in the potential $\phi(\overrightarrow{X})$ is exactly the same as the one in the univariate case, we concentrate the efforts in manipulating only the Barrier term given by

$$\Theta(\overrightarrow{X}) = -\sum_{i=1}^{m} \log\det F_i(X_1, \ldots, X_r).$$

And later, we add the contributions from the cost term.

Analogous to the derivation in the previous Section 4.3, the gradient term $\mathbb{Q}_t$ is obtained from the first-order approximation of the potential function:

$$D\big(D\Theta(\overrightarrow{X})[\delta_{X_t}]\big)[\delta_{V_t}].$$

Thus, we first need to take the derivative of the potential $\Theta(\overrightarrow{X})$ in the direction δ_{X_t}. From the formulas for the derivative of the log det function (see Section 4.2), we have:

$$\begin{aligned}
D\Theta(\overrightarrow{X})[\delta_{X_t}] &= -\operatorname{Tr}\left\{ \sum_{i=1}^{m} F_i^{-1} DF_i(\overrightarrow{X})[\delta_{X_t}] \right\} \\
&= -\operatorname{Tr}\left\{ \sum_{i=1}^{m} F_i^{-1} \operatorname{sym}\left\{ \sum_{\ell=1}^{k(i,t)} A_\ell^{i,t} \delta_{X_t} B_\ell^{i,t} \right\} \right\} \\
&= -\operatorname{Tr}\left\{ \operatorname{sym}\left\{ \sum_{i=1}^{m} \sum_{\ell=1}^{k(i,t)} F_i^{-1} A_\ell^{i,t} \delta_{X_t} B_\ell^{i,t} \right\} \right\} \\
&= -\operatorname{Tr}\left\{ \operatorname{sym}\left\{ \delta_{X_t} \sum_{i=1}^{m} \sum_{\ell=1}^{k(i,t)} B_\ell^{i,t} F_i^{-1} A_\ell^{i,t} \right\} \right\}
\end{aligned}$$

Now, regarding the above result as a function of δ_{X_t}, we need to compute its first derivative along the direction δ_{V_t}:

$$\begin{aligned} D\big(D\Theta(\overrightarrow{X})[\delta_{X_t}]\big)[\delta_{V_t}] &= -\operatorname{Tr}\left\{\operatorname{sym}\left\{\delta_{V_t}\sum_{i=1}^{m}\sum_{\ell=1}^{k(i,t)} B_\ell^{i,t}F_i^{-1}A_\ell^{i,t}\right\}\right\} \\ &= -\operatorname{Tr}\left\{\delta_{V_t}\mathbb{Q}_t^T + \mathbb{Q}_t\delta_{V_t}^T\right\}. \end{aligned}$$

Thus, we obtain the gradient term $\mathbb{Q}_t$ as

$$\mathbb{Q}_t = \sum_{i=1}^{m}\sum_{\ell=1}^{k(i,t)} (A_\ell^{i,t})^T F_i^{-1}(B_\ell^{i,t})^T$$

for $1 \leq t \leq r$.

The first-order approximation of the potential function provided the gradient term $\mathbb{Q}_t$. On the other hand, the formulas for the Hessian maps $\mathbb{H}_{11}(\delta_{X_1})$, $\mathbb{H}_{12}(\delta_{X_2})$, $\ldots$, $\mathbb{H}_{rr}(\delta_{X_r})$ are obtained by taking directional derivatives of the second-order approximation of the potential function

$$\frac{1}{2}\sum_{s=1}^{r}\sum_{t=1}^{r} \mathrm{D}^2\,\phi(\overrightarrow{X})[\delta_{X_s},\delta_{X_t}]$$

along the direction δ_{V_t}, for $1 \leq t \leq r$. Since the above expression is linear as a function of either one of the directions δ_{X_t} or δ_{X_s}, the formula for each $\mathbb{H}_{ts}(\delta_{X_s})$, for $1 \leq t < s \leq r$, is given by

$$\begin{aligned} \mathbb{H}_{ts}(\delta_{X_s}) &= \frac{1}{2}D\left(\mathrm{D}^2\,\phi(\overrightarrow{X})[\delta_{X_t},\delta_{X_s}] + \mathrm{D}^2\,\phi(\overrightarrow{X})[\delta_{X_s},\delta_{X_t}]\right)[\delta_{V_t}] \\ &= D\left(\mathrm{D}^2\,\phi(\overrightarrow{X})[\delta_{X_t},\delta_{X_s}]\right)[\delta_{V_t}] \end{aligned} \tag{4.72}$$

For the particular case where $1 \leq t = s \leq r$, the map $\mathbb{H}_{tt}$ is given by

$$\mathbb{H}_{tt}(\delta_{X_t}) = \frac{1}{2}D\left(\mathrm{D}^2\,\phi(\overrightarrow{X})[\delta_{X_t},\delta_{X_t}]\right)[\delta_{V_t}].$$

Since the Hessian is self-adjoint, each $\mathbb{H}_{st}(\delta_{X_t})$ is the adjoint map of $\mathbb{H}_{ts}(\delta_{X_s})$.

To find the expressions for the above formulas, the second derivative of the potential function needs to be computed. Since we are not considering the cost term yet, we just present the formulas for the second directional derivative of the barrier $\Theta(\overrightarrow{X})$, taken first in the direction δ_{X_t} and second in the direction δ_{X_s}:

$$\begin{aligned} \mathrm{D}^2\,\Theta(\overrightarrow{X})[\delta_{X_t},\delta_{X_s}] = \operatorname{Tr}&\left\{\sum_{i=1}^{m} F_i^{-1}DF_i(\overrightarrow{X})[\delta_{X_s}]F_i^{-1}DF_i(\overrightarrow{X})[\delta_{X_t}]\right\} \\ &- \operatorname{Tr}\left\{\sum_{i=1}^{m} F_i^{-1}\,\mathrm{D}^2\,F_i(\overrightarrow{X})[\delta_{X_t},\delta_{X_s}]\right\} \end{aligned}. \tag{4.73}$$

Again, for the specific case where the directional derivatives are taken along the same direction δ_{X_t}, the formulas reduce to

$$\begin{aligned}\mathrm{D}^2\,\Theta(\vec{X})[\delta_{X_t},\delta_{X_t}] = \mathrm{Tr}\left\{\sum_{i=1}^{m}\left(F_i^{-1}DF_i(\vec{X})[\delta_{X_t}]\right)^2\right\} \\ -\mathrm{Tr}\left\{\sum_{i=1}^{m}F_i^{-1}\,\mathrm{D}^2\,F_i(\vec{X})[\delta_{X_t},\delta_{X_t}]\right\}.\end{aligned}$$

Which is equivalent to the univariate case presented in Section 4.2.2. In this way, the formula for the map $\mathbb{H}_{tt}(\delta_{X_t})$, for $t = 1,\ldots,r$, is easily obtained from Lemma 4.3.3 by noting that instead of a single function $F(X)$ we have $\sum_{i=1}^{m}F_i(\vec{X})$:

$$\begin{aligned}\mathbb{H}_{tt}(\delta_{X_t}) = &\sum_{i=1}^{m}\sum_{\ell=1}^{k(i,t)}\sum_{\eta=1}^{k(i,t)}(A_\ell^{i,t})^T F_i^{-1}A_\eta^{i,t}\delta_{X_t}B_\eta^{i,t}F_i^{-1}(B_\ell^{i,t})^T \\ &+\sum_{i=1}^{m}\sum_{\ell=1}^{k(i,t)}\sum_{\eta=1}^{k(i,t)}(A_\ell^{i,t})^T F_i^{-1}(B_\eta^{i,t})^T\delta_{X_t}^T(A_\eta^{i,t})^T F_i^{-1}(B_\ell^{i,t})^T \\ &-\frac{1}{2}\sum_{i=1}^{m}\sum_{\ell=1}^{w_1(i,t)}(N_\ell^{i,t})^T{\delta_{X}}_t^T(M_\ell^{i,t})^T F_i^{-1}(T_\ell^{i,t})^T+(M_\ell^{i,t})^T F_i^{-1}(T_\ell^{i,t})^T{\delta_{X}}_t^T(N_\ell^{i,t})^T \\ &-\frac{1}{2}\sum_{i=1}^{m}\sum_{\ell=1+w_1(i,t)}^{w_2(i,t)}(N_\ell^{i,t})^T\delta_{Xt}(M_\ell^{i,t})^T F_i^{-1}(T_\ell^{i,t})^T+N_\ell^{i,t}\delta_{Xt}T_\ell^{i,t}F_i^{-1}M_\ell^{i,t} \\ &-\frac{1}{2}\sum_{i=1}^{m}\sum_{\ell=1+w_2(i,t)}^{w_3(i,t)}(M_\ell^{i,t})^T F_i^{-1}(T_\ell^{i,t})^T\delta_{Xt}(N_\ell^{i,t})^T+T_\ell^{i,t}F_i^{-1}M_\ell^{i,t}\delta_{Xt}N_\ell^{i,t}\end{aligned}$$

where the terms A, B, M, N, T are obtained from the first and second directional derivatives of $F_i(\vec{X})$ given by (4.69) and (4.71).

We should now proceed deriving the formulas for $\mathbb{H}_{ts}(\delta_{X_s})$, for the case where $1 \leq t < s \leq r$, presented in (4.72), by first evaluating expression (4.73) and later substituting the result inside (4.72). Considering the expressions for the first and second directional

derivative of $F_i(\overrightarrow{X})$, the expression for $\mathrm{D}^2\,\Theta(\overrightarrow{X})[\delta_{X_t},\delta_{X_s}]$ becomes

$$\begin{aligned}
\mathrm{D}^2\,\Theta(\overrightarrow{X})[\delta_{X_t},\delta_{X_s}] = \\
&\mathrm{Tr}\left\{\sum_{i=1}^{m} F_i^{-1}\,\mathrm{sym}\left\{\sum_{\ell=1}^{k(i,s)} A_\ell^{i,s}\delta_{X_s}B_\ell^{i,s}\right\}F_i^{-1}\,\mathrm{sym}\left\{\sum_{\eta=1}^{k(i,t)} A_\eta^{i,t}\delta_{X_t}B_\eta^{i,t}\right\}\right\} \\
-\,&\mathrm{Tr}\left\{\sum_{i=1}^{m} F_i^{-1}\,\mathrm{sym}\left\{\sum_{\ell=1}^{w_1(i,ts)} M_\ell^{i,ts}\delta_{X_t}N_\ell^{i,ts}\delta_{X_s}T_\ell^{i,ts} + \sum_{\ell=1+w_1(i,ts)}^{w_2(i,ts)} M_\ell^{i,ts}\delta_{X_t}^T N_\ell^{i,ts}\delta_{X_s}T_\ell^{i,ts}\right.\right. \\
&\left.\left. + \sum_{\ell=1+w_2(i,ts)}^{w_3(i,ts)} M_\ell^{i,ts}\delta_{X_t}N_\ell^{i,ts}\delta_{X_s}^T T_\ell^{i,ts} + \sum_{\ell=1+w_3(i,ts)}^{w_4(i,ts)} M_\ell^{i,ts}\delta_{X_t}^T N_\ell^{i,ts}\delta_{X_s}^T T_\ell^{i,ts}\right\}\right\} \quad (4.74)
\end{aligned}$$

To compute the above second directional derivative, let us split this expression into five parts, H_1, H_2, H_3, H_4 and H_5, so that the directional derivative can be applied to each one of the terms separately:

$$\begin{aligned}
\mathsf{H}_1 &= \frac{1}{2}\,\mathrm{Tr}\left\{\sum_{i=1}^{m} F_i^{-1}\,\mathrm{sym}\left\{\sum_{\ell=1}^{k(i,s)} A_\ell^{i,s}\delta_{X_s}B_\ell^{i,s}\right\}F_i^{-1}\,\mathrm{sym}\left\{\sum_{\eta=1}^{k(i,t)} A_\eta^{i,t}\delta_{X_t}B_\eta^{i,t}\right\}\right\} \\
\mathsf{H}_2 &= -\frac{1}{2}\,\mathrm{Tr}\left\{\sum_{i=1}^{m} F_i^{-1}\,\mathrm{sym}\left\{\sum_{\ell=1}^{w_1(i,ts)} M_\ell^{i,ts}\delta_{X_t}N_\ell^{i,ts}\delta_{X_s}T_\ell^{i,ts}\right\}\right\} \\
\mathsf{H}_3 &= -\frac{1}{2}\,\mathrm{Tr}\left\{\sum_{i=1}^{m} F_i^{-1}\,\mathrm{sym}\left\{\sum_{\ell=1+w_1(i,ts)}^{w_2(i,ts)} M_\ell^{i,ts}\delta_{X_t}^T N_\ell^{i,ts}\delta_{X_s}T_\ell^{i,ts}\right\}\right\} \\
\mathsf{H}_4 &= -\frac{1}{2}\,\mathrm{Tr}\left\{\sum_{i=1}^{m} F_i^{-1}\,\mathrm{sym}\left\{\sum_{\ell=1+w_2(i,ts)}^{w_3(i,ts)} M_\ell^{i,ts}\delta_{X_t}N_\ell^{i,ts}\delta_{X_s}^T T_\ell^{i,ts}\right\}\right\} \\
\mathsf{H}_5 &= -\frac{1}{2}\,\mathrm{Tr}\left\{\sum_{i=1}^{m} F_i^{-1}\,\mathrm{sym}\left\{\sum_{\ell=1+w_3(i,ts)}^{w_4(i,ts)} M_\ell^{i,ts}\delta_{X_t}^T N_\ell^{i,ts}\delta_{X_s}^T T_\ell^{i,ts}\right\}\right\}
\end{aligned}$$

After applying directional derivatives (See Appendix B.2), the first term H_1 provides

$$\begin{aligned}
\mathbb{H}_1(\delta_{X_s}) = &\sum_{i=1}^{m}\sum_{\ell=1}^{k(i,s)}\sum_{\eta=1}^{k(i,t)}\left((A_\eta^{i,t})^T F_i^{-1}A_\ell^{i,s}\delta_{X_s}B_\ell^{i,s}F_i^{-1}(B_\eta^{i,t})^T\right) \\
&+\sum_{i=1}^{m}\sum_{\ell=1}^{k(i,s)}\sum_{\eta=1}^{k(i,t)}\left((A_\eta^{i,t})^T F_i^{-1}(B_\ell^{i,s})^T\delta_{X_s}^T(A_\ell^{i,s})^T F_i^{-1}(B_\eta^{i,t})^T\right)
\end{aligned}$$

the second term H_2 gives

$$\mathbb{H}_2 = -\sum_{i=1}^{m}\sum_{\ell=1}^{w_1(i,ts)}(M_\ell^{i,ts})^T F_i^{-1}(T_\ell^{i,ts})^T\delta_{X_s}^T(N_\ell^{i,ts})^T$$

the third term H_3 gives

$$\mathbb{H}_3 = -\sum_{i=1}^{m} \sum_{\ell=1+w_1(i,ts)}^{w_2(i,ts)} N_\ell^{i,ts} \delta_{X_s} T_\ell^{i,ts} F_i^{-1} M_\ell^{i,ts}$$

the fourth term H_4 provides

$$\mathbb{H}_4 = -\sum_{i=1}^{m} \sum_{\ell=1+w_2(i,ts)}^{w_3(i,ts)} (M_\ell^{i,ts})^T F_i^{-1} (T_\ell^{i,ts})^T \delta_{X_s} (N_\ell^{i,ts})^T$$

and finally the fifth term H_5 gives

$$\mathbb{H}_5 = -\sum_{i=1}^{m} \sum_{\ell=1+w_3(i,ts)}^{w_4(i,ts)} N_\ell^{i,ts} \delta_{X_s}^T T_\ell^{i,ts} F_i^{-1} M_\ell^{i,ts}$$

The final term

$$\mathbb{H}_{ts}(\delta_{X_s}) = \frac{1}{2} \sum_{i=1}^{5} \mathbb{H}_i$$

is thus given by

$$\begin{aligned}
\mathbb{H}_{ts}(\delta_{X_s}) = {} & \frac{1}{2} \sum_{i=1}^{m} \sum_{\ell=1}^{k(i,s)} \sum_{\eta=1}^{k(i,t)} \left((A_\eta^{i,t})^T F_i^{-1} A_\ell^{i,s} \delta_{X_s} B_\ell^{i,s} F_i^{-1} (B_\eta^{i,t})^T \right) \\
& + \frac{1}{2} \sum_{i=1}^{m} \sum_{\ell=1}^{k(i,s)} \sum_{\eta=1}^{k(i,t)} \left((A_\eta^{i,t})^T F_i^{-1} (B_\ell^{i,s})^T \delta_{X_s}^T (A_\ell^{i,s})^T F_i^{-1} (B_\eta^{i,t})^T \right) \\
& - \frac{1}{2} \sum_{i=1}^{m} \sum_{\ell=1}^{w_1(i,ts)} (M_\ell^{i,ts})^T F_i^{-1} (T_\ell^{i,ts})^T \delta_{X_s}^T (N_\ell^{i,ts})^T \\
& - \frac{1}{2} \sum_{i=1}^{m} \sum_{\ell=1+w_1(i,ts)}^{w_2(i,ts)} N_\ell^{i,ts} \delta_{X_s} T_\ell^{i,ts} F_i^{-1} M_\ell^{i,ts} \\
& - \frac{1}{2} \sum_{i=1}^{m} \sum_{\ell=1+w_2(i,ts)}^{w_3(i,ts)} (M_\ell^{i,ts})^T F_i^{-1} (T_\ell^{i,ts})^T \delta_{X_s} (N_\ell^{i,ts})^T \\
& - \frac{1}{2} \sum_{i=1}^{m} \sum_{\ell=1+w_3(i,ts)}^{w_4(i,ts)} N_\ell^{i,ts} \delta_{X_s}^T T_\ell^{i,ts} F_i^{-1} M_\ell^{i,ts}
\end{aligned}$$

We conclude this section by presenting Proposition 4.5.1 which summarizes the results obtained for the multivariate case.

Proposition 4.5.1 *For $j = 1, \ldots, r$, let $\mathcal{V}_j$ be a subspace of $\mathbb{R}^{p_j, q_j}$ and $\mathcal{C}_j$ be a bounded convex domain in $\mathcal{V}_j$. Denote by $\mathcal{C}$ the usual Cartesian product $\mathcal{C} = \mathcal{C}_1 \times \cdots \times \mathcal{C}_r$. For each*

$i = 1, \ldots, m$, let the map $F_i(X_1, \ldots, X_r) : \mathcal{C} \to \mathbb{S}^{n_i}$ be concave. Let $\zeta \geq 1$. Consider the following unconstrained auxiliary potential function

$$\phi_\gamma(X_1, \ldots, X_r) = \zeta \log\left(1/(\gamma - \operatorname{Tr}\{X_1\})\right) - \sum_{i=1}^{m} \log\det F_i(X_1, \ldots, X_r) : \mathcal{G}_\gamma \to \mathbb{R},$$

where the feasibility domain $\mathcal{G}$ is given by

$$\mathcal{G} = \Big\{(X_1, \ldots, X_r) \in \mathcal{C} : F_i(X_1, \ldots, X_r) > 0\Big\},$$

and the domain $\mathcal{G}_\gamma$ is given by

$$\mathcal{G}_\gamma = \{X_1 \in \mathcal{G} : \operatorname{Tr}\{X_1\} < \gamma\}$$

Then the update directions $\delta^*_{X_1}, \ldots, \delta^*_{X_r}$ toward the central path for the above potential is the solution of the following symbolically computable algebraic linear system of equations:

$$\begin{aligned}
&\operatorname{Tr}\left\{\delta_{V_1}\left(\sum_{s=1}^{r}\mathbb{H}_{1s}(\delta_{X_s}) - \mathbb{Q}_1\right)^T + \left(\sum_{s=1}^{r}\mathbb{H}_{1s}(\delta_{X_s}) - \mathbb{Q}_1\right)\delta_{V_1}^T\right\} = 0, \\
&\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad \textit{for all } \delta_{V_1} \in \mathcal{V}_1 \\
&\operatorname{Tr}\left\{\delta_{V_2}\left(\sum_{s=1}^{r}\mathbb{H}_{2s}(\delta_{X_s}) - \mathbb{Q}_2\right)^T + \left(\sum_{s=1}^{r}\mathbb{H}_{2s}(\delta_{X_s}) - \mathbb{Q}_2\right)\delta_{V_2}^T\right\} = 0, \\
&\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad \textit{for all } \delta_{V_2} \in \mathcal{V}_2 \\
&\qquad\qquad\qquad\qquad\vdots \qquad\qquad\qquad\qquad\qquad\vdots \qquad\qquad\vdots \\
&\operatorname{Tr}\left\{\delta_{V_r}\left(\sum_{s=1}^{r}\mathbb{H}_{rs}(\delta_{X_s}) - \mathbb{Q}_r\right)^T + \left(\sum_{s=1}^{r}\mathbb{H}_{rs}(\delta_{X_s}) - \mathbb{Q}_r\right)\delta_{V_r}^T\right\} = 0, \\
&\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad \textit{for all } \delta_{V_r} \in \mathcal{V}_r
\end{aligned}$$

or equivalently

$$\begin{aligned}
\langle(\mathbb{H}_{1s}(\delta_{X_s}) - \mathbb{Q}_1), \delta_{V_1}\rangle_s = 0, \quad &\textit{for all } \delta_{V_1} \in \mathcal{V}_1 \\
\langle(\mathbb{H}_{2s}(\delta_{X_s}) - \mathbb{Q}_2), \delta_{V_2}\rangle_s = 0, \quad &\textit{for all } \delta_{V_1} \in \mathcal{V}_2 \\
\vdots \qquad\qquad\qquad &\vdots \\
\langle(\mathbb{H}_{rs}(\delta_{X_s}) - \mathbb{Q}_r), \delta_{V_r}\rangle_s = 0, \quad &\textit{for all } \delta_{V_1} \in \mathcal{V}_r
\end{aligned}$$

where each $\mathbb{H}_{ts}(\delta_{X_s})$ is linear in δ_{X_s}, and $\mathbb{Q}_t$ is an independent term that does not contain the update directions. Moreover, the gradient terms are given by

$$\mathbb{Q}_1 = \sum_{i=1}^{m}\sum_{\ell=1}^{k(i,1)} B_\ell^{i,1} F_i^{-1} A_\ell^{i,1} - \frac{1}{2}\zeta\left(\gamma - \operatorname{Tr}\{X_1\}\right)^{-1} I$$

and for $2 \leq t \leq r$

$$\mathbb{Q}_t = \sum_{i=1}^{m} \sum_{\ell=1}^{k(i,t)} B_\ell^{i,t} F_i^{-1} A_\ell^{i,t}$$

The Hessian map, for the case where $1 \leq t < s \leq r$*, is given by*

$$\begin{aligned}
\mathbb{H}_{ts}(\delta_{X_s}) = &\frac{1}{2} \sum_{i=1}^{m} \sum_{\ell=1}^{k(i,s)} \sum_{\eta=1}^{k(i,t)} \left((A_\eta^{i,t})^T F_i^{-1} A_\ell^{i,s} \delta_{X_s} B_\ell^{i,s} F_i^{-1} (B_\eta^{i,t})^T \right) \\
&+ \frac{1}{2} \sum_{i=1}^{m} \sum_{\ell=1}^{k(i,s)} \sum_{\eta=1}^{k(i,t)} \left((A_\eta^{i,t})^T F_i^{-1} (B_\ell^{i,s})^T \delta_{X_s}^T (A_\ell^{i,s})^T F_i^{-1} (B_\eta^{i,t})^T \right) \\
&- \frac{1}{2} \sum_{i=1}^{m} \sum_{\ell=1}^{w_1(i,ts)} (M_\ell^{i,ts})^T F_i^{-1} (T_\ell^{i,ts})^T \delta_{X_s}^T (N_\ell^{i,ts})^T \\
&- \frac{1}{2} \sum_{i=1}^{m} \sum_{\ell=1+w_1(i,ts)}^{w_2(i,ts)} N_\ell^{i,ts} \delta_{X_s} T_\ell^{i,ts} F_i^{-1} M_\ell^{i,ts} \\
&- \frac{1}{2} \sum_{i=1}^{m} \sum_{\ell=1+w_2(i,ts)}^{w_3(i,ts)} (M_\ell^{i,ts})^T F_i^{-1} (T_\ell^{i,ts})^T \delta_{X_s} (N_\ell^{i,ts})^T \\
&- \frac{1}{2} \sum_{i=1}^{m} \sum_{\ell=1+w_3(i,ts)}^{w_4(i,ts)} N_\ell^{i,ts} \delta_{X_s}^T T_\ell^{i,ts} F_i^{-1} M_\ell^{i,ts}
\end{aligned}$$

and the terms $\mathbb{H}_{tt}(\delta_{X_t})$*, for* $t = 1, \ldots, r$*, are given by:*

$$\begin{aligned}
\mathbb{H}_{tt}(\delta_{X_t}) = &\sum_{i=1}^{m} \sum_{\ell=1}^{k(i,t)} \sum_{\eta=1}^{k(i,t)} (A_\ell^{i,t})^T F_i^{-1} (B_\eta^{i,t})^T \delta_{X_t}^T (A_\eta^{i,t})^T F_i^{-1} (B_\ell^{i,t})^T \\
&+ \sum_{i=1}^{m} \sum_{\ell=1}^{k(i,t)} \sum_{\eta=1}^{k(i,t)} (A_\ell^{i,t})^T F_i^{-1} A_\eta^{i,t} \delta_{X_t} B_\eta^{i,t} F_i^{-1} (B_\ell^{i,t})^T \\
&- \frac{1}{2} \sum_{i=1}^{m} \sum_{\ell=1}^{w_1(i,t)} (N_\ell^{i,t})^T \delta_{X_t}^T (M_\ell^{i,t})^T F_i^{-1} (T_\ell^{i,t})^T + (M_\ell^{i,t})^T F_i^{-1} (T_\ell^{i,t})^T \delta_{X_t}^T (N_\ell^{i,t})^T \\
&- \frac{1}{2} \sum_{i=1}^{m} \sum_{\ell=1+w_1(i,t)}^{w_2(i,t)} (N_\ell^{i,t})^T \delta_{X_t} (M_\ell^{i,t})^T F_i^{-1} (T_\ell^{i,t})^T + N_\ell^{i,t} \delta_{X_t} T_\ell^{i,t} F_i^{-1} M_\ell^{i,t} \\
&- \frac{1}{2} \sum_{i=1}^{m} \sum_{\ell=1+w_2(i,t)}^{w_3(i,t)} (M_\ell^{i,t})^T F_i^{-1} (T_\ell^{i,t})^T \delta_{X_t} (N_\ell^{i,t})^T + T_\ell^{i,t} F_i^{-1} M_\ell^{i,t} \delta_{X_t} N_\ell^{i,t}
\end{aligned}$$

with $\mathbb{H}_{11}(\delta_{X_1})$ *containing the cost term:*

$$\frac{1}{2} \zeta \left(\gamma - \operatorname{Tr}\{X_1\} \right)^{-2} \operatorname{Tr}\{\delta_{X_1}\} I$$

4.6 Improving the Evaluation Time for the Linear Subproblem

It was briefly stated that an important question is how one can symbolically simplify the expressions that appear in the linear subproblem in such a way that when we substitute matrices for the symbols which appear in these expressions, the evaluation time is reduced, thereby improving the overall time spent by the linear solver. This section addresses this issue more closely.

4.6.1 The algebraic equation

Recall that the algebraic linear system of equations, which provides the necessary conditions that the update δ_X must satisfy in order to be a Newton direction, has basically the following structure:

$$\sum_{i}^{N} \mathcal{A}_i \delta_X \mathcal{B}_i + \sum_{i}^{N} \mathcal{B}_i^T \delta_X \mathcal{A}_i^T = \mathbb{Q} \tag{4.75}$$

where the $\mathcal{A}$'s and $\mathcal{B}$'s are obtained by collecting the terms on the left and on the right side of the update direction δ_X that appear inside the Hessian map $\mathbb{H}(\delta_X)$, and $\mathbb{Q}$ is an independent term that does not contain δ_X. In this expression, the integer $2N$ has been defined as the Sylvester index.

The next subsection explains in further detail how the evaluation time of our linear solver can be reduced by collecting the terms which appear in the expressions for the Hessian map; in other words, the aim is to represent the Hessian map $\mathbb{H}(\delta_X)$ with the Sylvester index N as small as possible.

4.6.2 Basic ideas on collecting terms in an expression

Even though we will not present the details of how our optimization code is implemented, we expose the fact that the algorithm can be split into two parts: a symbolic part and a numerical part.

Roughly speaking, at the symbolic level, Mathematica computes the first and second directional derivatives of the unconstrained auxiliary potential function $\phi_\gamma(X)$, which incorporate the objective and the constraints. From those derivatives one obtain the maps for the Hessian $\mathbb{H}(\delta_X)$ and for the Gradient $\mathbb{Q}$, producing in this way an algebraic linear

equation like (4.75). As already emphasized, the aim is to simplify symbolically the final expressions such that when we substitute matrices for the symbols, the time spent on formulas evaluation can be minimized.

To attain this goal, we should observe that even if two symbolic rational functions may at a first glance, look different, they in fact can be totally equivalent. This happens frequently inside noncommutative rational functions containing a large number of terms. It is also important to collect terms in an expression. This is illustrated by a very simple example which, in practices, appears in a more complex fashion. Suppose one has an expression like

$$A_1\delta_X + \cdots + A_p\delta_X$$

To evaluate this expression, after the δ_X and the A_i have been replaced by matrices, one would need p matrix additions and also p matrix multiplications. On the other hand, collecting the above expression in δ_X gives

$$(\sum_{i=1}^{p} A_i)\delta_X$$

Now, the Sylvester index has dropped from p to 1, and one needs p matrix additions and only one matrix multiplication.

The process of collecting terms in an expression may not be unique. Suppose that $\mathbb{H}(\delta_X)$ is given by

$$\mathbb{H}(\delta_X) = A\delta_X A^T + X^T\delta_X X + B\delta_X B^T - A\delta_X X - X^T\delta_X A^T + B\delta_X A^T + A\delta_X B^T \quad (4.76)$$

The Sylvester index in this case is seven. This expression can be collected in at least two different ways, having the same number of terms. One possibility is:

$$\mathbb{H}(\delta_X) = (A - X^T)\delta_X(A - X^T)^T + (A + B)\delta_X(A + B)^T - A\delta_X A^T = \sum_{i=1}^{3} \mathcal{A}_i\delta_X\mathcal{B}_i$$

for $\mathcal{A}_i$ and $\mathcal{B}_i$ given by

$$\begin{aligned} \mathcal{A}_1 &= (A - X^T), & \mathcal{A}_2 &= (A + B), & \mathcal{A}_3 &= -A \\ \mathcal{B}_1 &= (A - X^T)^T, & \mathcal{B}_2 &= (A + B)^T, & \mathcal{B}_3 &= A^T \end{aligned}$$

Another one is

$$\mathbb{H}(\delta_X) = (A + B - X^T)\delta_X(A + B - X^T)^T + B\delta_X X + X^T\delta_X B^T = \sum_{i=1}^{3} \mathcal{A}_i\delta_X\mathcal{B}_i$$

for $\mathcal{A}_i$ and $\mathcal{B}_i$ given by

$$\begin{aligned} \mathcal{A}_1 &= (A + B - X^T), \qquad \mathcal{A}_2 = B, \qquad \mathcal{A}_3 = X^T \\ \mathcal{B}_1 &= (A + B - X^T)^T, \qquad \mathcal{B}_2 = X, \qquad \mathcal{B}_3 = B^T \end{aligned}$$

In both cases the Sylvester index decreased to $N = 3$. There is yet another possible way to collect terms in the expression (4.76). This is presented in the next subsection where our NCCollectSylvester command is introduced.

4.6.3 Implementing a simple NCCollectSylvester command

The recipe presented here was implemented in NCAlgebra/Mathematica and is used by our optimization code for solving matrix inequalities. The idea is as follows:

1. The user identifies the terms in which the expression should be collected. In the example given by expression (4.76), this term is δ_X.

2. Now, we build a "right list" of terms that multiplies δ_X from the right side (including δ_X itself). For the expression (4.76), we would obtain

$$\text{RightList}=\{\delta_X A^T,\ \delta_X X,\ \delta_X B^T\}.$$

3. For each element inside RightList, we add together all the terms that multiplies this element from the left side, thereby producing a list which is defined as "CollectList."

   To apply this idea to our example, using the above RightList, we proceed as follows: the first element $\delta_X A^T$ in RightList appears inside the expression (4.76) in the terms $A\delta_X A^T$, $B\delta_X A^T$, and $-X^T\delta_X A^T$, thus the first entry in CollectList is $(A+B-X^T)$; in a similar fashion, the element $\delta_X X$ in RightList appears in the expression (4.76) as $X^T\delta_X X - A\delta_X X$, thereby providing the second entry in CollectList given by $(X^T - A)$; finally the element $\delta_X B^T$ appears in the expression (4.76) as $A\delta_X B^T + B\delta_X B^T$, thereby providing the term $(A+B)$ as the third entry in CollectList. Thus, we obtain

$$\text{CollectList}=\{(A + B - X^T),\ (X^T - A),\ (A + B)\}.$$

4. The collected expression is now readily obtained by combining together the CollectList and the RightList.

$$\mathbb{H}(\delta_X) = (A + B - X^T)\delta_X A^T + (X^T - A)\delta_X X + (A + B)\delta_X B^T \tag{4.77}$$

The above "right sided" implementation of the collecting algorithm begins by building a list of multipliers from the right side of δ_X. Evidently, a similar implementation can also be done by obtaining a "left list" of terms that multiplies δ_X from the left side, instead of the right side. In this way, we can implement two collect commands that differ only by the side in which the process of collecting begins, thus, we can have a **NCRightSylvester[]** command and a **NCLeftSylvester[]** command. The implementation used in our NCSDP optimization code is nothing more than a single collect command, defined as **NCCollect-Sylvester**[expr, var], that sequentially applies both commands NCRightSylvester[] and NCLeftSylvester[] to the expression. Thus, the command **NCCollectSylvester**[expr, var] collects the Sylvester terms of expression *expr* according to the element *var*. Now, we show how to use this command in NCAlgebra/Mathematica language[17]. First, define the expression (4.76) in Mathematica as:

```
In[21]:= P := A ** DX ** tp[A] + tp[X] ** DX ** X + B ** DX ** tp[B] - A ** DX ** X - tp[X]
          ** DX ** tp[A] + B ** DX ** tp[A] + A ** DX ** tp[B];
```

To collect this expression in DX (which represents δ_X), we apply our NCCollectSylvester[] command to the expression P using the following syntax:

```
In[22]:= NCCollectSylvester[P, DX]
```

This command outputs the expression:

```
(A + B) ** DX ** tp[B] + (A + B - tp[X]) ** DX ** tp[A] + (-A + tp[X]) **
DX ** X
```

Which, as expected, is the same expression as the one given in (4.77).

Based on these ideas, a few important questions can be formulated. For instance, given an expression for the Hessian map $\mathbb{H}(\delta_X)$ it is fundamental to know what is the minimum Sylvester index associated with this expression and if there exists a theory that shows how to provide this minimum Sylvester index. It is also a fundamental question to know how many different ways of collecting an expression achieving this minimal Sylvester index are possible. Once the formula for the minimum is obtained, can a practical Collect algorithm be implemented which guarantees this minimum Sylvester index? These fundamental questions remain open.

[17] In NCAlgebra, tp[] stands for transpose and the noncommutative multiplication is represented by **.

4.6.4 Illustrating our NCCollectSylvester command by an example

The previous examples were presented in order to illustrate what we mean by collecting terms in an expression, and did not present any numerical evidence validating the usefulness of the idea. Thus, we now explore the main point of this section: the time saving obtained on formulas evaluation by applying our NCCollectSylvester command. For this purpose, let us use the optimization problem presented in Section 4.7. The example is the following eigenvalue minimization problem

$$\min \lambda_{\max}(CXC^T)$$

subject to

$$\begin{aligned} 0 < F(X) := A_1X + XA_1^T - XR_1^{-1}X + Q_1 - \\ (A_2^TX + XA_2)\left(A_3X + XA_3^T - XR_3^{-1}X + Q_3\right)^{-1}(A_2^TX + XA_2) \end{aligned}$$

$$0 < G(X) := A_3X + XA_3^T - XR_3^{-1}X + Q_3$$

with all the matrices having dimension $n \times n$.

As already described, we need to compute symbolically, at the level of Mathematica, the Hessian of a potential function. For the above example, the auxiliary potential function is given by the following formula

$$\phi_\gamma(X) = -\log\det F(X) - \log\det G(X) - \log\det \gamma I - CXC^T$$

where γ is a scalar which is not relevant here. The above expression $\phi_\gamma(X)$ is a function of the unknown X. If the update direction is taken to be δ_X, the Hessian map $\mathbb{H}(\delta_X)$, as a function of δ_X, will have a structure similar to:

$$\mathbb{H}(\delta_X) = \sum_i^N \mathcal{A}_i\delta_X\mathcal{B}_i + \sum_i^N \mathcal{B}_i^T\delta_X\mathcal{A}_i^T$$

where the $\mathcal{A}$'s and $\mathcal{B}$'s are noncommutative rational expressions, functions of the symbols C, A_1, A_2, A_3, R_1, R_3, Q_1, Q_3, X. To find the update direction δ_X, we must be able to solve the algebraic linear system of equations given by

$$\mathbb{H}(\delta_X) = \mathbb{Q} \tag{4.78}$$

where $\mathbb{Q}$ is the gradient map obtained from the first directional derivative of $\phi_\gamma(X)$ along the direction δ_X. As already described, using the vec operation, the above system can be equivalently written as

$$\mathcal{H}v = g \tag{4.79}$$

where $\mathcal{H} = \sum_i^N \mathcal{B}_i^T \otimes \mathcal{A}_i + \sum_i^N \mathcal{A}_i \otimes \mathcal{B}_i^T$, $g = \mathrm{vec}(\mathbb{Q})$, and the unknown is now $v = \mathrm{vec}(\delta X)$.

We do not show the formulas for $\mathbb{H}(\delta_X)$ and $\mathbb{Q}$, since these expressions are quite large and will consume several pages. What is important is the fact that the formula for $\mathbb{H}(\delta_X)$ as computed originally, before applying any simplification rule, has 1014 Sylvester terms. However, after applying our NCCollectSylvester command, the Sylvester index decreases to just $N = 43$.

In order to numerically solve the linear system given in (4.79) one needs:

1. to substitute matrices for the symbols appearing in the expressions for the $\mathcal{A}_i$ and $\mathcal{B}_i$;
2. to evaluate the Hessian matrix $\mathcal{H}$ by applying $2N$ Kronecker products:

$$\mathcal{H} = \sum_i^N \mathcal{B}_i^T \otimes \mathcal{A}_i + \sum_i^N \mathcal{A}_i \otimes \mathcal{B}_i^T$$

These are the two main steps where collecting terms in the expression for $\mathbb{H}(\delta_X)$ can significantly affect the evaluation time.

Time saving obtained by applying NCCollectSylvester

To find out how much time is actually saved at the numerical level, the NCSDP code is executed using the collected formulas for $\mathbb{H}(\delta_X)$ with $N = 43$, and the not collected formula for $\mathbb{H}(\delta_X)$ with $N = 1014$. For this set of experiments, the size n of the matrices involved assume the following values $n = 4, 8, 16, 32, 64$. For each one of this size, we execute ten times the inner loop where the linear system (4.79) is numerically solved. We also measure the overall CPU time (over 10 iterations) spent on the above items 1 (formula evaluations) and 2 (Kronecker product). In this way, we can analyze how the time spent on formula evaluations behaves as a function of the size of the matrices involved in the expressions, as well as the Sylvester index.

The results are presented in Table 4.6, where C stand for the Collected case (the Sylvester index is $N = 43$), and NC stands for the Not Collected case (the Sylvester index is $N = 1014$). In this table, the row labeled "ratio" is the ratio between the Not Collect column and the Collect column. The time spent on solving the linear system, presented in the row labeled "Solv. System," is not affected by the expression being or not being collected. The other labels are as follows: MS for matrix size, SI for the Sylvester index, FE for formula evaluation, KP for Kronecker product, and LS for the linear solver.

MS	4		8		16		32		64	
SI	C	NC	C	NC	C	NC	C	NC	C	NC
FE	0.46	2.12	0.45	1.95	0.73	2.79	2.6	9	17	64
KP	0.02	0.28	0.04	0.55	0.90	14.05	23	226	1166	4467
LS	0.01	0.01	0.02	0.02	0.36	0.33	15	14	588	543
Ratio	NC/C		NC/C		NC/C		NC/C		NC/C	
FE	4.6		4.3		3.8		3.5		3.8	
KP	14		13.8		15.6		9.8		3.8	

Table 4.6: Timing (seconds): formulas evaluation, Kronecker products, and linear solver

The results provided in Table 4.6 show that collecting terms in the expression for the Hessian map $\mathbb{H}(\delta_X)$ represents a huge saving, since the average time spent on substituting matrices for the symbols that appear in the expressions for the $\mathcal{A}_i$ and $\mathcal{B}_i$ when the expressions are not collected is approximately four times longer than the time for the collected case (row labeled Formula Eval). Collecting the expressions significantly improves the time spent on evaluating Kronecker products: the timing improved by a factor of approximately 14 for matrices of dimension 16 and under. In this same range of matrix size, the overall time spent (over 10 iterations) on numerically solving the equation $\mathcal{H}v = g$ for the unknown v was relatively insignificant. However, for matrices of size 32 and over, the ratio between the Collected and Not Collected case for the time spent on Kronecker products decreases with the dimension of the matrices. For matrices of size 32 this ratio is 9.8, and for matrices of size 64 the ratio[18] goes down to 3.8. Moreover, the time spent on solving the linear subproblem becomes significantly larger than the time spent on substituting matrices for the symbols. The computer used for these experiments was a dual processor Pentium III (Coppermine), with 1004.530 MHz cpu clock, 4GB of RAM, 4GB of SWAP, running Linux (kernel 2.4.18-27.7.xsmp) and Matlab version 6.1.0.450 (R12.1).

We have just seen that for matrices of large size, the time spent on numerically solving the linear system of equations $\mathcal{H}v = g$ for the unknown v becomes large. To understand this fact better, suppose the dimension of the matrices involved is chosen to be $n = 32$. Thus, the unknown matrix X having size 32×32 implies that the unknown vector v and the system to be solved will have size $32^2 = 1024$. (Our implementation at this point does not take advantage of the symmetry). If one could solve the linear system of equations for δ_X in its original structured form given by $\mathbb{H}(\delta_X) = \mathbb{Q}$, without applying Kronecker products

[18] We believe that for matrices of dimension 32 and over, a considerable amount of time might be spent on allocating dynamically memory for the matrix $\mathcal{H}$ at each inner loop. This fact may have interfered with the ratio.

and keeping the dimensions of the linear system low, a huge saving on the numerical linear solver would probably be attained. This is an open area which we hope will be pursued by others.

4.6.5 Applying our NCCollectSylvester command to a variety of matrix inequalities

Another interesting experiment is to analyze how the Sylvester index behaves by applying our NCCollectSylvester command to a variety of formulas. We would like to find out how much reduction of the Sylvester index can be accomplished by applying our collect command to a variety of matrix inequalities which appear in control design. The example just presented, taken from Section 4.7, has shown a great improvement since the Sylvester index reduced from $N = 1014$ to $N = 43$. Now, two more examples are presented, so that the reader can have a more realistic understanding about how the process of collecting behaves on matrix inequalities that appear frequently in control problems.

Example 4.6.1 For the following standard Riccati inequality:

$$AX + XA^T - XRX + Q > 0$$

the Hessian map $\mathbb{H}(\delta_X)$ for the not collected case has a Sylvester index of $N = 20$, while the collected expression has a Sylvester index of $N = 6$.

Example 4.6.2 Now, a more realistic example is used: the mixed H_2/H_∞ control problem presented in Chapter 2:

$$\begin{gathered} \min \operatorname{Tr}\{Q\} \\ Q - (C_2X + D_{2u}F)X^{-1}(C_2X + D_{2u}F)^T > 0 \end{gathered} \tag{4.80}$$

$$\begin{aligned} AX + XA^T + B_uF + F^TB_u^T + B_wB_w^T + \\ \left[XC_1^T + F^TD_{1u}^T + B_wD_{1w}^T\right] R^{-1} \left[XC_1^T + F^TD_{1u}^T + B_wD_{1w}^T\right]^T < 0 \end{aligned}$$

with $R = \eta^2 I - D_{1w}D_{1w}^T > 0$.

For the above control problem, there are three unknowns denoted by $Q = Q^T$, $X = X^T$, and F (not symmetric). Thus, the linear subproblem to be solved will have dimension

3×3, and consequently each entry on this system will contain a Sylvester operator. For instance, the (1,1) entry will have an expression of the form

$$\sum_{i}^{N_{11}} \mathcal{A}_i^{11} \delta_Q \mathcal{B}_i^{11} + \sum_{i}^{\hat{N}_{11}} \hat{\mathcal{A}}_i^{11} \delta_Q^T \hat{\mathcal{B}}_i^{11}$$

The (1,2) entry will have the form

$$\sum_{i}^{N_{12}} \mathcal{A}_i^{12} \delta_F \mathcal{B}_i^{12} + \sum_{i}^{\hat{N}_{12}} \hat{\mathcal{A}}_i^{12} \delta_F^T \hat{\mathcal{B}}_i^{12}$$

The (1,3) entry will have the form

$$\sum_{i}^{N_{13}} \mathcal{A}_i^{13} \delta_X \mathcal{B}_i^{13} + \sum_{i}^{\hat{N}_{13}} \hat{\mathcal{A}}_i^{13} \delta_X^T \hat{\mathcal{B}}_i^{13}$$

The (2,1) entry is the transpose of the (1,2) entry. The (2,2) entry will have the form

$$\sum_{i}^{N_{22}} \mathcal{A}_i^{22} \delta_F \mathcal{B}_i^{22} + \sum_{i}^{\hat{N}_{22}} (\hat{\mathcal{A}}_i^{22})^T \delta_F^T (\hat{\mathcal{B}}_i^{22})^T$$

and so forth. It should be noticed that the Sylvester index $\hat{N}_{11}$, $\hat{N}_{12}$, and $\hat{N}_{22}$ are zero, since the corresponding variables Q and X are symmetric. For the MIs given above in (4.80), the set of Sylvester indexes N and $\hat{N}$ for the case where the Hessian map $\mathbb{H}(\delta_Q, \delta_X, \delta_F)$ was collected and was not collected is provided in Table 4.7.

In this Table 4.7, the variables X and F are associated with the entries

$$\Big[22, \quad 23, \quad 32, \quad 33\Big]$$

for each one of the subtables. If we only pay attention to the Sylvester index N, we see that the submatrix associated with X and F for the

not collected case $\begin{array}{|c|c|}\hline 73 & 33 \\ \hline 33 & 35 \\ \hline \end{array}$ reduces to only $\begin{array}{|c|c|}\hline 9 & 2 \\ \hline 2 & 2 \\ \hline \end{array}$ in the collected case.

Similarly, a large reduction is also obtained for the Sylvester index $\hat{N}$. Thus, for the variables X and F we found that a large reduction on the Sylvester index N and $\hat{N}$ is obtained after applying our NCCollectSylvester command. Naturally, this will represent a considerable saving on the evaluation time for the numerical linear solver.

Remark 4.6.1 *Another step is taken in order to improve the overall timing, and it is not related to the idea of simplifying expressions by collecting terms, but it is valuable. We look*

	Hessian $\mathbb{H}(\delta_Q, \delta_X, \delta_F)$ Not Collected					
	Sylv. index N_{ij}			Sylv. index $\hat{N}_{ij}$		
i,j	1	2	3	1	2	3
1	8	8	4	0	0	4
2	8	73	33	0	0	33
3	4	33	35	4	33	32

	Hessian $\mathbb{H}(\delta_Q, \delta_X, \delta_F)$ Collected					
	Sylv. index N_{ij}			Sylv. index $\hat{N}_{ij}$		
i,j	1	2	3	1	2	3
1	6	2	1	0	0	1
2	2	9	2	0	0	2
3	1	2	2	1	2	4

Table 4.7: Sylvester index N and $\hat{N}$ for the Collected and Not Collected cases

for inverses of matrices which appear inside the expressions for the Hessian map and we replace each occurrence by a new variable. In this way, all the inverses are evaluated only once at the beginning of the code. This can considerably improve the overall performance, since numerically evaluating an inverse of a matrix may consume a large amount of time, mainly for matrices of large dimensions.

It is also true that at the symbolic level of Mathematica, the process of collecting terms on an expression and the process of simplifying rational functions, can consume a considerable amount of time. However, this computation is done only once at the beginning of the run. This is in contrast with the numerical part, where solving the linear system to provide the update direction takes place at each inner iteration (which occurs several times). Therefore, the ability to collect factors in an expression (decreasing the Sylvester index) plays a very important role.

4.7 Numerical Experiments: Timing of the NCSDP Solver

The previous section has shown how the theory could be implemented for a simple example: the problem of finding feasible solutions to a Riccati inequality. For this purpose, we have shown the details of the derivations of the formulas for the update direction (which can also be done automatically using the NCAlg toolbox for Mathematica), we have made

available a Matlab code which implements those equations, and finally we have presented a few numerical results using this code. In this section, however, our main focus is to compare the timing of our NCSDP solver against available professional SDP solvers.

In this thesis, we have focused solely on convex problems, as we have made no effort in providing a reliable implementation of a nonconvex code based on the proposed methodology. However, with a simple modification of our convex code, basically by implementing a rudimentary line search and a "strategy" for dealing with indefinite Hessian, we were able to successfully run a few nonconvex examples. See the Appendix D for a collection of problems that our code was successful for.

4.7.1 The problem used in our tests

The optimization problem to be used in this section is the following eigenvalue minimization problem

$$\alpha^* = \min \{\alpha : (X, \alpha) \in \text{closure}(\mathcal{G})\} \tag{4.81}$$

where the feasibility set $\mathcal{G}$ is the convex domain given by

$$\mathcal{G} = \left\{(X, \alpha) \in \mathbb{S}^n \times \mathbb{R} : \alpha I - CXC^T > 0,\ F_3(X) > 0,\ F(X) > 0\right\}$$

with $F(X) = F_1(X) - F_2(X)F_3(X)^{-1}F_2(X)$ and the function $F_i(X)$ given by

$$F_i(X) = A_i X + X A_i^T - X R_i^{-1} X + Q_i$$

In our experiment we have set $F_2 := A_2^T X + X A_2$, thus $F(X)$ is given by

$$\begin{aligned} F(X) = {} & A_1 X + X A_1^T - X R_1^{-1} X + Q_1 \\ & - (A_2^T X + X A_2) \left(A_3 X + X A_3^T - X R_3^{-1} X + Q_3\right)^{-1} (A_2^T X + X A_2) \end{aligned} \tag{4.82}$$

The matrices C, A_1, A_2, and A_3 belong to $\mathbb{R}^{n\times n}$, the invertible matrices R_1, R_3, belong to $\mathbb{S}^n_{++}$ and the matrices Q_1, Q_3, and X belong to $\mathbb{S}^n$. In this example all matrices are square matrices of dimension $n \times n$. Note that by Schur complement techniques the above problem (4.81) can be equivalently restated as the following LMI problem

$$\begin{gathered} \alpha^* = \min \alpha \qquad \text{subject to} \\ \alpha I - CXC^T > 0 \\ 0 < \hat{F}(X) := \begin{bmatrix} A_1 X + X A_1^T + Q_1 & A_2^T X + X A_2 & X & 0 \\ A_2^T X + X A_2 & A_3 X + X A_3^T + Q_3 & 0 & X \\ X & 0 & R_1 & 0 \\ 0 & X & 0 & R_3 \end{bmatrix} \end{gathered} \tag{4.83}$$

It is clear that the feasibility set for both problems (4.81) and (4.83) are equivalent. However the dimensions of the equations for each one of the above problems are quite different. Since our NCSDP method is solely a primal method, the only dimension that counts is the dimension of the unknown matrix X, which is $N = \dim(\mathbb{S}^n) = n(n+1)/2$, on the other hand, for the primal-dual method, the dimension of the dual system has to be taken into account, which for this problem is four times bigger, being $4N$. This happens because the dimension of the dual variable is related to the dimension of the LMI in (4.83), more precisely, it is related to the dimension of the range of $\hat{F}(X)$. Therefore, for matrices of large size, the computation time will increase considerably.

4.7.2 An implementation of the method of centers for linear matrix inequalities

We will provide a simple implementation of the method of centers for LMIs, which is denoted by MCLMI. This implementation is based on the algorithm proposed in Colaneri et al. (1997). Then, we apply MCLMI to the LMI formulation (4.83), and compare its performance to our NCSDP code applied directly to the matrix inequality problem stated in (4.81). Since we are using the same method of centers for both codes, NCSDP and MCLMI, one point to be illustrated is how much improvement can be obtained by dealing with the constraint in its natural form (4.82) instead of in the LMI form (4.83).

For the particular case of LMIs, the implementation is easier, given that there is no need of symbolic computation to obtain the gradient vector and the Hessian matrix. These formulas are obtained by noting that any LMI can be represented in the following affine form in x:

$$\hat{F}(x) = A_0 + A_1 x_1 + A_2 x_2 + \cdots + A_N x_N$$

with A_i a symmetric matrix, N the dimension of X and $x \in \mathbb{R}^N$. Thus, for a suitable choice of matrices A_i the LMIs in (4.83) have this affine representation. For this implementation, one needs the assumption that the feasibility set $\mathcal{G} = \{X : \hat{F}(X) > 0\}$ is nonempty and bounded, which implies that the matrices $A_1, \ldots, A_N$ are linearly independent. This is a common assumption, and is verified in most of the control problems of interest. All those facts are quite standard and can be found in Boyd et al. (1994); Colaneri et al. (1997). From the above affine representation for $F(x)$, it can be shown that the gradient vector is given by

$$g_i = -\operatorname{Tr}\left\{F^{-1}A_i\right\}, \qquad i = 1, \ldots, N \tag{4.84}$$

and that the Hessian matrix is given by

$$\mathcal{H}_{ij} = \mathrm{Tr}\left\{F^{-1}A_iF^{-1}A_j\right\}, \qquad i = 1,\ldots,N, \quad j = 1,\ldots,N \tag{4.85}$$

In this case, the update direction v is the solution of the linear system

$$\mathcal{H}v = g \tag{4.86}$$

4.7.3 Timing of NCSDP against MCLMI

Before comparing our NCSDP solver with other SDP solvers, we show how the time is spent among the main parts of the code. There are two distinct parts: The symbolic computation, which is done in Mathematica, and the numeric part, which is done in Matlab.

Inside the numeric part of our NCSDP code, we will be timing: 1) The time spent on evaluating the Sylvester terms. That means, the time Matlab spent on calculating the terms $\mathcal{A}_i$, $\mathcal{B}_i$, and $\mathbb{Q}$. 2) The time spent on building the Hessian matrix $\mathcal{H}$. This is the time spent on the Kronecker products. 3) And finally, the time spent on solving the linear system $\mathcal{H}v = g$ for v.

The results for this first numerical experiment are presented in Table 4.8 for the MCLMI code (the LMI implementation), and in Table 4.9 for the NCSDP code. In those tables, the first column n shows the dimension of the unknown matrix X (all matrices in the formula for $F(X)$ have the same dimension $n \times n$). The second column IT/NeNe shows the total number of outer iterations required to achieve the objective within an accuracy of 10^{-5}, and the total number of Newton steps required to compute the analytic center within an accuracy of 10^{-3}. For the computation of the analytic center, the line search plays an important role. For the LMI case, the MCLMI code, the suboptimal line search given in (4.46) has been used. The NCSDP code implements the Nesterov-Nemirosvky step length given in (4.45). The last three columns in Table 4.8 present the CPU time spent on computing: the gradient g, the Hessian $\mathcal{H}$, and solving the linear system for v (given respectively by (4.84)-(4.86)). In Table 4.9, the column FE, formula evaluations, shows the time spent on computing the Sylvester terms, since this time may be large in some cases[19], the column KP presents the time spent on the Kronecker product, and column v presents the time spent on the linear solver. The starting feasible point was the same for all the experiments.

[19]The time spent on computing the Sylvester terms are usually large when the corresponding symbolic expressions are also large. So, it might be convenient to simplify the formulas symbolically.

n	IT/NeNe	g	$\mathcal{H}$	v
1	12 / 63	3.7E-02	4.7E-02	5.2E-03
2	9 / 42	4.6E-02	1.0E-01	4.9E-03
4	8 / 45	2.4E-01	2.0E+00	1.1E-02
8	8 / 45	4.4E+00	1.7E+02	4.0E-02
16	7 / 41	1.1E+02	1.6E+04	6.0E-01
32	–	–	–	–
64	–	–	–	–

Table 4.8: MCLMI

n	IT/NeNe	FE	KP	v
1	48 / 144	8.4E-01	3.1E-01	3.0E-02
2	33 / 99	1.4E+00	1.1E-01	7.0E-02
4	43 / 129	1.9E+00	3.0E-01	8.0E-02
8	38 / 114	2.3E+00	5.7E-01	3.0E-01
16	30 / 90	3.3E+00	1.8E+01	3.7E+00
32	27 / 81	1.1E+01	2.4E+02	1.0E+02
64	18 / 90	7.1E+01	3.0E+03	2.9E+03

Table 4.9: NCSDP

For the LMI case, we did not run the code for matrices of dimension 32×32 and greater, since it would take more than 100 days, as one can conclude by extrapolating the data on Table 4.8. In this table, one finds that the most expensive part, for the LMI implementation, is the evaluation of the Hessian matrix. This can be seen from the formulas for the gradient and for the Hessian, given in (4.84)–(4.85). Thus, in order to obtain the gradient one needs to evaluate approximately N trace operations and N matrix multiplications. Recall that $N = n(n+1)/2$ is the dimension of the space $\mathbb{S}^n$. To compute the Hessian matrix, one needs to evaluate approximately $N^2/2$ trace operations and $3/2N^2$ matrix multiplications. Thus, the time spent on the Hessian is over $N^2/2$ the time spent on calculating the gradient, which agrees with the results presented in Table 4.8. For the NCSDP code, as seen from column FE on Table 4.9, the most expensive part for matrices of small size is the time spent on evaluating the Sylvester terms $\mathcal{A}_i$, $\mathcal{B}_i$, and $\mathbb{Q}$. However, when the size of the matrices increases above 8, the time spent on Kronecker products, column KP, and the time spent on solving the linear system, column v, begin to dominate.

4.7.4 Timing of NCSDP against others SDP solvers

For a second set of experiments, the NCSDP code is compared to many available semidefinite programing solvers, most of them are Primal-Dual methods. The results are presented in Figure 4.3. The LMILab toolbox, which is the only implementation based on the projective method of Gahinet et al. (1995), is one of the most widely used solvers for linear matrix inequalities. It has a GUI editor for interactive problem specification. From Table 4.10 and Figure 4.3, one sees that for the eigenvalues minimization problem stated in (4.81), the LMILab was the fastest code. As the size of the matrices increases, our NCSDP code approximates LMILab. And probably, for matrices of dimensions larger than 64×64, NCSDP may be faster than the LMILab solver. We did not run this experiment since the time would be significantly long.

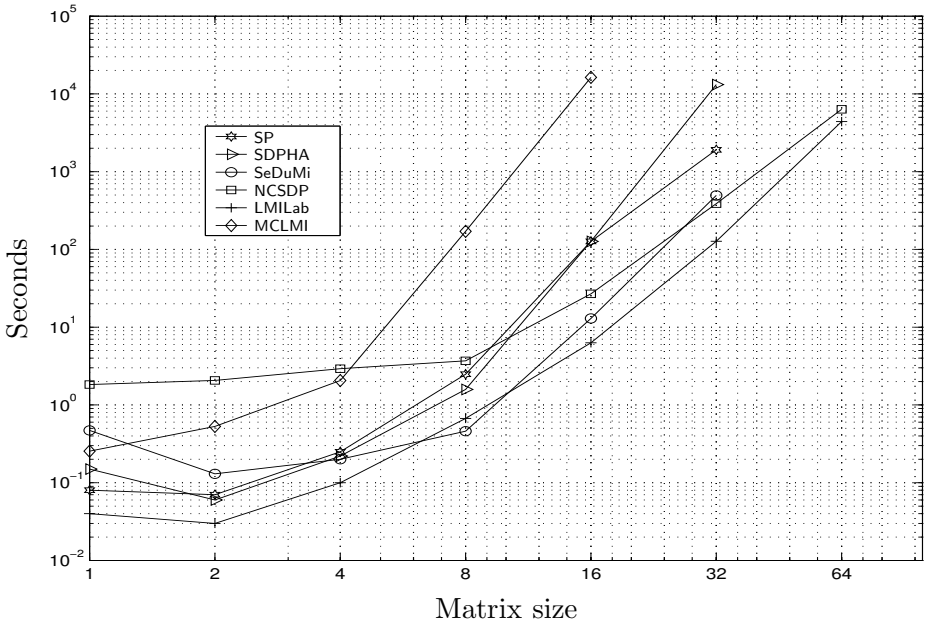

Figure 4.3: Performance of the LMI Solvers

At this stage, it is important to emphasize that while our NCSDP code is "completely"[20] implemented using Matlab functions, most of the other solvers have their core subroutines written in Fortran or C. The fact that either piece of the code or the whole code is compiled, significantly improves the overall performance. The performance can increase by a factor of 10, or even more. The implementation of an efficient line search is also important and it significantly improves the overall performance.

[20]The part that manipulates the Kronecker product was implemented in C, since the Matlab command kron.m was very inefficient.

n	cpu time
1	4.00000E-02
2	3.00000E-02
4	1.00000E-01
8	6.70000E-01
16	6.29000E+00
32	1.27250E+02
64	4.40163E+03

Table 4.10: LMILab

n	cpu time
1	3.00000E-02
2	1.00000E-02
4	3.00000E-02
8	5.20000E-01
16	2.62000E+01
32	8.14900E+02
64	–

Table 4.11: SP

The performance of the SP code is shown on Table 4.11. This code implements the Nesterov and Todd's primal-dual potential reduction method (Vandenberghe and Boyd (1995)). The code is written in C/C++ with calls to BLAS and LAPACK. It was one of the first software tools that was developed for semidefinite programing. Table 4.12 presents the SDPpack code (Alizadeh et al. (1998)), which was implemented using Matlab MEX files. It is a primal-dual path following method, which implements $XZ + ZX$ search direction, Mehrotra predictor-corrector, and other specialized routines. Table 4.13 presents the SDPHA code (Fujisawa et al. (1997)), which is another primal-dual path following method that uses Mehrotra predictor-corrector. And finally Table 4.14 presents the SeDuMi code from Sturm (1999). This is a recent code which implements the self-dual embedding technique for optimization over self-dual homogeneous cones.

n	cpu time
1	8.000E-01
2	7.000E-02
4	2.500E-01
8	2.480E+00
16	1.269E+02
32	1.907E+03
64	–

Table 4.12: SDPpack

n	cpu time
1	1.500E-01
2	6.000E-02
4	2.200E-01
8	1.580E+00
16	1.270E+02
32	1.314E+04
64	–

Table 4.13: SDPHA

n	cpu time
1	4.700E-01
2	1.300E-01
4	2.000E-01
8	4.600E-01
16	1.295E+01
32	4.951E+02
64	–

Table 4.14: SeDuMi

Since the above codes are for general SDP problems, where the data should be expressed in a "standard" SDP form, which is not particularly the standard LMI form or even an LMI matrix representation (which appears frequently in engineering), we make use of

the Matlab package LMITOOL, and its graphic GUI editor TKLMITOOL, to act as an interface for the above SDP codes. In this way, these interfaces provided all the necessary conversions from LMI to the SDP form. We should make it clear that the timing presented does not incorporate the time consumed by these interfaces. However, for matrices of size 64 or larger, we were not able to run LMITOOL, since Matlab runs out of memory. We believe this is not a lack of RAM/Swap memory but rather a Matlab inefficiency on managing a large amount of memory. The computer used was a dual processor Pentium III (Coppermine), with 1004.530 MHz cpu clock, 4GB of RAM, 4GB of SWAP, running Linux (kernel 2.4.18-27.7.xsmp) and Matlab version 6.1.0.450 (R12.1).

Chapter 5

Convexifying Method for Integrating Structure and Control Design

5.1 Introduction

The history of structure design can be characterized by four eras: In the first era, the design sought simply to oppose gravity – a statics problem. In the second era, the dynamic response was important. The third era sought to add control features to an existing structure design. In the fourth era, the design of the structure and the design of the controller are integrated so that the dynamics of the control system and the dynamics of the structure are cooperating, rather than competing, to reduce a selected performance objective. During the last two decades, the mathematical tools of control theory have produced algorithms which allow one to bound the dynamic response given a class of uncertain time varying disturbances. Such tools can now be used for structure design, even if no control is involved. In this context our approach for structure design allows performance bounds on the dynamic response of the output, whereas the more standard structural design code focus on the static response and eigenvalues.

It is a well known fact that the design of the structure and the design of a controller for a given system are not independent. Consequently, it might happen that both the control design and the structural design are competing with each other in order to achieve some prescribed dynamic behavior. That is, more control energy than necessary might be

required to achieve the objectives. This is an important consideration, for instance, in the control of civil structures since the apparent necessity of large control forces have impeded the acceptability of control as a viable method to impose structure design. Simultaneous design (Grigoriadis and Skelton (1998); Housner et al. (1997); Skelton et al. (1992)) can significantly improve the overall performance of the system, in the sense that either the new system (designed by this "hybrid" approach) would need a smaller amount of control energy to attain the same performance, or the new system would provide superior performance than the systems designed by standard techniques. Unfortunately, the simultaneous design of structure and controller is not in practice tractable, and results in a nonconvex optimization problem. The available algorithms are computationally expensive (see Grandhi (1989); Jin and Sepulveda (1995); Onoda and Haftka (1987); Yang and Chen (1996) and references therein), without guaranteeing a local minimum. It can be shown that the integrated structure and control design problem is equivalent to a decentralized output feedback control problem, which is well known to be hard to solve.

Following a two-step redesign approach, one idea extensively used by Grigoriadis et al. (1996) and Skelton and Kim (1992) was as follows. In the first step, a controller for a given nominal structure was designed to meet some prescribed closed loop performance bound Ω. In a second step, the structure and the controller were simultaneously redesigned in order to minimize the active control energy subject to the constraint that the closed loop system matrix is kept constant. This preserves the same level of performance γ from one iteration to the next. The feature that makes the joint structure/control problem convex is the constraint that holds the closed loop system matrix constant. Based on this idea, the algorithm for solving the integrated control and structure problem can be stated as: i) for a given nominal structure, design the controller; ii) redesign structure and controller (keeping the closed loop plant matrix constant); iii) with this new plant return to step i). This constraint on the closed loop system reduces the redesign (iteration ii) to a constrained convex quadratic programming problem. This was a significant improvement over the existing methodologies. A further improvement was given in Lu and Skelton (2000), where the authors considered more general structures and used the mixed H_2/H_∞ performance criteria via a Linear Matrix Inequality (LMI) framework, but in the redesign step they still needed the convexifying constraint of matching the system matrix.

A more direct approach via LMI to deal with the integrated structure and control design that does not impose constraints in the closed loop system matrices was used in Grigoriadis and Skelton (1998) and Grigoriadis and Wu (1997). The techniques proposed solve the two-step redesign procedure by iterating between two convex subproblems posed

as LMIs. The algorithm can be summarized as follows: while keeping the parameters of the structure fixed, solve a convex control problem; and in a second step, fix the Lyapunov matrix (which provides the controller) and optimize for the parameters of the structure. This approach has the drawback that it does not allow the mass matrix to be optimized and it may have a slow convergence, since the Lyapunov matrix in the structure redesign step is fixed.

There are few techniques available in the literature that allow one to treat the mass as an uncertain parameter (Grigoriadis et al. (1996); Housner et al. (1997); Jin and Sepulveda (1995); Skelton et al. (1992), among others), though these techniques are not in the LMI framework. Our algorithm has also the advantage of optimizing directly the physical parameters of the structure instead of optimizing an uncertain matrix ΔA (Grigoriadis and Skelton (1998); Hsieh (1992)), and at the end of the redesign, trying to find suitable physical parameters that match this uncertain matrix ΔA, which might not exist.

This chapter presents a new theory for the simultaneous design of structure and controller that improves the existing methodologies in two different ways. Our approach is completely posed in the LMI framework, so many different type of convex performances and convex constraints can be incorporated. The proposed methodology is an improvement over the result given in Grigoriadis and Skelton (1998) and Grigoriadis and Wu (1997) since we do not constrain the Lyapunov matrix to be fixed in any step of the algorithm. The method also allows one to optimize the mass parameter of the system. More precisely, it allows one to optimize any parameter that appears affinely in any of the system matrices.

Here we define some notation that will be used in this chapter. The superscript $(\cdot)^T$ and $(\cdot)^{-1}$ means respectively the transpose and the inverse of a matrix. The operator $\mathrm{Tr}\,\{\}$ is the usual trace of a matrix. The operator $\mathrm{diag}(\alpha_1, \ldots, \alpha_n)$ stand for a diagonal matrix whose entries are the elements $\alpha_1, \ldots, \alpha_n$. The function $\mathcal{E}[\cdot]$ is the expectation operator.

5.2 Problem Statement

A large class of dynamic systems in the field of mechanics and structures can be represented by a second-order differential equation of the form

$$M\ddot{q} + D\dot{q} + Sq = f(t), \tag{5.1}$$

where $q \in \mathbb{R}^n$ is the vector of generalized coordinates, $M \in \mathbb{R}^{n\times n}$ is the mass matrix, $S \in \mathbb{R}^{n\times n}$ is the stiffness matrix, and $D \in \mathbb{R}^{n\times n}$ is the damping matrix (with only viscous

damping force $D = D^T$, but if gyroscopic terms are presented, then D is not symmetric). The mass matrix is assumed to be symmetric and positive define, $M = M^T > 0$. The vector $f(t)$ is an external force applied to the system. For our purpose, this force will represent the control input and the disturbance input actuating on the system. So $f(t)$ takes the form

$$f(t) = \hat{B}_u u(t) + \hat{B}_w w(t),$$

where $u(t)$ is the control signal to be determined, and $w(t)$ is the exogenous disturbance. We cast our problem in the stochastic framework so that $w(t)$ is a white noise process. However, equivalent results apply when $w(t) \in \mathcal{L}_2$ (meaning that $w(t)$ is bounded in the sense of two norm. In other words $w(t)$ has a finite power spectrum).

Using a convenient change of variables given by $x = \begin{bmatrix} q^T & \dot{q}^T \end{bmatrix}^T$, the second-order differential equation (5.1) is promptly written in the state space form

$$\begin{bmatrix} I & 0 \\ 0 & M \end{bmatrix} \dot{x} = \begin{bmatrix} 0 & I \\ -S & -D \end{bmatrix} x + \begin{bmatrix} 0 \\ \hat{B}_u \end{bmatrix} u + \begin{bmatrix} 0 \\ \hat{B}_w \end{bmatrix} w$$

or equivalently

$$E\dot{x} = Ax + B_u u + B_w w. \tag{5.2}$$

This is a *descriptor representation* for this system. This form is frequently adopted when the matrix E is not invertible.

In our approach for the integrated control and structure design problem, the parameters of the structure which are available for optimization appear in the mass, the damping, and the stiffness matrices. A very important property assumed here is that the system equation (5.1) is affine in these parameters. By this affine representation we mean that

$$M(\eta) = M_0 + \sum_s \eta_s M_s, \quad D(\beta) = D_0 + \sum_j \beta_j D_j, \quad \text{and} \quad S(\gamma) = S_0 + \sum_k \gamma_k S_k.$$

where matrices M_s, D_j, and S_k are given. Since matrix A in (5.2) is affine in D and in S, and matrix E is affine in M, the system (5.2) can also be written as

$$E(\alpha)\ \dot{x} = A(\alpha)x + B_u u + B_w(\alpha)w,$$

for $A(\alpha)$ and $E(\alpha)$ affine matrices given by

$$A(\alpha) = A_0 + \sum_i \alpha_i A_i, \qquad E(\alpha) = E_0 + \sum_i \alpha_i E_i, \qquad B_w(\alpha) = B_{w0} + \sum_i \alpha_i B_{wi},$$

where the variable α contains, in a convenient way, the variables η, β, and γ, i.e., $\alpha = (\eta, \beta, \gamma)$. Notice that it is paramount to adopt the descriptor representation (5.2) in order to preserve the affine property of the mass matrix M, consequently, allowing the mass of the system to be incorporated into the optimization problem.

Remark 5.2.1 *To apply our methodology we do not need explicitly to assume the second-order representation. Any descriptor system which is affine in the parameters can be used. We use the second-order representation since we are mainly concerned with mechanical systems and structures.*

5.2.1 The integrated structure and control problem

For simplicity of presentation, we first present the result for the full state feedback case, with the control gain given by $u(t) = Kx(t)$. Later, in Section 5.5 and Section 5.6, we present the derivation for the static output and dynamic feedback case, which does not require much more sophistication. The output vector for performance evaluation is

$$z(t) = C_z x(t). \tag{5.3}$$

We first present Theorem 5.2.2 which characterizes the control problem for the structure and control design, with a stochastic interpretation; later, we show its equivalence to the standard H_2 problem. The exogenous disturbance $w(t)$ applied to the system is assumed to be a stochastic white noise process with intensity $W = W^T > 0$, i.e., $\mathcal{E}[w(t)w(\tau)^T] = W\delta(t-\tau)$. Our performance criteria is to minimize the variance of the control $u(t)$ applied to the system, while the output $z(t)$ is bounded in the sense

$$\lim_{t\to\infty} \mathcal{E}[z(t)z(t)^T] < \Omega, \tag{5.4}$$

for some given positive definite matrix Ω.

Theorem 5.2.2 *Assume that the disturbance $w(t)$ is a stochastic white noise process with intensity $W = W^T > 0$. Define $F = KP$. Let Ω be a given positive definite matrix, and consider the descriptor system given in* (5.2)*. Then the following statements are equivalent:*

(i) There exists structure parameter α, and a stabilizing state feedback gain $u(t) = Kx(t)$ such that

$$\lim_{t\to\infty} \mathcal{E}[z(t)z(t)^T] < \Omega$$

and

$$\lim_{t\to\infty} \mathcal{E}[u(t)^T u(t)] < \gamma.$$

(ii) There exists matrices of compatible dimensions $P = P^T > 0$, $U = U^T > 0$, and F, and parameter α, such that the following inequalities are satisfied

$$\begin{bmatrix} A(\alpha)PE(\alpha) + E(\alpha)PA(\alpha)^T + B_u FE(\alpha) + E(\alpha)F^T B_u^T & B_w(\alpha) \\ B_w(\alpha)^T & -W^{-1} \end{bmatrix} < 0, \tag{5.5}$$

$$\mathrm{Tr}\,\{U\} < \gamma, \qquad \begin{bmatrix} U & F \\ F^T & P \end{bmatrix} > 0, \qquad C_z P C_z^T < \Omega. \tag{5.6}$$

(iii) For some constant matrix Z, there exists matrices of compatible dimensions $Q = Q^T > 0$, $U = U^T > 0$, and K, and parameter α, such that the following LMI are satisfied

$$\begin{bmatrix} (*) & B_w(\alpha) & A(\alpha) + B_u K & E(\alpha) \\ B_w(\alpha)^T & -W^{-1} & 0 & 0 \\ A(\alpha)^T + K^T B_u^T & 0 & -Q & 0 \\ E(\alpha)^T & 0 & 0 & -Q \end{bmatrix} < 0, \tag{5.7}$$

$$\mathrm{Tr}\,\{U\} < \gamma, \qquad \begin{bmatrix} U & K \\ K^T & Q \end{bmatrix} > 0, \qquad \begin{bmatrix} \Omega & C_z \\ C_z^T & Q \end{bmatrix} > 0. \tag{5.8}$$

where $(*)$ refers to the term

$$(-A(\alpha) + B_u K - E(\alpha))\mathsf{Z}^T - \mathsf{Z}(A(\alpha) + B_u K - E(\alpha))^T + \mathsf{Z}Q\mathsf{Z}^T$$

We present the proof of this theorem in Section 5.4.1, after the necessary tools provided by the convexifying algorithm have been introduced.

If the matrices A and E do not depend on the structure parameter α, then the constraint (5.5) in item (ii) is an LMI, hence a convex set, in U, P, and F. In other words, if the structure is known, the problem reduces to a standard convex state feedback control problem. If the matrices A and E depend on α, then the product $A(\alpha)PE(\alpha) + B_u FE(\alpha)$ is nonlinear in the decision variables α, F, and P. In this case, it is hard to find a solution. Even for the pure structural passive design case, where the control gain K is given, the problem is still nonconvex.

Note that independently of the control parameter K, the product of the system matrix $A(\alpha)$ and the "Lyapunov" matrix P is always present in (5.5). When the mass matrix M is fixed (matrix E does not depend on the parameter α), the procedure adopted in

Grigoriadis and Skelton (1998) and Grigoriadis and Wu (1997) is to iterate between two convex subproblems: first, for fixed structure parameter α solve for the Lyapunov matrix P in (5.5); second, for fixed matrix P solve for the parameter of the structure α in (5.5). This strategy in practice converges slowly to a solution, although there are no guarantee for a local minimum.

The algorithm we propose in this chapter also iterates between two subproblems, but in a more elaborate way. Before iterating, we apply some convexifying potential functions (see Definition 5.3.7) to the nonconvex constraint in order to generate the conditions in item (*iii*). Notice that, for a constant matrix Z, these conditions are simultaneously affine in the variables Q, U, K and α. In this sense the joint structure/control problem has been "convexified." Therefore, there will be no need to fix the Lyapunov matrix P in the redesign step (instead, the fixed matrix will be the added potential matrix Z). The convexifying potential method and the algorithm will be detailed in the next Section.

Remark 5.2.3 *The relation between the control problem presented in Theorem 5.2.2 and the H_2 control control problem is stated in the next lemma. See Boyd et al. (1994); Skelton et al. (1998) for a proof.*

Lemma 5.2.4 (H_2 Control Problem) *Assume that the disturbance w belongs to space $\mathcal{L}_2$. Then the H_2 norm of the closed loop transfer function*

$$H_{wz}(s) := C_z[sI - E(\alpha)^{-1}A_{cl}(\alpha)]E(\alpha)^{-1}B_w(\alpha)$$

where $A_{cl}(\alpha) := A(\alpha) + B_u K$, is bounded by $\sqrt{\mathrm{Tr}\,\{\Omega\}}$, i.e, $||H_{wz}(s)||_2^2 < \mathrm{Tr}\,\{\Omega\}$ if and only if the constraints (5.5–5.6) and (5.7–5.8) in Theorem 5.2.2 are feasible for $\gamma \to \infty$.

5.3 The Theory Behind the Convexifying Algorithm

This section describes the convexifying algorithm, which is a practical tool for solving control problems with structure imposed on the controller. It was shown in de Oliveira et al. (2000) that many standard control problems such as H_2 and H_∞ problems with some imposed structure in the controller can be formulated as an LMI problem having an extra nonconvex constraint. While most algorithms in the literature are aimed at the feasibility problem, this new algorithm enable us to pursue the improvement of solutions for suboptimal control optimization problems that are available.

We shall now introduce some notation and definitions which will be used throughout this section. The main reference for the optimality condition presented in this section is Luenberger (1969) and Maurer and Zowe (1979). In their setup, they have consider quite general topological spaces (Banach spaces). However, since our setup has an Euclidean structure, we present the theorems in their original setting, and later, we specialize them to our finite dimensional case. For vector valued functions, see Canon et al. (1966) and Mangasarian (1994). The topology and the definitions of derivatives we shall need are precisely described in Section 4.2.3 from Chapter 4. Thus, we do not repeat them here. In this topology, the inner product of two matrices A and B is

$$\langle A, B\rangle = \operatorname{Tr}\left\{AB^T\right\},$$

and the norm induced by this inner product is $\|X\| = \sqrt{\operatorname{Tr}\{XX^T\}}$. The notation $\mathcal{W}^\star$ stand for the dual space of $\mathcal{W}$ (the space of all bounded linear operators on $\mathcal{W}$). The composition of two functions $f_1(f_2)(x)$ is also denoted by $f_1(y) \circ f_2(x)$.

Definition 5.3.1 (Positive cone) *Let $\mathcal{K}$ be a closed convex cone in a real Banach*[1] *space $\mathcal{W}$ with vertex at the origin, which is also pointed and proper ($\mathcal{K} \bigcap \{-\mathcal{K}\} = \{0\}$). For $x, y \in \mathcal{W}$, we write $x \geq y$ (with respect to $\mathcal{K}$) if $x - y \in \mathcal{K}$. The cone $\mathcal{K}$ defining this ordering is called the* **positive cone** *in $\mathcal{W}$. The dual (or polar) cone of $\mathcal{K}$, which we denoted by $\mathcal{K}^\circ$, is defined as*

$$\mathcal{K}^\circ := \{\ell \in \mathcal{W}^\star : \ell(k) \geq 0 \quad \textit{for all} \quad k \in \mathcal{K}\}.$$

Evidently, if $\mathcal{W}$ is taken to be $\mathbb{S}^n$, the cone $\mathcal{K}$ induces the natural ordering on the set of positive semidefinite matrices, i.e, for matrices X and Y in $\mathbb{S}^n$, we write $X \leq Y$ if $Y - X \in \mathbb{S}^n_+$, and we write $X < Y$ if $Y - X$ lies in $\mathbb{S}^n_{++}$, the set of positive definite matrices (with analogous definition for $\geq$ and $>$).

Definition 5.3.2 (Regular point) *Let $\mathcal{V}$ and $\mathcal{W}$ be real Banach spaces. Let $\mathcal{K}$ be a positive cone satisfying Definition 5.3.1. Let $G(x)$ be a mapping from $\mathcal{V}$ to $\mathcal{W}$. Assume the Fréchet derivative of $G(x)$, denoted by $G'(x)$, exists at $\bar{x} \in \mathcal{V}$. Then $\bar{x}$ is said to be a* **regular point** *with respect to the constraint $G(x) \leq 0$ if $G(\bar{x}) \leq 0$ and there is a $\delta_x \in \mathcal{V}$ such that*

$$G(\bar{x}) + G'(\bar{x})\delta_x < 0.$$

[1] *A Banach space is a complete normed space (Reed and Simon (2000)).*

This condition implies the assumption used in Maurer and Zowe (1979):

$$0 \in \operatorname{int}\left\{G(\bar{x}) + G'(\bar{x})\delta_x + k : \delta_X \in \mathcal{V},\ k \in \mathcal{K}\right\},$$

where int *denotes the topological interior.*

This regularity condition essentially eliminates the possibility of the constraint boundary forming a cusp at a point. It also exclude the possibility of incorporating equality constraints by reducing the cone to a point or by including a constraint and its negative.

The next Theorem 5.3.3 provides the first-order necessary optimality conditions for the minimization problem stated in (P). This theorem is presented in Luenberger (1969) [9.4, Theorem 1] and Maurer and Zowe (1979) [3, Theorem 3.2].

Theorem 5.3.3 (Kuhn-Tucker theorem) *Let $\mathcal{V}$ and $\mathcal{W}$ be real Banach spaces. Let $\mathcal{K}$ be a positive cone in $\mathcal{W}$ which has a nonempty interior. Let $f(x) : \mathcal{V} \to \mathbb{R}$ and $G(x) : \mathcal{V} \to \mathcal{W}$ be Fréchet differentiable. Suppose x^* is a solution of*

$$\min f(x) \quad \textit{subject to} \quad G(x) \leq 0 \tag{P}$$

and that x^* is a regular point with respect to the constraint $G(x) \leq 0$. Then, there is a linear functional $\ell \in \mathcal{K}^\circ$ such that

$$\begin{aligned} f'(x^*) + \ell \circ G'(x^*) &= 0 \\ \ell \circ G(x^*) &= 0. \end{aligned} \tag{5.9}$$

Remark 5.3.4 *In this section we take $\mathcal{V} \subset \mathbb{R}^{p\times q}$ and $\mathcal{W}$ to be the space of all symmetric matrices $\mathbb{S}^n$ with $\mathcal{K}$ the usual cone of positive semidefinite matrices $\mathbb{S}^n_+$. For this particular case, using the Riesz representation Theorem[2], the above condition* (5.9) *reduces to*

$$\begin{aligned} f'(x^*) + \langle G'(x^*), \psi\rangle &= 0 \\ \langle G(x^*), \psi\rangle &= 0, \end{aligned} \tag{5.10}$$

with ψ a self-adjoint positive semidefinite matrix, i.e., $\psi \in \mathbb{S}^n_+$.

The next Theorem 5.3.5, from Maurer and Zowe (1979) [5, Theorem 5.2], gives a second-order sufficient condition for a local minimum of the minimization problem stated above in (P), when the spaces $\mathcal{V}$ and $\mathcal{W}$ are assumed to be of finite dimension.

[2] *See footnote on page 100.*

Theorem 5.3.5 (Second-order sufficient condition) *Let $\mathcal{P}$ denote the set of feasible points for the minimization problem* (P). *Let $x^* \in \mathcal{P}$. Let the following set*[3]

$$\mathcal{K}_G = \{k - \lambda G(x^*) : \lambda \in \mathbb{R},\ k \in \mathcal{K}\}$$

be closed and suppose $L(x) = f(x) + \ell \circ G(x)$ is a Lagrangian for (P) *at x^*. Assume the second Fréchet derivative of $f''(x)$ and $G''(x)$ exist at x^*. If*

$$L''(x)(\delta_x, \delta_x) > 0 \quad \textit{for all} \quad \delta_x \in \mathcal{B}, \quad \delta_x \neq 0$$

with $\mathcal{B}$ given by

$$\mathcal{B} = \left\{\delta_x \in \mathcal{V} : -G'(x^*)\delta_x \in \mathcal{K}_G\right\} \bigcap \left\{\delta_x : f'(x^*)\delta_x = 0\right\},$$

then there are $\alpha > 0$ and $\rho > 0$ such that $f(x) \geq f(x^*) + \alpha\|x - x^*\|^2$ for all $\{x : G(x) \leq 0\}$ with $\|x - x^*\| \leq \rho$.

The interpretation for the above finite dimensional case, when $f(x) : \mathbb{R}^n \to \mathbb{R}$ and $G(x) : \mathbb{R}^n \to \mathbb{R}^m$, is that the Hessian $L''(x)$ of the Lagrangian $L(x) = f(x) + \sum_{i=1}^m \lambda_i G_i(x)$ must be positive definite on the set of those nonzero directions

$$\delta_x \in \{\delta_x : G_i'(x^*)\delta_x \leq 0 \text{ for } i \in I_1 \text{ and } G_i'(x^*)\delta_x = 0 \text{ for } i \in I_2\}$$

with

$$I_1 = \{i : G_i(x^*) = 0,\ \lambda_i = 0\}, \quad \text{and} \quad I_2 = \{i : G_i(x^*) = 0,\ \lambda_i \neq 0\}.$$

Definition 5.3.6 (Potential matrix function) *Let $\mathcal{V} \subset \mathbb{R}^{p\times q}$. Let the matrix function $H(x, \xi) : \mathcal{V} \times \mathcal{V} \to \mathbb{S}^n$ has a Fréchet derivative $H'(x, \xi)$ in x defined for all $x, \xi \in \mathcal{V}$. Then $H(x, \xi)$ is called a* **potential matrix function** *if the following conditions are satisfied:*

i) the matrix $H(x, \xi)$ is positive semidefinite for all $x, \xi \in \mathcal{V}$;

ii) for all $x, \xi \in \mathcal{V}$ satisfying $\|x - \xi\| < \delta$, there exists $\epsilon > 0$ such that $H(x, \xi) \leq \epsilon\|x - \xi\|$;

iii) for all $x, \xi \in \mathcal{V}$ satisfying $\|x - \xi\| < \delta$, there exists $\epsilon > 0$ such that $H'(x, \xi) \leq \epsilon\|x - \xi\|$.

We are especially interested in potential functions with the following property.

[3] *The set $\mathcal{T} = \{\delta_x \in \mathcal{V} : -G'(x^*)\delta_x \in \mathcal{K}_G\}$ is called the linearizing cone of $\mathcal{P}$ at x^*.*

Definition 5.3.7 (Convexifying function) *Let $\mathcal{V} \subset \mathbb{R}^{p\times q}$. A potential matrix function $H(x,\xi) : \mathcal{V} \times \mathcal{V} \to \mathbb{S}^n$ defined for all $x, \xi \in \mathcal{V}$ is said to be a* **convexifying function** *if, for a given $\xi \in \mathcal{V}$, the function $H(x,\xi)$ added to the nonconvex matrix function $G(x) : \mathcal{V} \to \mathbb{S}^n$ makes the expression*

$$G(x) + H(x,\xi)$$

a convex[4] *matrix function in x for all $x \in \mathcal{V}$.*

We will be looking for potentials that are able to "convexify" a given nonconvex matrix function $G(x) : \mathcal{V} \subset \mathbb{R}^{p\times q} \to \mathbb{S}^n$. However, there might exist many candidates for such convexifying functions. Independently of a particular choice of the convexifying function, we state a simple algorithm to find suboptimal solutions to the nonconvex optimization problem

$$\min_{x\in\Omega} f(x), \quad \Omega := \{x \in \mathcal{V} : G(x) \leq 0\}. \tag{5.11}$$

Without loss of generality, we can assume $f(x)$ to be linear. We also assume that $G(x)$ is a nonconvex matrix function and that the set Ω is compact and has a nonempty interior.

Algorithm 5.3.8 *Let $\epsilon > 0$, $x^0 \in \Omega$ and a convexifying potential matrix function $H(x,\xi) : \mathcal{V} \to \mathbb{S}^n$ be given:*

1. *For $k = 0, 1, 2, \ldots$, solve the convex optimization problem*

$$x^{k+1} = \arg\min_{x\in\Omega_k} f(x), \quad \Omega_k := \left\{x \in \mathcal{V} : G(x) + H(x, x^k) \leq 0\right\}. \tag{5.12}$$

The above convex problem is significantly simpler than (5.11), and we assume that its solution can be obtained by some available convex programming technique.

We now present some properties of the above Algorithm 5.3.8. Assume that for each k the set Ω_k has a nonempty interior. At every iteration k, we have $\Omega_k \subset \Omega$, since $H(x, x^k) \geq 0$ for all $x \in \Omega_k$ implies

$$G(x) \leq G(x) + H(x, x^k) \leq 0.$$

Thus $x \in \Omega_k$ implies $x \in \Omega$. In particular, this holds for the solution x^{k+1} of the convex subproblem. Moreover, the solution x^{k+1} is a feasible starting point for the next iteration, since $G(x^{k+1}) \leq 0$ and consequently

$$x^{k+1} \in \Omega_{k+1} := \left\{x \in \mathcal{V} : G(x) + H(x, x^{k+1}) \leq 0\right\}.$$

[4] *A precise definition of convex matrix function is found in Section 3.2.4 from Chapter 3, where theoretical and numerical tools for checking convexity of matrix function are also provided.*

This proves that when $x^0 \in \Omega$, Algorithm 5.3.8 generates a sequence of feasible solutions. Furthermore, as x^k and x^{k+1} belong to Ω_k, we have from (5.12) that $f(x^{k+1}) \leq f(x^k)$. Thus, the sequence $\{f(x^k)\}$ is monotonically decreasing. Since the set Ω is bounded, the linear function $f(x)$ is bounded from below, and consequently the sequence $\{f(x^k)\}$ converges (to a point which we denote by f^*).

Assuming at each iteration k, that the suboptimal value x^{k+1} from the convex subproblem (5.12) is a regular point, then from the Kuhn-Tucker condition (5.10), there exists a positive semidefinite symmetric matrix $\psi^{k+1} \geq 0$ such that

$$\begin{aligned} f'(x^{k+1}) + \langle G'(x^{k+1}) + H'(x^{k+1}, x^k), \psi^{k+1} \rangle = 0 \\ \langle G(x^{k+1}) + H(x^{k+1}, x^k), \psi^{k+1} \rangle = 0 \end{aligned} \tag{5.13}$$

Since the range of $\{x^k\}$ lies in the compact set Ω, then some subsequence $\{x^{k_j}\}$ converges to a point in Ω, analogously, $\|x^{k_j+1} - x^{k_j}\| \to 0$ as $k \to \infty$. Assume this subsequence is such that $f(x^{k_j}) \to f^*$. Then, from the definition of the potential matrix function given in (5.3.6), we have

$$\|H(x^{k_j+1}, x^{k_j})\| \to 0 \quad \text{and} \quad \|H'(x^{k_j+1}, x^{k_j})\| \to 0.$$

Consequently, at the limit, (5.13) equals (5.10), which is the generalized Kuhn-Tucker condition for the original nonconvex problem. If a solution x^* satisfies the conditions in Theorem 5.3.5, then x^* is a minimum of the original nonconvex problem.

Remark 5.3.9 *In practice, the algorithm needs to stop in a finite number of iterations. This can be ensured by enforcing the stopping criteria*

$$\|f(x^{k+1}) - f(x^k)\| < \epsilon. \tag{5.14}$$

Using this criteria, one can no longer guarantee the existence of a convergent subsequence. However, this does not exclude the possibility of a solution be attained in a finite number of iterations, i.e., at some iteration k, the solution x^{k+1} of the convex subproblem may satisfies

$$\begin{aligned} \|f'(x^{k+1}) + \langle G'(x^{k+1}), \psi \rangle\| < \delta \\ \|\langle G(x^{k+1}), \psi \rangle\| < \delta \end{aligned} \tag{5.15}$$

for some $\psi \geq 0$. Since, there exists a x^{k+1} for k large enough such that (5.15) *holds, a possible stopping criteria is to impose* (5.14) *and* (5.15) *with $\epsilon << \delta$.*

Remark 5.3.10 *The presentation in this section has considered uniquely functions of a single variable x. However, the extension to the multivariate case where the functions are defined on a tuple $x = \{x_1, \ldots, x_r\}$ is immediate.*

5.4 Applying the Convexifying Theory

In the integrated design problem stated herein, we do not impose constraints on the control gain matrix, although the control law could be subject to arbitrary affine structural constraints, enabling one to solve complex joint structure/control design problems. However, it is possible to show that the free structural parameters create the equivalence of a decentralized control problem where the "control" gain matrix is diagonal. In order to elaborate more on this point define

$$\Delta = \begin{bmatrix} \alpha_1 I & & & \\ & \ddots & & \\ & & \alpha_i I & \\ & & & \ddots \end{bmatrix}, \qquad \mathbb{I} = \begin{bmatrix} I & \cdots & I & \cdots \end{bmatrix},$$

$$\mathbb{A} = \begin{bmatrix} A_1 \\ \vdots \\ A_i \\ \vdots \end{bmatrix}, \qquad \mathbb{B}_w = \begin{bmatrix} B_{w1} \\ \vdots \\ B_{wi} \\ \vdots \end{bmatrix}, \qquad \mathbb{E} = \begin{bmatrix} E_1 \\ \vdots \\ E_i \\ \vdots \end{bmatrix}.$$

Then the matrices $A(\alpha)$, $B_w(\alpha)$, and $E(\alpha)$ can be written as

$$\begin{aligned} A(\alpha) &= A_0 + \mathbb{I}\Delta\mathbb{A} \\ B_w(\alpha) &= B_{w0} + \mathbb{I}\Delta\mathbb{B}_w \\ E(\alpha) &= E_0 + \mathbb{I}\Delta\mathbb{E} \end{aligned}$$

and the system equation (5.2) as

$$(E_0 + \mathbb{I}\Delta\mathbb{E})\dot{x} = (A_0 + \mathbb{I}\Delta\mathbb{A})x + B_u K x + (B_{w0} + \mathbb{I}\Delta\mathbb{B}_w)w$$

which after some manipulation gives:

$$E_0\dot{x} = (A_0 + B_u K)x + \mathbb{I}\tilde{u} + B_{w0}w$$

with

$$\begin{aligned} u_1 &= \Delta y_1 & \qquad y_1 &= \mathbb{E}\dot{x} \\ u_2 &= \Delta y_2 & \qquad y_2 &= \mathbb{A}x \\ u_3 &= \Delta y_3 & \qquad y_3 &= \mathbb{B}_w w \end{aligned}$$

and $\tilde{u}$ given by $\tilde{u} = u_2 + u_3 - u_1$

If y_i were a set of available measurements, and Δ were a diagonal control gain multiplying the measurements y_i, then the structure design problem has the same mathematical structure as a problem where the "control" signal u_i depends only on the ith "measurement" signal y_i. In control jargon, this is called "decentralized" control. In general the imposition of structure in the control gain matrix is called a "decentralized control problem," and the mathematics needed to solve this problem are the same (find diagonal Δ) as the structural design problem posed herein.

In order to apply the convexifying idea to the integrated structure and control design, we shall define the nonconvex function $G(x)$ we are interested in. From the set of conditions (ii) given in Theorem 5.2.2 we have that the constraint with nonlinear terms is (5.5). Completing the squares, this inequality can be manipulated into

$$\begin{bmatrix} A_{cl}(\alpha,K)PE(\alpha) + E(\alpha)PA_{cl}(\alpha,K)^T & B_w(\alpha) \\ B_w(\alpha)^T & -W^{-1} \end{bmatrix} = \begin{bmatrix} (*) & B_w(\alpha) \\ B_w(\alpha)^T & -W^{-1} \end{bmatrix} < 0.$$

with the term $(*)$ given by

$$(*) = A_{cl}(\alpha,K)PA_{cl}(\alpha,K)^T + E(\alpha)PE(\alpha)^T - \left(A_{cl}(\alpha,K) - E(\alpha)\right) P \left(A_{cl}(\alpha,K) - E(\alpha)\right)^T$$

and $A_{cl}(\alpha,K) = A(\alpha) + B_u K$. Using Schur complements[5], the above inequality can be equivalently written as

$$G(x) := \begin{bmatrix} -\left(A_{cl}(\alpha,K) - E(\alpha)\right) P \left(A_{cl}(\alpha,K) - E(\alpha)\right)^T & B_w(\alpha) & A_{cl}(\alpha,K) & E(\alpha) \\ B_w(\alpha)^T & -W^{-1} & 0 & 0 \\ A_{cl}(\alpha,K)^T & 0 & -Q & 0 \\ E(\alpha)^T & 0 & 0 & -Q \end{bmatrix} < 0. \tag{5.16}$$

where $Q = P^{-1}$ and $x := (P, \alpha, K)$.

We can now define a convexifying function $H(x,\eta)$ that makes $G(x) + H(x,\eta)$ matrix convex in x. For this purpose, let $\eta := (\bar{P}, \bar{\alpha}, \bar{K})$, the matrix $\mathsf{Z}(\eta)$ be

$$\mathsf{Z}(\eta) := \left(A_{cl}(\bar{\alpha},\bar{K}) - E(\alpha)\right) \bar{P},$$

and the function $H(x,\eta)$ be given by

$$\begin{aligned} H(x,\eta) &= (\mathsf{Z}(x) - \mathsf{Z}(\eta))(\mathsf{Z}(x) - \mathsf{Z}(\eta))^T \\ &= (A_{cl}(\alpha,K) - E(\alpha) - \mathsf{Z}(\eta)P^{-1})P(A_{cl}(\alpha,K) - E(\alpha) - \mathsf{Z}(\eta)P^{-1})^T. \end{aligned} \tag{5.17}$$

[5]The matrix $\Pi = \begin{bmatrix} \Phi & \Gamma \\ \Gamma^T & \Delta \end{bmatrix}$ is negative definite ($\Pi < 0$) if and only if $\Delta < 0$ and $\Phi - \Gamma\Delta^{-1}\Gamma^T < 0$. The matrix $\Phi - \Gamma\Delta^{-1}\Gamma^T$ is called a Schur complement of matrix Π.

The above convexifying function $H(x,\eta)$ satisfies all the assumptions given in Definition 5.3.6. It satisfies the first assumption *i)*, since $H(x,\eta)$ is positive semidefinite for all x,η. It is also immediate to see that $H(x,\eta) \to 0$ whenever $x \to \eta$, satisfying in this way assumption *ii)*. Noting that $H(x,\eta)$ is affine in x, its first directional derivative $DH(x,\eta)[\delta_x]$ in the direction δ_x has the form $(\mathsf{Z}(x)-\mathsf{Z}(\eta))\mathsf{Z}(\delta_x)^T + \mathsf{Z}(\delta_x)(\mathsf{Z}(x)-\mathsf{Z}(\eta))^T$. Consequently, whenever $x \to \eta$, one obtain that $DH(x,\eta)[\delta_x] \to 0$ for all δ_x (similarly, $H'(x,\eta) \to 0$). Thus $H(x,\eta)$ also satisfies condition *iii)*.

Adding the convexifying function $H(x,\eta)$ just defined in (5.17) to the first block of the nonconvex matrix function $G(x)$ given in (5.16), we obtain the following matrix inequality

$$\begin{bmatrix} (*) & B_w(\alpha) & A(\alpha)+B_uK & E(\alpha) \\ B_w(\alpha)^T & -W^{-1} & 0 & 0 \\ A(\alpha)^T + K^TB_u^T & 0 & -Q & 0 \\ E(\alpha)^T & 0 & 0 & -Q \end{bmatrix} < 0.$$

with the term $(*)$ given by

$$-(A(\alpha)+B_uK-E(\alpha))\mathsf{Z}(\eta)^T - \mathsf{Z}(\eta)(A(\alpha)+B_uK-E(\alpha))^T + \mathsf{Z}(\eta)Q\mathsf{Z}(\eta)^T$$

Which is the inequality (5.7) given in Theorem 5.2.2. In this form, the Lyapunov matrix $P = Q^{-1}$ and the system matrices $A(\alpha)$ and $E(\alpha)$ no longer appear as products. Instead, these products have been replaced by products with $\mathsf{Z}(\eta)$, which has been introduced with the convexifying function. Notice that η is kept constant and equal to $\eta = x^k$ in the convex subproblems to be solved of the form (5.12).

Considering as the objective function to be minimized an upper bound on the covariance of the control energy, that is, $f = \gamma > \mathcal{E}[u(t)^T u(t)]$, the ideas explained so far are summarized in the algorithm below. The feasible set $\mathcal{G}$ for this problem is given by

$$\mathcal{G} := \{(\gamma, \alpha, K, Q, U) : \text{satisfying the set of inequalities (5.7)-(5.8)}\}.$$

Convexifying Algorithm for Structural Control – CASC

Let $f = \min \gamma$.
Set the nominal values for α_0, A_0, and E_0.
Compute K_0 and P_0 by finding a feasible solution to
the convex conditions given in item (ii) of Theorem 5.2.2.
Set ϵ to some prescribed tolerance and $k = 0$.

Repeat

Set $\mathsf{Z}^k \leftarrow (A_{cl}^k(\alpha^k) - E(\alpha^k))P^k$.

For fixed $\mathsf{Z} = \mathsf{Z}^k$, solve $f^k = \min \gamma$

subject to $(\gamma, \alpha, K, Q, U) \in \text{closure}(\mathcal{G})$.

Denote the solution $(\alpha^*, K^*, Q^*, U^*)$.

Set $\left(\alpha^{k+1}, K^{k+1}, P^{k+1}\right) \leftarrow \left(\alpha^*, K^*, Q^{*-1}\right)$.

$k \leftarrow k + 1$.

Until $||f^k - f^{k-1}|| < \epsilon$

It is possible to add an extra step to the above algorithm: before setting $\mathsf{Z}^k \leftarrow (A_{cl}^k(\alpha^k) - E(\alpha^k))P^k$, we update P^k by solving the LMI (5.7-5.8) with $\alpha = \alpha^k$. In our experiments, this extra step sometimes provides a faster convergence of the CASC algorithm.

5.4.1 Proof of Theorem 5.2.2

This section provides the technical details needed for the proof of Theorem 5.2.2.

Proof. The discussion in the previous section can be used to show the equivalence between conditions (ii) and (iii) given in Theorem 5.2.2. If the constraints in (ii) have a feasible solution $\overline{x} := (\overline{P}, \overline{\alpha}, \overline{F}\,\overline{P}^{-1})$, then $G(\overline{x}) < 0$. Hence the constraints in (iii) also have a feasible solution for some $\mathsf{Z}(\bar{x})$, since $H(\overline{x}, \overline{x}) = 0$ from the definition of the convexifying function. Conversely, if the constraints (iii) have a feasible solution $\overline{x}$, $\overline{\eta}$, then $G(\overline{x}) + H(\overline{x}, \overline{\eta}) < 0$. Since $H(\overline{x}, \overline{\eta}) \geq 0$ by assumption, we have that $G(\overline{x}) \leq G(\overline{x}) + H(\overline{x}, \overline{\eta}) < 0$, and consequently the constraints in (ii) are also feasible.

The equivalence between (i) and (ii) in Theorem 5.2.2 is provided by the following argument. Since, by assumption, the mass matrix $M(\alpha)$ is positive definite for all α of interest, the matrix $E(\alpha)$ is invertible. Hence, for the state feedback law given by $u(t) = Kx$, the closed loop descriptor system

$$\begin{aligned} E(\alpha)\ \dot{x} &= (A(\alpha) + B_u K)x + B_w(\alpha)w, \\ E(\alpha)\ \dot{x} &= A_{cl}(\alpha)x + B_w(\alpha)w. \end{aligned}$$

can be equivalently written in standard state space form as

$$\begin{aligned} \dot{x} &= E(\alpha)^{-1}(A_{cl}(\alpha)x + B_w(\alpha)w), \\ \dot{x} &= \overline{A}_{cl}(\alpha)x + \overline{B}_w(\alpha)w. \end{aligned}$$

For the above stable state space system it is a standard result (Skelton et al. (1998)) that the conditions in item (i) of Theorem 5.2.2 hold if, and only if

$$\mathcal{E}[u(t)^T u(t)] = \mathrm{Tr}\left\{KPK^T\right\} < \gamma \qquad \text{and} \qquad \mathcal{E}[z(t)z(t)^T] = C_z P C_z^T < \Omega,$$

where the symmetric positive definite matrix P is a feasible solution to the Lyapunov inequality

$$\overline{A}_{cl}(\alpha)P + P\overline{A}_{cl}(\alpha)^T + \overline{B}_w(\alpha)W\overline{B}_w(\alpha)^T < 0.$$

This matrix P is an upper bound to the closed loop controllability Grammian. Applying a congruence transformation (which preserves the inertia of the inequality) by multiplying on the left and on the right side by the symmetric matrix $E(\alpha)$, we obtain the equivalent inequality

$$A_{cl}(\alpha)PE(\alpha) + E(\alpha)PA_{cl}(\alpha)^T + B_w(\alpha)WB_w(\alpha)^T < 0.$$

Using a Schur complement, it is possible to show that the above inequality is equivalent to

$$\begin{bmatrix} A_{cl}(\alpha)PE(\alpha) + E(\alpha)PA_{cl}(\alpha)^T & B_w(\alpha) \\ B_w(\alpha)^T & -W^{-1} \end{bmatrix} < 0. \tag{5.18}$$

Noting that $A_{cl}(\alpha) = A(\alpha) + B_u K$ and $F = KP$, inequality (5.18) becomes inequality (5.5) given in (ii). To show (5.6) we introduce the auxiliary symmetric variable U such that

$$U > KPK^T = FP^{-1}F^T$$

then $\gamma > \mathrm{Tr}\{U\} > \mathrm{Tr}\left\{KPK^T\right\}$. Hence, using a Schur complement, this inequality is equivalent to

$$\mathrm{Tr}\{U\} < \gamma, \qquad \begin{bmatrix} U & F \\ F^T & P \end{bmatrix} > 0.$$

This completes the proof. ∎

5.5 Static Output Feedback

This section extends the result provided by Theorem 5.2.2 to the static output feedback case without noise measurements. Let us define the available noise free measurements $y(t)$ for feedback by $y(t) = C_y x(t)$. Thus, the control law is now given by $u = KC_y x$.

Theorem 5.5.1 *Assume that the disturbance $w(t)$ is a stochastic white noise process with intensity $W = W^T > 0$. Let Ω be a given positive definite matrix, and consider the descriptor system given in (5.2). Then the following statements are equivalent:*

(i) There exists structure parameter α, and a stabilizing static output feedback gain $u(t) = KC_y x(t)$ such that

$$\lim_{t\to\infty} \mathcal{E}[z(t)z(t)^T] < \Omega$$

and

$$\lim_{t\to\infty} \mathcal{E}[u(t)^T u(t)] < \gamma.$$

(ii) For some constant matrix Z, there exists matrices of compatible dimensions $Q = Q^T > 0$, $U = U^T > 0$, and K, and parameter α, such that the following LMI are satisfied

$$\begin{bmatrix} (*) & B_w(\alpha) & (A(\alpha) + B_u K C_y) & E(\alpha) \\ B_w(\alpha)^T & -W^{-1} & 0 & 0 \\ (A(\alpha) + B_u K C_y)^T & 0 & -Q & 0 \\ E(\alpha)^T & 0 & 0 & -Q \end{bmatrix} < 0,$$

$$\begin{bmatrix} \Omega & C_z \\ C_z^T & Q \end{bmatrix} > 0, \qquad \mathrm{Tr}\,\{U\} < \gamma, \qquad \begin{bmatrix} U & KC_y \\ C_y^T K^T & Q \end{bmatrix} > 0.$$

where $(*)$ refers to the term

$$(*) = -(A(\alpha) + B_u K C_y - E(\alpha))\mathsf{Z}^T - \mathsf{Z}(A(\alpha) + B_u K C_y - E(\alpha))^T + \mathsf{Z}Q\mathsf{Z}^T$$

Proof. The proof is quite straight, and follow from the proof of Theorem 5.2.2 by replacing the state feedback gain K by the static output feedback gain KC_y. ∎

5.6 Full-order Dynamic Output Feedback

This section now extends the result to the full-order dynamic feedback case. Assume that the disturbances $w(t)$ and $v(t)$ are uncorrelated stochastic white noise process with intensity $W = W^T > 0$ and $V = V^T > 0$ respectively. Let the controlled output for performance evaluation be $z(t) = C_z x$, and the measurements available for feedback be given by $y(t) = C_y x + v$. Then, the system takes the form:

$$\begin{aligned} E(\alpha)\ \dot{x} &= A(\alpha)x + B_u u + B_w(\alpha)w \\ z &= C_z x \\ y &= C_y x + \begin{pmatrix} 0 & I \end{pmatrix} \begin{pmatrix} w \\ v \end{pmatrix}. \end{aligned} \tag{5.19}$$

Where the controller, instead of a constant gain, is now a strictly proper ($D_c = 0$) dynamic feedback law given by

$$\begin{aligned} u &= C_c x_c \\ \dot{x}_c &= A_c x_c + B_c y. \end{aligned} \tag{5.20}$$

Theorem 5.6.1 below characterize the integrate structure and control design problem for the dynamic output feedback case.

Theorem 5.6.1 *Assume that the disturbances $w(t)$ and $v(t)$ are uncorrelated stochastic white noise process with intensity $W = W^T > 0$ and $V = V^T > 0$ respectively. Define $\mathcal{V} = C_y^T V^{-1} C_y$. Let the controller be given by* (5.20)*. Let Ω be a given positive definite matrix, and consider the descriptor system given in* (5.19)*. Then the following statements are equivalent:*

(i) There exists structure parameter α, and a dynamic output feedback controller such that

$$\lim_{t\to\infty} \mathcal{E}[z(t)z(t)^T] < \Omega$$

and

$$\lim_{t\to\infty} \mathcal{E}[u(t)^T u(t)] < \gamma.$$

(ii) There exists matrices of compatible dimensions $P = P^T > 0$, $X = X^T > 0$, and F, and parameter α, such that the following inequalities are satisfied

$$\begin{bmatrix} A(\alpha)PE(\alpha) + E(\alpha)PA(\alpha)^T + B_u F E(\alpha) + E(\alpha)F^T B_u^T & B_w(\alpha) \\ B_w(\alpha)^T & -W^{-1} \end{bmatrix} < 0 \tag{5.21}$$

$$\begin{bmatrix} XE(\alpha)^{-1}A(\alpha) + A(\alpha)^T E(\alpha)^{-1}X - \mathcal{V} & XE(\alpha)^{-1}B_w(\alpha) \\ B_w(\alpha)^T E(\alpha)^{-1}X & -W^{-1} \end{bmatrix} < 0 \tag{5.22}$$

$$\Omega > C_z P C_z^T, \qquad \begin{bmatrix} U & F & 0 \\ F^T & P & I \\ 0 & I & X \end{bmatrix} > 0, \qquad \mathrm{Tr}\,\{U\} < \gamma. \tag{5.23}$$

(iii) For some constant matrices Z, Z_1, Z_2, Z_3, and Z_4, there exists matrices of compatible dimensions $Q = Q^T > 0$, $X = X^T > 0$, and K, and parameter α, such that the

following LMI are satisfied:

$$\begin{bmatrix} (*1) & B_w(\alpha) & A(\alpha)+B_uK & E(\alpha) \\ B_w(\alpha)^T & -W^{-1} & 0 & 0 \\ A(\alpha)^T+K^TB_u^T & 0 & -Q & 0 \\ E(\alpha)^T & 0 & 0 & -Q \end{bmatrix} < 0 \tag{5.24}$$

$$\begin{bmatrix} (*2) & B_w(\alpha) & A(\alpha) & E(\alpha) & \mathsf{Z}_2\mathcal{V} \\ B_w(\alpha)^T & -W^{-1} & 0 & 0 & 0 \\ A(\alpha)^T & 0 & -X & 0 & 0 \\ E(\alpha)^T & 0 & 0 & -0.5X & 0 \\ \mathcal{V}\mathsf{Z}_2^T & 0 & 0 & 0 & -X \end{bmatrix} < 0 \tag{5.25}$$

$$\begin{bmatrix} \Omega & C_z \\ C_z^T & Q \end{bmatrix} > 0, \quad \begin{bmatrix} U & K \\ K^T & Q+\mathsf{Z}_4X\mathsf{Z}_4^T-Q\mathsf{Z}_4^T-\mathsf{Z}_4Q \end{bmatrix} > 0, \qquad \mathrm{Tr}\,\{U\}<\gamma, \tag{5.26}$$

where $(*1)$ *refers to the term*

$$(*1) = -(A(\alpha)+B_uK-E(\alpha))\mathsf{Z}^T - \mathsf{Z}(A(\alpha)+B_uK-E(\alpha))^T + \mathsf{Z}Q\mathsf{Z}^T$$

and $(*2)$ *refers to the term*

$$\begin{aligned}(*2) = \mathsf{Z}_1X\mathsf{Z}_1^T - \mathsf{Z}_1(A(\alpha)-E(\alpha))^T - (A(\alpha)-E(\alpha))\mathsf{Z}_1^T + \mathsf{Z}_2\mathcal{V}\mathsf{Z}_2^T \\ + \mathsf{Z}_3X\mathsf{Z}_3^T - \mathsf{Z}_3(\mathsf{Z}_2\mathcal{V}-E(\alpha))^T - (\mathsf{Z}_2\mathcal{V}-E(\alpha))\mathsf{Z}_3^T.\end{aligned}$$

In this case, one such dynamic controller is given by

$$\begin{aligned} A_c &= E(\alpha)^{-1}B_uC_c + \left[X^{-1}E(\alpha)^{-1}A(\alpha) + (E(\alpha)^{-1}A(\alpha)-B_cC_y)P \right. \\ &\qquad \left. +E(\alpha)^{-1}B_w(\alpha)WB_w(\alpha)^TE(\alpha)^{-1}\right](P-X^{-1})^{-1} \\ B_c &= X^{-1}C_y^TV^{-1} \\ C_c &= F(P-X^{-1})^{-1} \\ D_c &= 0 \end{aligned} \tag{5.27}$$

with $P=Q^{-1}$ *and* $F=KP$.

5.6.1 Proof of Theorem 5.6.1

The equivalence between condition (i) and (ii) in Theorem 5.6.1, along the formula for the controller given by (5.27), is a standard result which is provided in (Skelton et al. (1998)). So, we should only demonstrate the equivalence between condition (ii) and (iii).

Let us start by showing the equivalence between (5.21) and (5.24). By defining $K = FP^{-1}$, the matrix inequality (5.21) becomes

$$\begin{bmatrix} (A(\alpha) + B_u K)PE(\alpha) + E(\alpha)P(A(\alpha)^T + K^T B_u^T) & B_w(\alpha) \\ B_w(\alpha)^T & -W^{-1} \end{bmatrix} < 0.$$

This expression is exactly the same as the MI given in (5.5) for the full state feedback case. Thus, by applying the same convexifying function $H(x, \eta)$ given in (5.17), we obtain LMI (5.24).

Let us proceed by showing that MI (5.22) is equivalent to LMI (5.25). Applying Schur complements, the MI (5.22) can be equivalently written as:

$$XE(\alpha)^{-1}A(\alpha) + A(\alpha)^T E(\alpha)^{-1}X - \mathcal{V} + XE(\alpha)^{-1}B_w(\alpha)WB_w(\alpha)^T E(\alpha)^{-1}X < 0.$$

Multiplying both sides of the above equation by $Y = X^{-1}$, we obtain

$$E(\alpha)^{-1}A(\alpha)Y + YA(\alpha)^T E(\alpha)^{-1} - Y\mathcal{V}Y + E(\alpha)^{-1}B_w(\alpha)WB_w(\alpha)^T E(\alpha)^{-1} < 0.$$

Multiplying by $E(\alpha)$ gives

$$A(\alpha)YE(\alpha) + E(\alpha)YA(\alpha)^T - E(\alpha)Y\mathcal{V}YE(\alpha) + B_w(\alpha)WB_w(\alpha)^T < 0.$$

And finally, by Schur complement, we have

$$\begin{bmatrix} A(\alpha)YE(\alpha) + E(\alpha)YA(\alpha)^T - E(\alpha)Y\mathcal{V}YE(\alpha) & B_w(\alpha) \\ B_w(\alpha)^T & -W^{-1} \end{bmatrix} < 0. \tag{5.28}$$

However, the first entry in this matrix inequality, the expression

$$A(\alpha)YE(\alpha) + E(\alpha)YA(\alpha)^T - E(\alpha)Y\mathcal{V}YE(\alpha)$$

is not convex and thus need to be convexified. To simplify the derivations, we split this expression in two terms

$$\Theta := A(\alpha)YE(\alpha) + E(\alpha)YA(\alpha)^T \qquad \text{and} \qquad \Gamma := -E(\alpha)Y\mathcal{V}YE(\alpha),$$

so that we can convexify each one of these terms independently.

In order to determine a suitable potential function, we need to express the term Θ in a more convenient way by completing its square. After completing its square, the term Θ becomes

$$\Theta = -(A(\alpha) - E(\alpha))Y(A(\alpha) - E(\alpha))^T + A(\alpha)YA(\alpha)^T + E(\alpha)YE(\alpha).$$

Substituting this formula back into the matrix inequality (5.28), and applying Schur complements, we obtain

$$\begin{bmatrix} (*) & B_w(\alpha) & A(\alpha) & E(\alpha) \\ B_w(\alpha) & -W^{-1} & 0 & 0 \\ A(\alpha)^T & 0 & -X & 0 \\ E(\alpha)^T & 0 & 0 & -X \end{bmatrix} < 0 \tag{5.29}$$

where the term $(*)$ is given by

$$(*) = -(A(\alpha) - E(\alpha))Y(A(\alpha) - E(\alpha))^T - E(\alpha)Y\mathcal{V}YE(\alpha),$$

and $X = Y^{-1}$. This MI is evidently equivalent to the MI given in (5.28), and thus the problem now resumes to convexify the following expression

$$-(A(\alpha) - E(\alpha))Y(A(\alpha) - E(\alpha))^T - E(\alpha)Y\mathcal{V}YE(\alpha).$$

To simplify subsequents derivations, we split again this expression as

$$\overline{\Theta} := -(A(\alpha) - E(\alpha))Y(A(\alpha) - E(\alpha))^T \qquad \text{and} \qquad \Gamma := -E(\alpha)Y\mathcal{V}YE(\alpha).$$

5.6.2 Convexifying the term $\overline{\Theta}$

We need to provide a potential function for the nonconvex term

$$\overline{\Theta} = -(A(\alpha) - E(\alpha))Y(A(\alpha) - E(\alpha))^T.$$

In the derivations to be presented, we have suppressed the dependence of $\mathsf{Z}(\eta)$ and $H(x, \eta)$ on x and η. Let us define $\mathsf{Z}_1 = (A(\alpha) - E(\alpha))Y$, and take the potential function H_1 to be

$$H_1 = (A(\alpha) - E(\alpha) - \mathsf{Z}_1 Y^{-1})Y(A(\alpha) - E(\alpha) - \mathsf{Z}_1 Y^{-1})^T.$$

Or equivalently

$$\begin{aligned} H_1 = (A(\alpha) - E(\alpha))Y(A(\alpha) - E(\alpha))^T + \mathsf{Z}_1 Y^{-1}\mathsf{Z}_1^T \\ - \mathsf{Z}_1(A(\alpha) - E(\alpha))^T - (A(\alpha) - E(\alpha))\mathsf{Z}_1^T. \end{aligned}$$

Adding the term H_1 to the MI given in (5.29), we obtain

$$\begin{bmatrix} (*) & B_w(\alpha) & A(\alpha) & E(\alpha) \\ B_w(\alpha)^T & -W^{-1} & 0 & 0 \\ A(\alpha)^T & 0 & -X & 0 \\ E(\alpha)^T & 0 & 0 & -X \end{bmatrix} < 0, \tag{5.30}$$

where the term $(*)$ is given by

$$(*) = -E(\alpha)Y\mathcal{V}YE(\alpha) + \mathsf{Z}_1 X \mathsf{Z}_1^T - \mathsf{Z}_1(A(\alpha) - E(\alpha))^T - (A(\alpha) - E(\alpha))\mathsf{Z}_1^T$$

with $Y = X^{-1}$. The term $\overline{\Theta}$ has now been convexified, however, the above MI (5.30) is still not convex, since it contains the term $\Gamma = -E(\alpha)Y\mathcal{V}YE(\alpha)$, which was not accounted in the above manipulations.

5.6.3 Convexifying the term Γ

Now, we show how to convexify the term Γ. For this specific term, we will have to apply two consecutive potential functions. Let us define the first potential function H_2 for the term Γ as

$$H_2 = (E(\alpha) + \mathsf{Z}_2 Y^{-1})Y\mathcal{V}Y(E(\alpha) + \mathsf{Z}_2 Y^{-1})^T$$

Defining $\mathsf{Z}_2 = -E(\alpha)Y$, this expression can be equivalently written as

$$H_2 = E(\alpha)Y\mathcal{V}YE(\alpha) + \mathsf{Z}_2\mathcal{V}\mathsf{Z}_2^T + \mathsf{Z}_2\mathcal{V}YE(\alpha) + E(\alpha)Y\mathcal{V}\mathsf{Z}_2^T$$

Adding this potential function H_2 to the MI (5.30), we obtain

$$\begin{bmatrix} (*) & B_w(\alpha) & A(\alpha) & E(\alpha) \\ B_w(\alpha)^T & -W^{-1} & 0 & 0 \\ A(\alpha)^T & 0 & -X & 0 \\ E(\alpha)^T & 0 & 0 & -X \end{bmatrix} < 0 \tag{5.31}$$

where the term $(*)$ is given by

$$(*) = \mathsf{Z}_1 X \mathsf{Z}_1^T - \mathsf{Z}_1(A(\alpha) - E(\alpha))^T - (A(\alpha) - E(\alpha))\mathsf{Z}_1^T + \mathsf{Z}_2\mathcal{V}\mathsf{Z}_2^T + \mathsf{Z}_2\mathcal{V}YE(\alpha) + E(\alpha)Y\mathcal{V}\mathsf{Z}_2^T$$

Due to the term $\mathsf{Z}_2\mathcal{V}YE(\alpha) + E(\alpha)Y\mathcal{V}\mathsf{Z}_2^T$, this MI (5.31) is still not convex. Thus, we should define a suitable potential function for this term. However, we first manipulate this expression by completing its square:

$$\mathsf{Z}_2\mathcal{V}YE(\alpha) + E(\alpha)Y\mathcal{V}\mathsf{Z}_2^T = - (\mathsf{Z}_2\mathcal{V} - E(\alpha))Y(\mathsf{Z}_2\mathcal{V} - E(\alpha))^T + \mathsf{Z}_2\mathcal{V}Y\mathcal{V}\mathsf{Z}_2^T + E(\alpha)YE(\alpha)^T.$$

Substituting this expression into the MI (5.31), we obtain

$$\begin{bmatrix} (*) & B_w(\alpha) & A(\alpha) & E(\alpha) \\ B_w(\alpha)^T & -W^{-1} & 0 & 0 \\ A(\alpha)^T & 0 & -X & 0 \\ E(\alpha)^T & 0 & 0 & -X \end{bmatrix} < 0$$

with the term $(*)$ given by

$$\begin{aligned}(*) = \mathsf{Z}_1 X \mathsf{Z}_1^T - \mathsf{Z}_1(A(\alpha) - E(\alpha))^T - (A(\alpha) - E(\alpha))\mathsf{Z}_1^T + \mathsf{Z}_2 \mathcal{V} \mathsf{Z}_2^T \\ - (\mathsf{Z}_2\mathcal{V} - E(\alpha))Y(\mathsf{Z}_2\mathcal{V} - E(\alpha))^T + \mathsf{Z}_2\mathcal{V}Y\mathcal{V}\mathsf{Z}_2^T + E(\alpha)YE(\alpha)^T.\end{aligned}$$

Finally, by applying Schur complements, this inequality can be further simplified to

$$\begin{bmatrix} (*) & B_w(\alpha) & A(\alpha) & E(\alpha) & \mathsf{Z}_2\mathcal{V} \\ B_w(\alpha)^T & -W^{-1} & 0 & 0 & 0 \\ A(\alpha)^T & 0 & -X & 0 & 0 \\ E(\alpha)^T & 0 & 0 & -0.5X & 0 \\ \mathcal{V}\mathsf{Z}_2^T & 0 & 0 & 0 & -X \end{bmatrix} < 0 \tag{5.32}$$

with the term $(*)$ given by

$$\begin{aligned}(*) = \mathsf{Z}_1 X \mathsf{Z}_1^T - \mathsf{Z}_1(A(\alpha) - E(\alpha))^T - (A(\alpha) - E(\alpha))\mathsf{Z}_1^T \\ + \mathsf{Z}_2\mathcal{V}\mathsf{Z}_2^T - (\mathsf{Z}_2\mathcal{V} - E(\alpha))Y(\mathsf{Z}_2\mathcal{V} - E(\alpha))^T.\end{aligned}$$

This MI (5.32) is naturally identical to the MI (5.31), since we have only manipulated the nonconvex term by completing its square. In this way, the nonconvex term to be convexified is now given by

$$-(\mathsf{Z}_2\mathcal{V} - E(\alpha))Y(\mathsf{Z}_2\mathcal{V} - E(\alpha))^T.$$

A possible potential function H_3 for this expression is given by

$$H_3 = (\mathsf{Z}_2\mathcal{V} - E(\alpha) - \mathsf{Z}_3 Y^{-1})Y(\mathsf{Z}_2\mathcal{V} - E(\alpha) - \mathsf{Z}_3 Y^{-1})^T.$$

Or equivalently

$$\begin{aligned}H_3 = (\mathsf{Z}_2\mathcal{V} - E(\alpha))Y(\mathsf{Z}_2\mathcal{V} - E(\alpha))^T + \mathsf{Z}_3 Y^{-1}\mathsf{Z}_3^T - \mathsf{Z}_3(\mathsf{Z}_2\mathcal{V} - E(\alpha))^T \\ - (\mathsf{Z}_2\mathcal{V} - E(\alpha))\mathsf{Z}_3^T\end{aligned}$$

with $\mathsf{Z}_3 = (\mathsf{Z}_2\mathcal{V} - E(\alpha))Y$.

To obtain the final expression, we must add this potential H_3 to the MI (5.32). By doing so, we obtain the following LMI:

$$\begin{bmatrix} (*) & B_w(\alpha) & A(\alpha) & E(\alpha) & \mathsf{Z}_2\mathcal{V} \\ B_w(\alpha)^T & -W^{-1} & 0 & 0 & 0 \\ A(\alpha)^T & 0 & -X & 0 & 0 \\ E(\alpha)^T & 0 & 0 & -0.5X & 0 \\ \mathcal{V}\mathsf{Z}_2^T & 0 & 0 & 0 & -X \end{bmatrix} < 0$$

with the term $(*)$ given by

$$(*) = \mathsf{Z}_1 X \mathsf{Z}_1^T - \mathsf{Z}_1(A(\alpha) - E(\alpha))^T - (A(\alpha) - E(\alpha))\mathsf{Z}_1^T + \mathsf{Z}_2\mathcal{V}\mathsf{Z}_2^T + \mathsf{Z}_3 X \mathsf{Z}_3^T \\ - \mathsf{Z}_3(\mathsf{Z}_2\mathcal{V} - E(\alpha))^T - (\mathsf{Z}_2\mathcal{V} - E(\alpha))\mathsf{Z}_3^T.$$

Which is exactly the LMI given in (5.25). The final convexifying function is thus given by $H_2 + H_3$.

The only part missing now, is to show the equivalence between (5.23) and (5.26). Since $Q = P^{-1}$ and $K = FP^{-1} = FQ$, the inequality (5.23) can be manipulate as

$$\begin{bmatrix} U & KP & 0 \\ PK^T & P & 0 \\ 0 & 0 & X \end{bmatrix} > 0 \quad \Leftrightarrow \quad \begin{bmatrix} U & KP \\ PK^T & P - X^{-1} \end{bmatrix} > 0.$$

Which, by applying the following congruence transformation

$$\begin{bmatrix} I & 0 \\ 0 & P^{-1} \end{bmatrix} \begin{bmatrix} U & KP \\ PK^T & P - X^{-1} \end{bmatrix} \begin{bmatrix} I & 0 \\ 0 & P^{-1} \end{bmatrix} > 0$$

gives

$$\begin{bmatrix} U & K \\ K^T & Q - QX^{-1}Q \end{bmatrix} > 0. \tag{5.33}$$

The above expression $QX^{-1}Q$ is not convex, but can be convexified with the following convexifying function:

$$H_4 = (QX^{-1} - \mathsf{Z}_4)X(QX^{-1} - \mathsf{Z}_4)^T = QX^{-1}Q + \mathsf{Z}_4 X \mathsf{Z}_4^T - Q\mathsf{Z}_4^T - \mathsf{Z}_4 Q.$$

Where $\mathsf{Z}_4 = QX^{-1}$. Now, by adding the potential H_4 to the above MI (5.33), we obtain the LMI given in (5.26):

$$\begin{bmatrix} U & K \\ K^T & Q + \mathsf{Z}_4 X \mathsf{Z}_4^T - Q\mathsf{Z}_4^T - \mathsf{Z}_4 Q \end{bmatrix} > 0.$$

This complete the proof of Theorem 5.6.1.

5.7 Numerical Results

5.7.1 The CASC algorithm *versus* the two-step redesign approach

We present an experiment which compares the behavior of the CASC algorithm with the TSRED algorithm (the two-step redesign approach).

Algorithm 5.7.1 (The TSRED algorithm) *Let α, Ω, E, W, and B_u be given.*

1. *For fixed α, evaluate $A(\alpha)$ and $B_w(\alpha)$. Then, design a controller $K = FP^{-1}$ by solving the LMI problem* (5.5)-(5.6) *in Theorem 5.2.2 for F, P, and U.*

2. *For fixed Lyapunov matrix P, redesign the structure parameter α by solving the LMI problem* (5.5)-(5.6) *for α, F, and U.*

3. *Until convergence, go back to 1.*

Note that the mass matrix E in this approach is not allowed to be redesigned. This is not the case for the CASC algorithm, where matrix E can also be affine in α.

The dynamical system for this example is a three-degree-of-freedom mass-spring system described in Figure 5.1. This class of model can represent many engineering systems. In the experiment to be presented, we did not use realistic data, since the main point is a comparative exposition of our method. However, in the next section, we provide a fairly realistic application of the CASC algorithm to the design of a civil engineering structure.

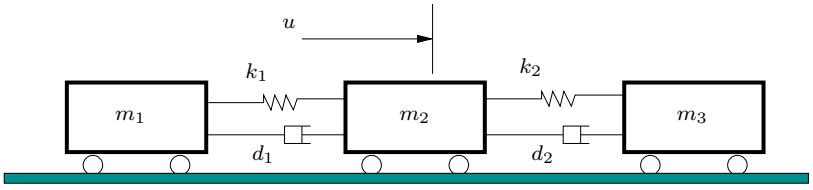

Figure 5.1: A 3-DOF mass-spring system

In this model, m_1, m_2, and m_3 are the mass of each one of the car. Their nominal values are taken to be $m_1 = 4$, $m_2 = 2$, and $m_3 = 10$, The nominal value of the stiffness of the spring elements connecting those cars are $k_1 = 1$ and $k_2 = 1$. The nominal value for the damping coefficient are $d_1 = 0.01$ and $d_2 = 0.01$. The controller, which we denote by u, is applied to the mass m_2. The states are the displacement and velocity of each mass m_i,

Table 5.1: Bound on the performance

mass	$\mathcal{E}\, x_i^2$	$\mathcal{E}\, \dot{x}_i^2$
m_1	50	250
m_2	150	250
m_3	150	500

relative to a fix referential. Thus x_i represents the displacement of m_i and $\dot{x}_i$ its velocity. The disturbance is a unitary white noise applied to each one of the states.

The problem we are interested is the simultaneous design of the controller u and the parameters k_1 and d_1. We minimize the variance of the controller u imposing that the variance of the displacement and of the velocity of each mass are bounded by some prescribed value. These performance bounds are presented in Table 5.1.

Since we minimize only two parameters k_1 and d_1, the levelcurve for this example can be easily described by a plot. By a brute force procedure, we found the optimal values for the parameters to be $k_1^* = 0.475$ and $d_1^* = 1.010$. For these values, the required control energy is given by $||u||_2^2 = 0.118$. We solve the simultaneous structure and control problem using the TSRED algorithm and the CASC algorithm for four different initial conditions, given by $(d_1, k_1) = \{(1,1), (6,0.5), (10,1.5), (1,2)\}$.

Figure 5.2 presents the results corresponding to TSRED algorithm. After 1000 iterations this approach did not converge to the optimal solution k_1^* and d_1^*, for any of the initial guess.

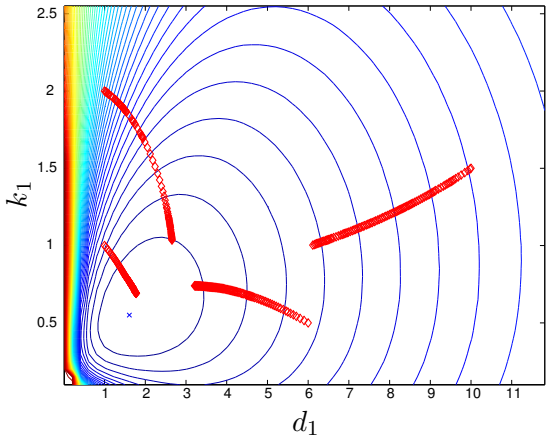

Figure 5.2: Solution path for the TSRED algorithm

On the other hand, the CASC algorithm converged in less than 150 iterations for all the above initial guess. The levelcurve and the solution path are presented in Figure 5.3.

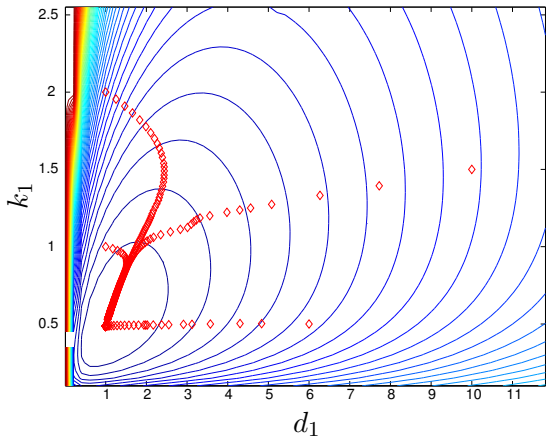

Figure 5.3: Solution path for the CASC algorithm

5.7.2 Isolating a civil engineering structure against earthquakes

To illustrate the proposed methodology for solving the integrated control and structure optimization problem in the LMI framework, we choose the problem of isolating a civil engineering structure against earthquakes. This will not be a comprehensive presentation on how to solve this specific structure problem, but rather on providing efficient tools for this purpose.

The field of controlling vibrations of structures against earthquakes has attracted the interest of many researchers. The references Kose et al. (1998); Ramalho et al. (2000); Spencer Jr. et al. (1998) provide a concise explanation of the structural problem, and a benchmark comparison of various structural control algorithms applied in an evaluation model obtained from experimental data.

The model of the system in consideration is shown in Figure 5.4. This is a three-degree-of-freedom version of the same structure used in Ramalho et al. (2000). The nominal values for this system are given in Table 5.2. The control inputs u_i are independent forces applied to each floor. Hence, for the model (5.2), $\hat{B}_u = I$, the matrix $\hat{B}_w$ is given by $\hat{B}_w = (m_1, m_2, m_3)^T$, and the disturbance vector w is assumed to be a white noise process with intensity $W = 16$ [m^2/s^4], which represents the earthquake acceleration of the ground motion $\ddot{x}_g$ in Figure 5.4.

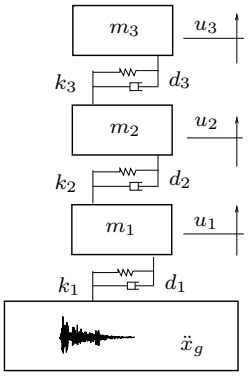

Figure 5.4: 3-DOF model

Table 5.2: Nominal Structural Parameters

Floor Masses	**Stiffness Coefficients**	**Damping Coefficients**
[Kg]	[kN/m]	[kN·s/m]
$m_1 = 5897$	$k_1 = 33732$	$d_1 = 67$
$m_2 = 5897$	$k_2 = 29093$	$d_2 = 58$
$m_3 = 5897$	$k_3 = 28621$	$d_3 = 57$

The dynamics of the above system is described by Eq. (5.2) with the mass matrix M given by $M = \text{diag}(m_1, m_2, m_3)$, and the stiffness matrix S and the damping matrix D given by:

$$S = \begin{bmatrix} k_1 + k_2 & -k_2 & 0 \\ -k_2 & k_2 + k_3 & -k_3 \\ 0 & -k_3 & k_3 \end{bmatrix}, \qquad D = \begin{bmatrix} d_1 + d_2 & -d_2 & 0 \\ -d_2 & d_2 + d_3 & -d_3 \\ 0 & -d_3 & d_3 \end{bmatrix}.$$

The states are the displacement and the velocity of each floor relative to the ground, i.e.: q_i represents the displacement of the mass m_i, and $\dot{q}_i$ its velocity.

We are interested in the simultaneous design of the parameters of the structure and the controller, using the control implementation stated in Theorem 5.2.2. We seek designs which limit the variance of the inter-story drift $z_1 = q_1$, $z_{i+1} = q_{i+1} - q_i$, $i = 1, 2$, and their

velocities $z_{i+3} = \dot{z}_i$. Thus the output C_z is given by

$$C_z = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 \\ -1 & 1 & 0 & 0 & 0 & 0 \\ 0 & -1 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & -1 & 1 & 0 \\ 0 & 0 & 0 & 0 & -1 & 1 \end{bmatrix}.$$

We seek to bound the output variances such that $\mathcal{E}\, z_i^2 \leq 0.0002$ [m] and $\mathcal{E}\, \dot{z}_i^2 \leq 0.3$ [m/s], for $i = 1, 2, 3$. Thus the diagonal of the output covariance matrix $C_z P C_z^T$ of the closed loop system should be bounded by Ω, that is $C_z^i P {C_z^i}^T < \Omega_i$, with the bound Ω given by

$$\Omega = \left(2 \times 10^{-4}, 2 \times 10^{-4}, 2 \times 10^{-4}, 0.3, 0.3, 0.3\right).$$

In our notation, C_z^i means the ith row of the matrix C_z. Note that $C_z^i P {C_z^i}^T < \Omega_i$ is a convex constraint.

For all runs the stopping criteria was the relative error on the control energy between two successive iterations $(\mathcal{E}[(u^{k+1})^T u^{k+1}] - \mathcal{E}[(u^k)^T u^k])/\,\mathcal{E}[(u^k)^T u^k]$ is less than 5×10^{-4} for ten consecutive times.

We assume that the lower and the upper bounds on all the parameters are 0.5 and 2.0 of the nominal values in Table 5.2. For brevity, we will call the standard deviation of the control $\sqrt{\mathcal{E}[u^T u]}$ the "control effort."

Example 5.7.1 $[k_2, d_2]$ In this first example, the parameters to be redesigned are the spring stiffness k_2 and the damping coefficient d_2. We found by an exhaustive search the global optimum for the integrated structure and control design problem. These optimal values are $k_2 = 26699$ [kN/m] and $d_2 = 116$ [kN·s/m], which gives the global minimal control effort $\sqrt{\mathcal{E}[u^T u]} = \sqrt{3276393}$ [kN] required to achieve the design output performance Ω.

Now, we simulate our CASC algorithm. First, an initial controller K_0 using the nominal parameters in Table 5.2 is determined, by solving the LMIs (5.5-5.6) in Theorem 5.2.2 (which for fixed α it is a convex problem). The initial controller evaluated in this way is

$$K_0 = \begin{bmatrix} -176699 & -319360 & -8245 & -11994 & -9657 & -5625 \\ 203413 & -134380 & -189180 & -9614 & -16822 & -16588 \\ 33250 & 47538 & -186040 & -5598 & -16624 & -27045 \end{bmatrix}. \tag{5.34}$$

For this initial controller, the control effort is $\sqrt{\mathcal{E}[u^T u]} = \sqrt{4408517}$ [kN]. Note that if the designer is not allowed to change the parameters of the structure, then the initial controller K_0 provides the best required performance using the least control effort, which is much higher than the globally optimal effort given by $\sqrt{\mathcal{E}[u^T u]} = \sqrt{3276393}$ [kN].

We proceed with the algorithm CASC (integrated design) generating the results in Figure 5.5. After 26 iterations the algorithm converged to the solution $k_2^* = 26751$ [kN/m], $d_2^* = 116$ [kN·s/m], and the control gain

$$K^* = \begin{bmatrix} -160292.2 & -148749.9 & -702.4 & -9389.4 & -7226.0 & -5614.1 \\ 187423.4 & -175483.0 & -149112.4 & -7176.1 & -14506.8 & -15039.6 \\ 93662.9 & -77905.2 & -134451.6 & -5640.8 & -15032.6 & -22546.2 \end{bmatrix},$$

which provides a control effort of $\sqrt{\mathcal{E}[u^T u]} = \sqrt{3276453}$ [kN].

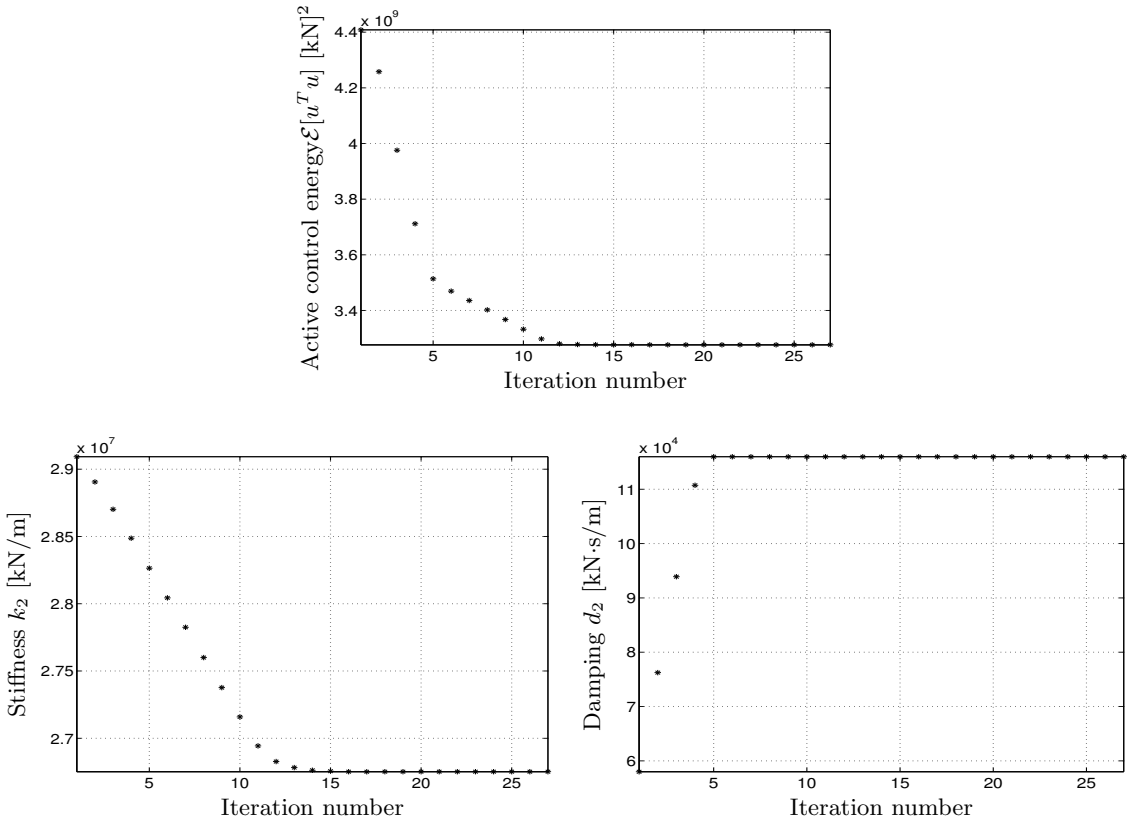

Figure 5.5: Example 1 (k_2, d_2): Integrated design using CASC

For this controller K^*, the achieved output performance, the diagonal of the matrix

$C_z P C_z^T$, is

$$\begin{pmatrix} 0.00019999806470 \\ 0.00019999608202 \\ 0.00006581424510 \\ 0.29217688567246 \\ 0.23757322272961 \\ 0.14359776862926 \end{pmatrix}.$$

This shows that the constraints $\mathcal{E}\, z_1^2$, $\mathcal{E}\, z_2^2$, and $\mathcal{E}\, \dot{z}_1^2$ are active (Table 5.4).

Example 5.7.2 (k_2, d_2, m_2) We start this example with the same initial controller K_0, but

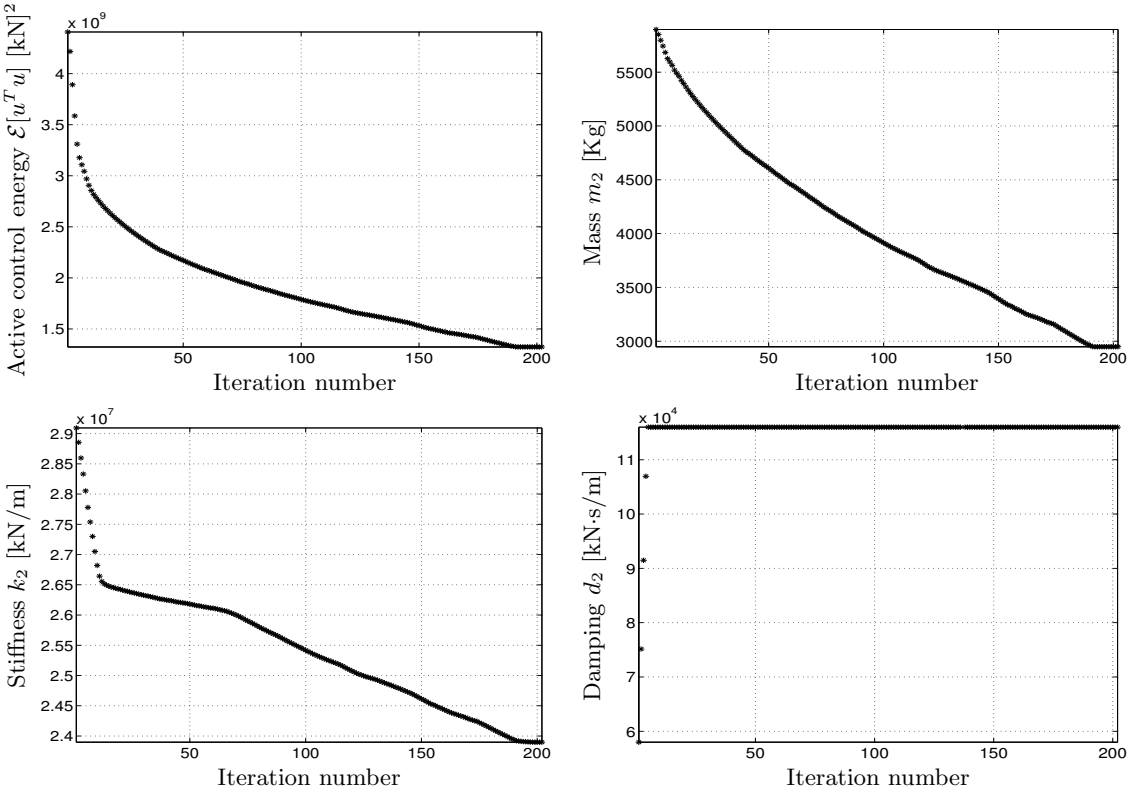

Figure 5.6: Example 2 (k_2, d_2, m_2): Integrated design using CASC

we add the additional parameter m_2 to be optimized. The results from the integrated design is presented in Figure 5.6. After 201 iterations, the algorithm converged to the solution: $m_2^* = 2948$ [Kg], $k_2^* = 23897$ [kN/m], and $d_2^* = 116$ [kN·s/m], with the control law given by

$$K^* = \begin{bmatrix} 190499.5 & -187499.3 & -1254.6 & -25891.8 & -5650.8 & 4386.9 \\ 574319.0 & -135201.6 & -179646.8 & -11181.2 & -6101.4 & -8618.9 \\ 447891.3 & -21021.9 & -163367.5 & 4487.7 & -4548.0 & -20177.4 \end{bmatrix}.$$

This gain provides a control effort of $\sqrt{\mathcal{E}[u^T u]} = \sqrt{1323584}$ [kN].

For this control gain K^* and parameters α^*, the achieved output performance, the diagonal of the matrix $C_z P C_z^T$, is

$$\begin{pmatrix} 0.00017774233782 \\ 0.00019999687426 \\ 0.00008083386592 \\ 0.29999795054617 \\ 0.26623945460839 \\ 0.14227944011144 \end{pmatrix}.$$

For this case the binding constraints are $\mathcal{E}\, z_2^2$, and $\mathcal{E}\, \dot{z}_1^2$.

Example 5.7.3 (k_1, k_2, k_3, d_1, d_2, d_3, m_1, m_2, m_3) In this example, we follow the same

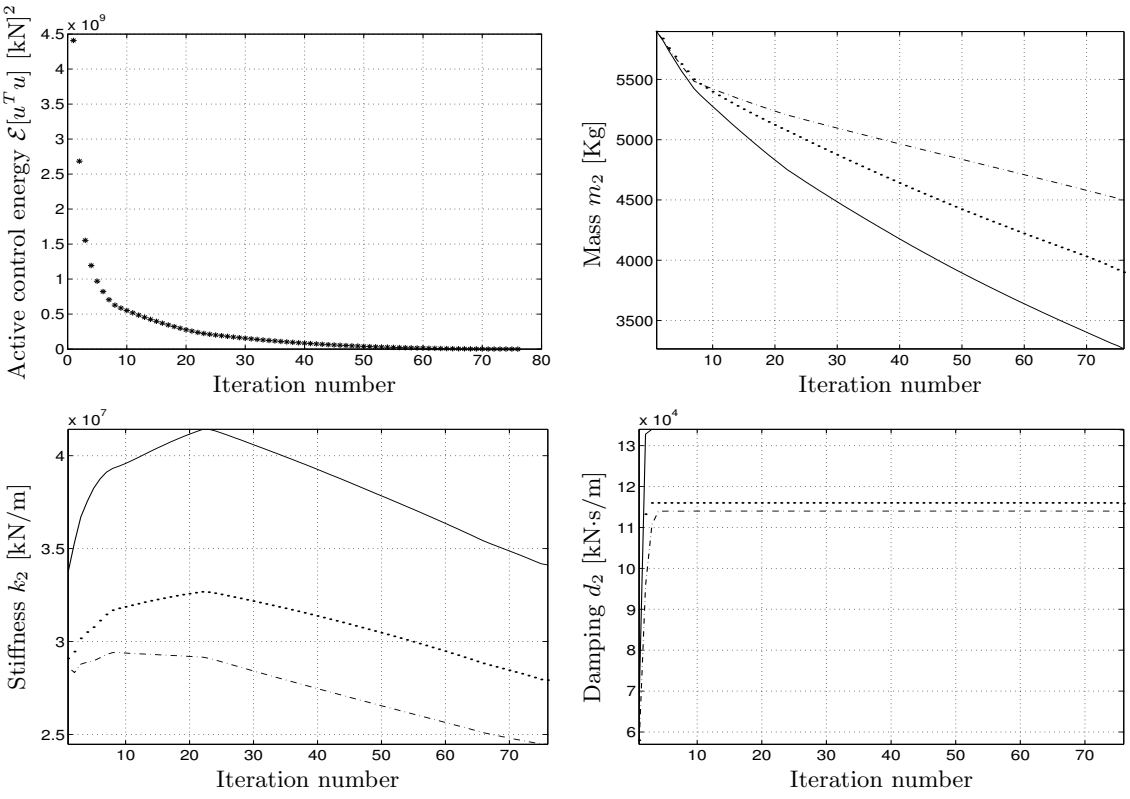

Figure 5.7: Example 3 (k_1, k_2, k_3, d_1, d_2, d_3, m_1, m_2, m_3): Integrated design using CASC

steps as in the previous Example 2, but now all the parameters of the structure are optimized: m_1, m_2, m_3, k_1, k_2, k_3, d_1, d_2, and d_3. The same initial controller K_0 given in

(5.34) is also used. After 75 iterations, the CASC algorithm gives the solution $\mathcal{E}[u^T u] = 0$ (see figure 5.7). Thus no active control is needed to achieve the prescribed performance, and the design is completely passive. The parameters are $m_1^* = 3264$ [Kg], $m_2^* = 3899$ [Kg], $m_3^* = 4498$ [Kg], $k_1^* = 34120$ [kN/m], $k_2^* = 27923$ [kN/m], $k_3^* = 24473$ [kN/m], $d_1^* = 133$ [kN·s/m], $d_2^* = 116$ [kN·s/m], and $d_3^* = 114$ [kN·s/m].

In this case, the achieved output performance, the diagonal of the matrix $C_z P C_z^T$, is

$$\begin{pmatrix} 0.00018566840635 \\ 0.00019924123968 \\ 0.00009712073257 \\ 0.29980689683670 \\ 0.29985557567524 \\ 0.16787420951997 \end{pmatrix}.$$

Table 5.3 summarizes our findings. Using an integrated approach, no control effort is required to achieve $\mathcal{E}\, z_i^2 \leq 0.0002$ [m^2] and $\mathcal{E}\, \dot{z}_i^2 \leq 0.3$ [m^2/s^2], $i = 1, 2, 3$. With feedback control fixed at the nominal parameters (K_0) the control effort needed is $\sqrt{4408517}$ [kN].

Table 5.3: Control energy [kN]^2 for performance guarantee Ω: $\mathcal{E}\, z_i^2 \leq 0.0002$ [m^2] and $\mathcal{E}\, \dot{z}_i \leq 0.3$ [m^2/s^2], $i = 1, 2, 3$

		Control Energy [kN]^2	
	Parameters	**Integrated**	**Active only**
Example 1:	k_2, d_2	$\sqrt{3276453}$	$\sqrt{4408517}$
Example 2:	k_2, d_2, m_2	$\sqrt{1323584}$	$\sqrt{4408517}$
Example 3:	all parameters	0	$\sqrt{4408517}$

For each of the previous three designs, the elements of the diagonal of the output covariance matrix are shown in Table 5.4.

Example 5.7.4 (k_1, k_2, k_3, d_1, d_2, d_3, m_1, m_2, m_3). This example employs all the free parameters, but we change Ω by scaling by a factor μ, that is $\Omega\mu$, in order to find the performance bound $\overline{\Omega}$ that represents the best performance that is achievable with only passive design. The active control energy $\mathcal{E}[u^T u]$ as a function of the scaling factor μ is shown in Figure 5.8. Thus $\mu = 0.64$, i.e., $\overline{\Omega} = 0.64\Omega$ represents the lowest bound on the output covariance for which the design is still completely passive.

Table 5.4: Achieved output performance (diagonal entries of $C_z P C_z^T$)

	$\mathcal{E}\, z_1^2$ [m]	$\mathcal{E}\, z_2^2$ [m]	$\mathcal{E}\, z_3^2$ [m]	$\mathcal{E}\, z_4^2$ [m/s]	$\mathcal{E}\, z_5^2$ [m/s]	$\mathcal{E}\, z_6^2$ [m/s]
Ex. 1	0.00020	0.00020	0.00007	0.29218	0.23757	0.14360
Ex. 2	0.00018	0.00020	0.00008	0.30000	0.26624	0.14228
Ex. 3	0.00019	0.00020	0.00010	0.29981	0.29986	0.16787
Bound	0.00020	0.00020	0.00020	0.30000	0.30000	0.30000

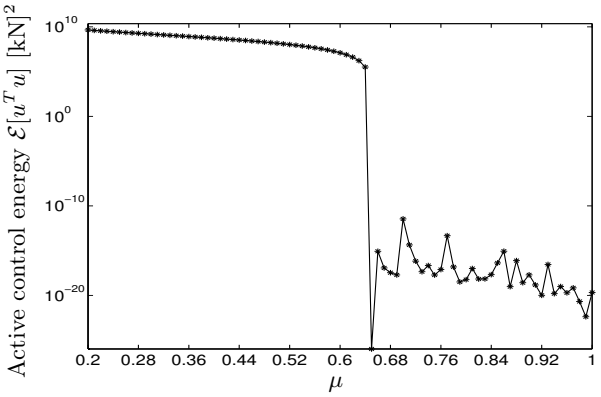

Figure 5.8: Changing the performance bound $\mu\Omega$

For this bound $\overline{\Omega}$, the CASC algorithm converged after 264 iterations to the passive ($K_0 = 0$ and $\mathcal{E}[u^T u] = 0$) solution : $m_1^* = 2948$ [Kg], $m_2^* = 2948$ [Kg], $m_3^* = 2948$ [Kg], $k_1^* = 37044$ [kN/m], $k_2^* = 28155$ [kN/m], $k_3^* = 16814$ [kN/m], $d_1^* = 134$ [kN·s/m], $d_2^* = 116$ [kN·s/m], and $d_3^* = 114$ [kN·s/m]. For this system, the achieved output performance is given in Table 5.5 below.

Table 5.5: Achieved output performance (diagonal entries of $C_z P C_z^T$)

	$\mathcal{E}\, z_1^2$ [m]	$\mathcal{E}\, z_2^2$ [m]	$\mathcal{E}\, z_3^2$ [m]	$\mathcal{E}\, z_4^2$ [m/s]	$\mathcal{E}\, z_5^2$ [m/s]	$\mathcal{E}\, z_6^2$ [m/s]
Ex. 4	0.00008	0.00009	0.00008	0.19200	0.19200	0.19200
Bound	0.00013	0.00013	0.00013	0.19200	0.19200	0.19200

Example 5.7.5 (k_1, k_2, k_3, d_1, d_2, d_3, m_1, m_2, m_3). Now, we imposed a tighter upper bound, and obtain an active control law. We choose μ to be 0.4, thus $\overline{\Omega} = 0.4\Omega$, yielding performance 2.5 times better than the examples which used the performance criterion $C_z P C_z^T < \Omega$, for earthquakes intensity $W = 16$. Note that the performance $C_z P C_z^T$ sim-

ply scales with W. So the design we now discuss can also be interpreted as delivering the same performance bound as Ω in the presence earthquakes intensity $W = 2.5(16) = 40$. For this bound, the CASC algorithm converged after 142 iterations to the active solution : $m_1^* = 2948$ [Kg], $m_2^* = 2948$ [Kg], $m_3^* = 2948$ [Kg], $k_1^* = 67343$ [kN/m], $k_2^* = 46289$ [kN/m], $k_3^* = 28354$ [kN/m], $d_1^* = 134$ [kN·s/m], $d_2^* = 116$ [kN·s/m], $d_3^* = 114$ [kN·s/m], with the control law given by

$$K^* = \begin{bmatrix} 307824.1 & -390158.3 & 80694.4 & -64017.3 & 3997.3 & 3588.3 \\ 827567.6 & -216198.7 & -125413.8 & 3072.2 & -47713.6 & 3965.2 \\ 190682.3 & 165905.7 & -122324.2 & 4226.7 & 3628.9 & -42525.5 \end{bmatrix}.$$

This gain provides a control effort of $\sqrt{\mathcal{E}[u^T u]} = \sqrt{1536541}$ [kN]. The achieved output performance is given in Table 5.6 below.

Table 5.6: Achieved output performance (diagonal entries of $C_z P C_z^T$)

	$\mathcal{E}\, z_1^2$ [m]	$\mathcal{E}\, z_2^2$ [m]	$\mathcal{E}\, z_3^2$ [m]	$\mathcal{E}\, z_4^2$ [m/s]	$\mathcal{E}\, z_5^2$ [m/s]	$\mathcal{E}\, z_6^2$ [m/s]
Ex. 5	0.00002	0.00002	0.00002	0.08999	0.08997	0.08997
Bound	0.00006	0.00006	0.00006	0.09000	0.09000	0.09000

Example 5.7.6 Now we simulate the passive system from Example 4 (denoted by CASC-Passive) and the active system from Example 5 (denoted by CASC-Active) obtained by the CASC design, and the nominal system, when subjected to El Centro earthquake taken from Spencer Jr. et al. (1998) (not white noise). The results for the displacements of each floor are presented in Figure 5.9 on page 209, and the respective velocities in Figure 5.10 on page 210. In these figures, solid line (-) stand for the CASC-Passive system, dashed line (- -) stand for the nominal system, and dotted line (·) stand for the CASC-Active system. These results show the superior performance of the passive system designed via the CASC algorithm over the nominal system (comparing only the passive systems). The active system shows a superior performance over the passive one. This was expected since we imposed a tighter output covariance upper bound.

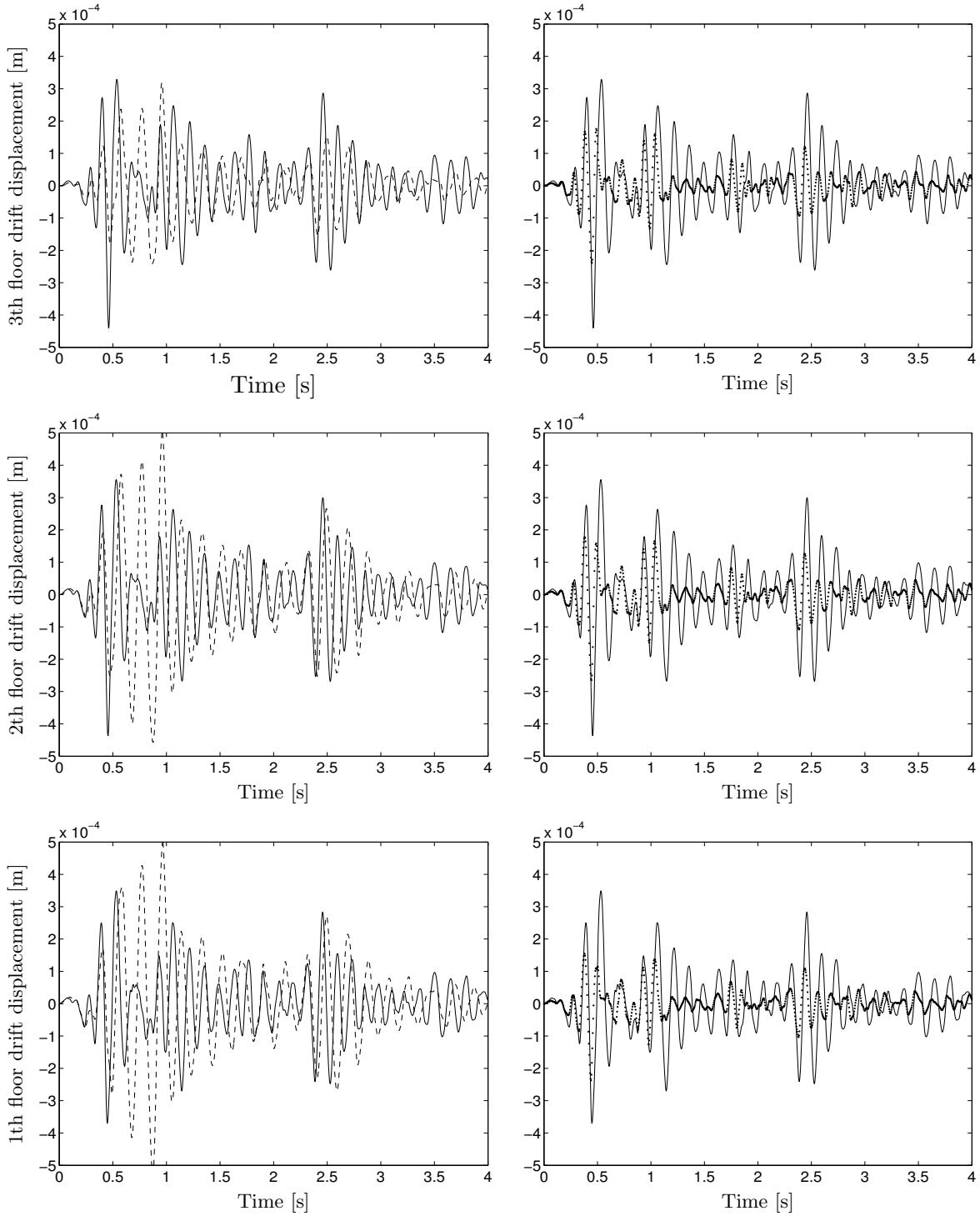

Figure 5.9: Response of the nominal system (--), the CASC-Passive (-), and the CASC-Active (·), due to El Centro earthquake: displacements

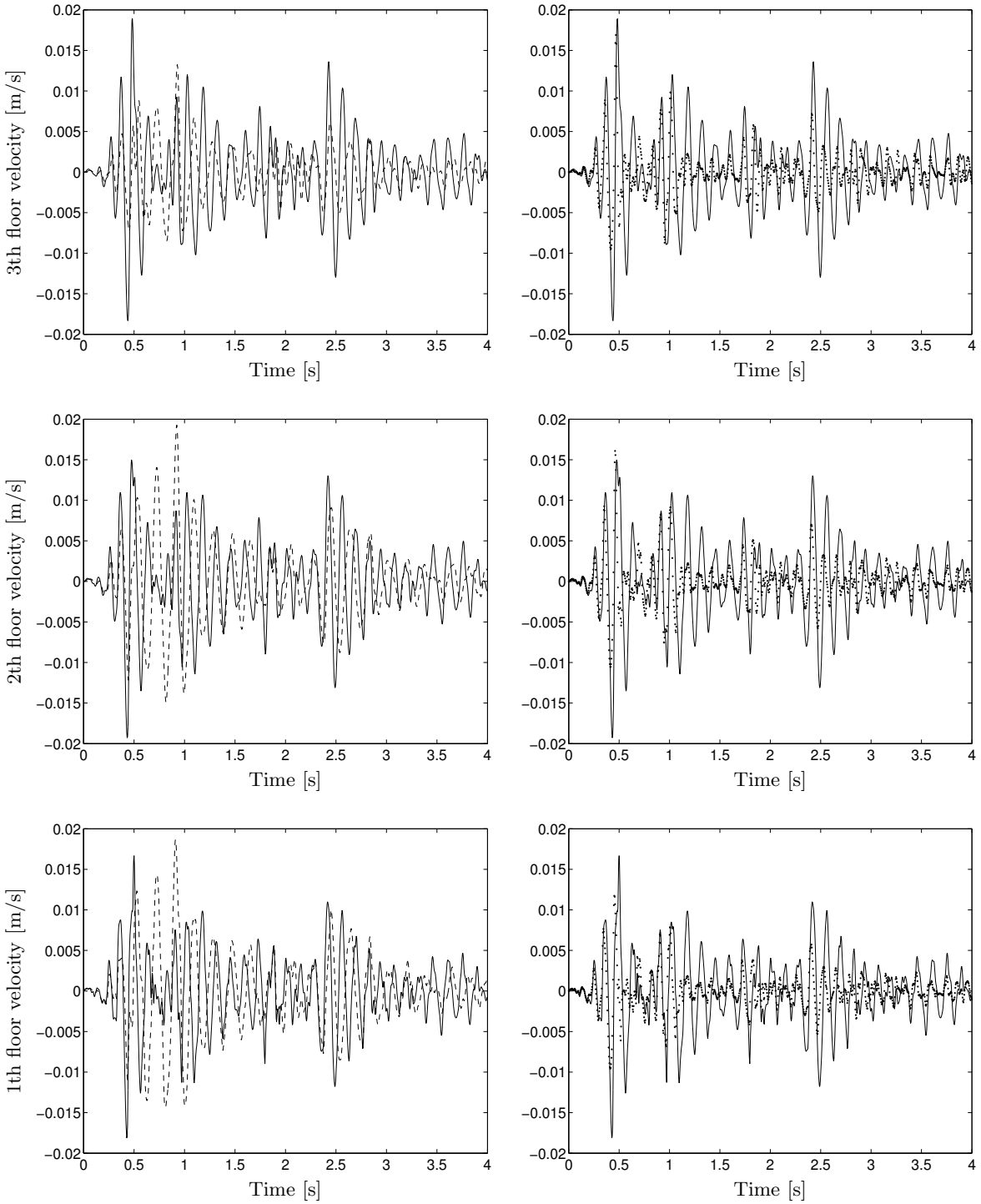

Figure 5.10: Response of the nominal system (--), the CASC-Passive (-), and the CASC-Active (·), due to El Centro earthquake: velocities

Chapter 6

Conclusion

This thesis provides tools which can solve a large class of optimization problems over matrix inequalities. This includes, for instance, systems and control problems, or any other type of engineering problem that can be posed as matrix inequalities. These tools possess similar advantages as the LMI framework, but without its disadvantages. Moreover, in order to use the method, no knowledge of LMIs or no knowledge of how to manipulate MIs to be expressed as LMIs is required. The method has two components: 1) a numerical algorithm that solves a large class of matrix optimization problems; 2) a symbolic convexity checker.

The symbolic convexity checker guarantees that the solution obtained from the numerical solver is indeed a global minimum inside a specific region. The implementation of the NCSDP solver is based on a barrier method. Other methods such primal-dual methods, could have been used. This solver has been shown to possess comparable performance to professional LMI solvers when the dimensions of the matrices involved are large (above 30×30).

In this thesis, a theory of noncommutative functions which results in an algorithm for determining where matrix inequalities are convex is developed. Of independent interest, is a theory of noncommutative quadratic functions and the resulting algorithm which calculates the region where these functions are matrix positive. Furthermore, an LDU algorithm for matrices with noncommutative entries and conditions guaranteeing that the decomposition is successful is also provided.

This thesis has also demonstrated the benefits of simultaneously designing the structure and the controller with an application to civil structures. This opens the doors to the

use of control methods to design structures, even when no control is intended. Since the control methods allow bounds to be placed on the dynamic response, this should be a welcomed improvement in structure design. The algorithm is proposed in the LMI framework, for which very efficient interior point methods are available. The design allows for changes in any parameters that appear affinely in the mass, the damping, and the stiffness matrices. The nonconvex simultaneous structure and control design problem is solved by a sequence of convex subproblems with the help of potential convexifying functions. The performance criteria used in the design is a bound on the output covariance of the closed loop system, but the methodology can incorporate many other convex criteria. This thesis improved the techniques available in the literature in the sense that: the methodology is completely in the LMI framework, having no need to solve a constrained quadratic optimization problem; the technique allows parameters in the mass matrix to be optimized; and the proposed algorithm does not require the Lyapunov matrix to be fixed in the structure design step.

Appendix A

Computer Algorithm for Representing the Quadratic $\mathcal{Q}(\vec{Z})[\vec{H}]$ with $M_{\mathcal{Q}}(\vec{Z})$ and $V(\vec{Z})[\vec{H}]$

In our approach, we are given a noncommutative function $\mathcal{Q}(\vec{Z})[\vec{H}]$, which is quadratic and hereditary in $\vec{H}$ but usually not quadratic in $\vec{Z}$, and we need to express this function as $V(\vec{Z})[\vec{H}]^T\ M_{\mathcal{Q}}(\vec{Z})\ V(\vec{Z})[\vec{H}]$. That means we have to construct the border vector $V(\vec{Z})[\vec{H}]$ and the coefficient matrix $M_{\mathcal{Q}}$. *This representation of* $\mathcal{Q}(\vec{Z})[\vec{H}]$ *may not be unique.*

This section describes a simplified version of the algorithm used. The algorithm is based on a simple pattern match, that is illustrated here for the case were $\vec{H} := \{H_1, H_2\}$. It can be easily expanded for the more general case where $\vec{H}$ has k entries. The algorithm explained here does not assume $\vec{H}$ necessarily symmetric. For the symmetric case, just let $\vec{H} = \vec{H}^T$ and the steps are the same.

1. Expand the quadratic function in H_1 and H_2.

2. In that case, there are four types of quadratic terms involving the H_i :

$$*H_1^T * H_1 * \ , \qquad *H_1^T * H_2 * \ , \qquad *H_2^T * H_1 * \ , \qquad \text{and} \qquad * H_2^T * H_2 * .$$

   The pattern matching symbol $*$ means any expression that does not contain H_i.

3. We work on each one of these quadratic terms $*H_i^T * H_j*$ individually. Let $i = j = 1$. Then find all pattern of the form $*H_1^T * H_1*$. Before the pattern matching is processed,

it is important that all the terms of the expression to be found are collected. That means, if there is an expression like

$$L_1^{1^T} H_1^T B_1 H_1 L_1^1 + \cdots + L_1^{1^T} H_1^T B_m H_1 L_1^1$$

then collect all of the B_i in $A_{1,1} = \sum_{i=1}^{m} B_i$. Follows this procedure, then at the end we may have a sum of terms like:

$$L_1^{1^T} H_1^T A_{1,1} H_1 L_1^1 + L_1^{1^T} H_1^T A_{1,2} H_1 L_2 + \cdots + L_{\ell_1}^{1^T} H_1^T A_{\ell_1,\ell_1} H_1 L_{\ell_1}^1$$

Where the $A_{i,j}$ for $i, j = 1, \ldots, \ell_1$ collect all the terms that match the expression $L_i^{1^T} H_1^T * H_1 L_j^1$. This step was illustrated in the example above, where all the terms that match the expression $L_1^{1^T} H_1^T * H_1 L_1^1$ are collected in the coefficient $A_{1,1}$.

4. The same procedure applies for the terms $*H_1^T * H_2*$, $*H_2^T * H_1*$, and $*H_2^T * H_2*$.

5. Once the finding of all the patterns is finished, the $A_{t,s}$ are the entries of the coefficient matrix $M_{\mathcal{Q}}$, and the $H_i L_j^i$ are the entries of the border vector $V(\vec{Z})[\vec{H}]$.

Appendix B

A Formula for the Hessian for the Uni and Multivariate Cases

B.1 A Formula for the Hessian Term $\mathbb{H}(\delta_X)$ for the Univariate Case

This section computes the formulas for the Hessian term $\mathbb{H}(\delta_X)$, which is obtained from the second-order approximation of the auxiliary potential function

$$\phi(X) = -\log\det F(X).$$

We abbreviate $F(X)^{-1}$ by just F^{-1}. The second directional derivative of the potential function $\phi(X)$, (see (4.35)), is given by

$$\begin{aligned} D^2\phi(X)[\delta_X, \delta_X] = \operatorname{Tr}&\left\{\left(F^{-1}\operatorname{sym}\left\{\sum_{i=1}^{k} A_i\delta_X B_i\right\}\right)^2\right\} \\ &- \operatorname{Tr}\left\{F^{-1}\operatorname{sym}\left\{\sum_{j=1}^{w_1} M_j\delta_X N_j\delta_X T_j\right\}\right\} \\ &- \operatorname{Tr}\left\{F^{-1}\operatorname{sym}\left\{\sum_{j=1+w_1}^{w_2} M_j{\delta_X}^T N_j\delta_X T_j\right\}\right\} \\ &- \operatorname{Tr}\left\{F^{-1}\operatorname{sym}\left\{\sum_{j=1+w_2}^{w_3} M_j\delta_X N_j{\delta_X}^T T_j\right\}\right\} \end{aligned}$$

To proceed with the derivation, let us partition $D^2\phi(X)[\delta_X, \delta_X]$ in four terms, $\mathsf{H}_1(\delta_X)$, $\mathsf{H}_2(\delta_X)$, $\mathsf{H}_3(\delta_X)$, and $\mathsf{H}_4(\delta_X)$, so that we can apply the directional derivative in each one of

the terms separately:

$$\mathsf{H}_1(\delta_X) = \mathrm{Tr}\left\{\left(F^{-1}\,\mathrm{sym}\left\{\sum_{i=1}^{k} A_i\delta_X B_i\right\}\right)^2\right\}$$

$$\mathsf{H}_2(\delta_X) = -\,\mathrm{Tr}\left\{F^{-1}\,\mathrm{sym}\left\{\sum_{j=1}^{w_1} M_j\delta_X N_j\delta_X T_j\right\}\right\}$$

$$\mathsf{H}_3(\delta_X) = -\,\mathrm{Tr}\left\{F^{-1}\,\mathrm{sym}\left\{\sum_{j=1+w_1}^{w_2} M_j\delta_X{}^T N_j\delta_X T_j\right\}\right\}$$

$$\mathsf{H}_4(\delta_X) = -\,\mathrm{Tr}\left\{F^{-1}\,\mathrm{sym}\left\{\sum_{j=1+w_2}^{w_3} M_j\delta_X N_j\delta_X{}^T T_j\right\}\right\}$$

For this derivation, we have omitted the fraction 1/2 that multiplies the Hessian. At the end, we just need to multiply the final result by 1/2.

B.1.1 The Term H_1

Expanding the first term $\mathsf{H}_1(\delta_X)$ we obtain

$$\begin{aligned}\mathsf{H}_1(\delta_X) = \\ \mathrm{Tr}\Bigg\{ & F^{-1}\sum_{i=1}^{k} A_i\delta_X B_i F^{-1}\sum_{j=1}^{k} A_j\delta_X B_j + F^{-1}\sum_{i=1}^{k} A_i\delta_X B_i F^{-1}\sum_{j=1}^{k} B_j^T\delta_X^T A_j^T \\ & + F^{-1}\sum_{i=1}^{k} B_i^T\delta_X^T A_i^T F^{-1}\sum_{j=1}^{k} A_j\delta_X B_j + F^{-1}\sum_{i=1}^{k} B_i^T\delta_X^T A_i^T F^{-1}\sum_{j=1}^{k} B_j^T\delta_X^T A_j^T\Bigg\}\end{aligned}$$

We should now apply the directional derivative of the above term as a function of δ_X along the direction δ_V. Doing so, we have

$$\begin{aligned}D\mathsf{H}_1(\delta_X)[\delta_V] = \\ & \mathrm{Tr}\left\{F^{-1}\sum_{i=1}^{k} A_i\delta_V B_i F^{-1}\sum_{j=1}^{k} A_j\delta_X B_j + F^{-1}\sum_{j=1}^{k} A_j\delta_X B_j F^{-1}\sum_{i=1}^{k} A_i\delta_V B_i\right\} \\ + & \mathrm{Tr}\left\{F^{-1}\sum_{i=1}^{k} A_i\delta_V B_i F^{-1}\sum_{j=1}^{k} B_j^T\delta_X^T A_j^T + F^{-1}\sum_{j=1}^{k} A_j\delta_X B_j F^{-1}\sum_{i=1}^{k} B_i^T\delta_V^T A_i^T\right\} \\ + & \mathrm{Tr}\left\{F^{-1}\sum_{i=1}^{k} B_i^T\delta_V^T A_i^T F^{-1}\sum_{j=1}^{k} A_j\delta_X B_j + F^{-1}\sum_{j=1}^{k} B_j^T\delta_X^T A_j^T F^{-1}\sum_{i=1}^{k} A_i\delta_V B_i\right\} \\ + & \mathrm{Tr}\left\{F^{-1}\sum_{i=1}^{k} B_i^T\delta_V^T A_i^T F^{-1}\sum_{j=1}^{k} B_j^T\delta_X^T A_j^T + F^{-1}\sum_{j=1}^{k} B_j^T\delta_X^T A_j^T F^{-1}\sum_{i=1}^{k} B_i^T\delta_V^T A_i^T\right\}\end{aligned}$$

Finally, using the cyclic property of the trace operator we can collect on δ_V and δ_V^T, obtaining

$$\begin{aligned} D\mathsf{H}_1(\delta_X)[\delta_V] = 2\,\mathrm{Tr}\Bigg\{ \delta_V \Bigg[\sum_{i=1}^{k} B_i F^{-1} \left(\sum_{j=1}^{k} A_j \delta_X B_j \right) F^{-1} A_i \\ + \sum_{i=1}^{k} B_i F^{-1} \left(\sum_{j=1}^{k} B_j^T \delta_X^T A_j^T \right) F^{-1} A_i \Bigg] \Bigg\} \\ + 2\,\mathrm{Tr}\Bigg\{ \Bigg[\sum_{i=1}^{k} A_i^T F^{-1} \left(\sum_{j=1}^{k} A_j \delta_X B_j \right) F^{-1} B_i^T \\ + \sum_{i=1}^{k} A_i^T F^{-1} \left(\sum_{j=1}^{k} B_j^T \delta_X^T A_j^T \right) F^{-1} B_i^T \Bigg] \delta_V^T \Bigg\} \end{aligned}$$

This expression is equivalently written as

$$D\mathsf{H}_1(\delta_X)[\delta_V] = \mathrm{Tr}\left\{ \delta_V \mathbb{H}_1(\delta_X)^T + \mathbb{H}_1(\delta_X) \delta_V^T \right\} \tag{B.1}$$

with

$$\begin{aligned} \mathbb{H}_1(\delta_X) = 2 \sum_{i=1}^{k} A_i^T F^{-1} \left(\sum_{j=1}^{k} A_j \delta_X B_j \right) F^{-1} B_i^T \\ + 2 \sum_{i=1}^{k} A_i^T F^{-1} \left(\sum_{j=1}^{k} B_j^T \delta_X^T A_j^T \right) F^{-1} B_i^T \end{aligned}$$

B.1.2 The Term H_2

Expanding the second term $\mathsf{H}_2(\delta_X)$ we obtain

$$\mathsf{H}_2(\delta_X) = -\,\mathrm{Tr}\left\{ F^{-1} \left(\sum_{j=1}^{w_1} M_j \delta_X N_j \delta_X T_j + \sum_{j=1}^{w_1} T_j^T \delta_X^T N_j^T \delta_X^T M_j^T \right) \right\}$$

Applying the directional derivative of the above term as a function of δ_X along the direction δ_V gives

$$\begin{aligned} D\mathsf{H}_2(\delta_X)[\delta_V] = -\,\mathrm{Tr}\left\{ F^{-1} \sum_{j=1}^{w_1} M_j \delta_V N_j \delta_X T_j + F^{-1} \sum_{j=1}^{w_1} M_j \delta_X N_j \delta_V T_j \right\} \\ - \mathrm{Tr}\left\{ F^{-1} \sum_{j=1}^{w_1} T_j^T {\delta_V}^T N_j^T {\delta_X}^T M_j^T + F^{-1} \sum_{j=1}^{w_1} T_j^T {\delta_X}^T N_j^T {\delta_V}^T M_j^T \right\} \end{aligned}$$

Collecting the above expression on δ_V and δ_V^T, we obtain

$$D\mathsf{H}_2(\delta_X)[\delta_V] = -\operatorname{Tr}\left\{\delta_V\left[\sum_{j=1}^{w_1} N_j\delta_X T_j F^{-1} M_j + \sum_{j=1}^{w_1} T_j F^{-1} M_j \delta_X N_j\right]\right\} - \operatorname{Tr}\left\{\left[\sum_{j=1}^{w_1} N_j^T \delta_X^T M_j^T F^{-1} T_j^T + \sum_{j=1}^{w_1} M_j^T F^{-1} T_j^T \delta_X^T N_j^T\right]\delta_V^T\right\}$$

This expression is equivalently written as

$$D\mathsf{H}_2(\delta_X)[\delta_V] = \operatorname{Tr}\left\{\delta_V \mathbb{H}_2(\delta_X)^T + \mathbb{H}_2(\delta_X)\delta_V^T\right\} \tag{B.2}$$

with

$$\mathbb{H}_2(\delta_X) = -\sum_{j=1}^{w_1} N_j^T \delta_X^T M_j^T F^{-1} T_j^T + \sum_{j=1}^{w_1} M_j^T F^{-1} T_j^T \delta_X^T N_j^T$$

B.1.3 The Term H_3

Expanding the second term $\mathsf{H}_3(\delta_X)$ we obtain

$$\mathsf{H}_3(\delta_X) = -\operatorname{Tr}\left\{F^{-1}\left(\sum_{j=1+w_1}^{w_2} M_j {\delta_X}^T N_j \delta_X T_j + \sum_{j=1+w_1}^{w_2} T_j^T {\delta_X}^T N_j^T \delta_X M_j^T\right)\right\}$$

Applying the directional derivative of the above term as a function of δ_X along the direction δ_V gives

$$D\mathsf{H}_3(\delta_X)[\delta_V] = -\operatorname{Tr}\left\{F^{-1}\sum_{j=1+w_1}^{w_2} M_j {\delta_V}^T N_j \delta_X T_j + F^{-1}\sum_{j=1+w_1}^{w_2} M_j {\delta_X}^T N_j \delta_V T_j\right\} - \operatorname{Tr}\left\{F^{-1}\sum_{j=1+w_1}^{w_2} T_j^T {\delta_V}^T N_j^T \delta_X M_j^T + F^{-1}\sum_{j=1+w_1}^{w_2} T_j^T {\delta_X}^T N_j^T \delta_V M_j^T\right\}$$

Collecting the above expression on δ_V and δ_V^T, we obtain

$$D\mathsf{H}_3(\delta_X)[\delta_V] = -\operatorname{Tr}\left\{\delta_V\left[\sum_{j=1+w_1}^{w_2} T_j F^{-1} M_j \delta_X^T N_j + \sum_{j=1+w_1}^{w_2} M_j^T F^{-1} T_j^T \delta_X^T N_j^T\right]\right\} - \operatorname{Tr}\left\{\left[\sum_{j=1+w_1}^{w_2} N_j \delta_X T_j F^{-1} M_j + \sum_{j=1+w_1}^{w_2} N_j^T \delta_X M_j^T F^{-1} T_j^T\right]\delta_V^T\right\}$$

This expression is equivalently written as

$$D\mathsf{H}_3(\delta_X)[\delta_V] = \operatorname{Tr}\left\{\delta_V \mathbb{H}_3(\delta_X)^T + \mathbb{H}_3(\delta_X)\delta_V^T\right\} \tag{B.3}$$

with

$$\mathbb{H}_3(\delta_X) = -\sum_{j=1+w_1}^{w_2} N_j \delta_X T_j F^{-1} M_j + \sum_{j=1+w_1}^{w_2} N_j^T \delta_X M_j^T F^{-1} T_j^T$$

B.1.4 The Term H_4

Expanding the second term $\mathsf{H}_4(\delta_X)$ we obtain

$$\mathsf{H}_4(\delta_X) = -\operatorname{Tr}\left\{ F^{-1} \left(\sum_{j=1+w_2}^{w_3} M_j \delta_X N_j {\delta_X}^T T_j + \sum_{j=1+w_2}^{w_3} T_j^T \delta_X N_j^T {\delta_X}^T M_j^T \right) \right\}$$

Applying the directional derivative of the above term as a function of δ_X along the direction δ_V gives

$$\begin{aligned} D\mathsf{H}_4(\delta_X)[\delta_V] = &-\operatorname{Tr}\left\{ F^{-1} \sum_{j=1+w_2}^{w_3} M_j \delta_V N_j {\delta_X}^T T_j + F^{-1} \sum_{j=1+w_2}^{w_3} M_j \delta_X N_j {\delta_V}^T T_j \right\} \\ &-\operatorname{Tr}\left\{ F^{-1} \sum_{j=1+w_2}^{w_3} T_j^T \delta_V N_j^T {\delta_X}^T M_j^T + F^{-1} \sum_{j=1+w_2}^{w_3} T_j^T \delta_X N_j^T {\delta_V}^T M_j^T \right\} \end{aligned}$$

Collecting the above expression on δ_V and δ_V^T, we obtain

$$\begin{aligned} D\mathsf{H}_4(\delta_X)[\delta_V] = &-\operatorname{Tr}\left\{ \delta_V \left[\sum_{j=1+w_2}^{w_3} N_j \delta_X^T T_j F^{-1} M_j + \sum_{j=1+w_2}^{w_3} N_j^T \delta_X^T M_j^T F^{-1} T_j^T \right] \right\} \\ &-\operatorname{Tr}\left\{ \left[\sum_{j=1+w_2}^{w_3} T_j F^{-1} M_j \delta_X N_j + \sum_{j=1+w_2}^{w_2} M_j^T F^{-1} T_j^T \delta_X N_j^T \right] \delta_V^T \right\} \end{aligned}$$

This expression is equivalently written as

$$D\mathsf{H}_4(\delta_X)[\delta_V] = \operatorname{Tr}\left\{ \delta_V \mathbb{H}_4(\delta_X)^T + \mathbb{H}_4(\delta_X) \delta_V^T \right\} \tag{B.4}$$

with

$$\mathbb{H}_4(\delta_X) = \sum_{j=1+w_2}^{w_3} T_j F^{-1} M_j \delta_X N_j + \sum_{j=1+w_2}^{w_2} M_j^T F^{-1} T_j^T \delta_X N_j^T$$

B.1.5 The Final Term $\mathbb{H}(\delta_X)$

The final term for the Hessian map $\mathbb{H}(\delta_X)$, which is obtained from the second directional derivative of the potential function, $D^2\phi(X)[\delta_X, \delta_X]$, is now readily available from the expressions given in (B.1)–(B.4). Thus, we have

$$D\left(\frac{1}{2} D^2\phi(X)[\delta_X, \delta_X]\right)[\delta_V] = \frac{1}{2} \sum_{i=1}^{4} D\mathsf{H}_i(\delta_X)[\delta_V] = \operatorname{Tr}\left\{ \delta_V \mathbb{H}(\delta_X)^T + \mathbb{H}(\delta_X) \delta_V^T \right\}$$

with

$$\mathbb{H}(\delta_X) = \frac{1}{2}\sum_{i=1}^{4}\mathbb{H}_i(\delta_X)$$

Thus, the term $\mathbb{H}(\delta_X)$ is naturally given by

$$\begin{aligned}\mathbb{H}(\delta_X) = &\sum_{i=1}^{k} A_i^T F^{-1}\left(\sum_{j=1}^{k} A_j \delta_X B_j\right) F^{-1} B_i^T + \sum_{i=1}^{k} A_i^T F^{-1}\left(\sum_{j=1}^{k} B_j^T \delta_X^T A_j^T\right) F^{-1} B_i^T \\ &- \frac{1}{2}\sum_{j=1}^{w_1} N_j^T \delta_X^T M_j^T F^{-1} T_j^T + M_j^T F^{-1} T_j^T \delta_X^T N_j^T \\ &- \frac{1}{2}\sum_{j=1+w_1}^{w_2} N_j \delta_X T_j F^{-1} M_j + N_j^T \delta_X M_j^T F^{-1} T_j^T \\ &- \frac{1}{2}\sum_{j=1+w_2}^{w_3} T_j F^{-1} M_j \delta_X N_j + M_j^T F^{-1} T_j^T \delta_X N_j^T\end{aligned}$$

as stated in Lemma 4.3.3.

B.2 A Formula for the Hessian Term $\mathbb{H}_{ts}(\delta_{X_s})$ for the Multivariate Case

This section presents the derivation of the formulas for the Hessian map $\mathbb{H}_{ts}(\delta_{X_s})$ for the multivariate case. It is important to emphasize that the formulas in this section assumes that $\delta_{X_t} \neq \delta_{X_s}$, since the particular case where $\delta_{X_t} = \delta_{X_s} = \delta_X$ was considered in the previous section for the univariate case.

The second directional derivative of the potential function, as presented in (4.74), is given by

$$\begin{aligned}\mathrm{D}^2\,\Theta(\overrightarrow{X})[\delta_{X_t}, \delta_{X_s}] = &\\ &\mathrm{Tr}\left\{\sum_{i=1}^{m} F_i^{-1}\,\mathrm{sym}\left\{\sum_{\ell=1}^{k(i,s)} A_\ell^{i,s}\delta_{X_s}B_\ell^{i,s}\right\} F_i^{-1}\,\mathrm{sym}\left\{\sum_{\eta=1}^{k(i,t)} A_\eta^{i,t}\delta_{X_t}B_\eta^{i,t}\right\}\right\} \\ -\,\mathrm{Tr}\Bigg\{\sum_{i=1}^{m} F_i^{-1}\,\mathrm{sym}\Bigg\{&\sum_{\ell=1}^{w_1(i,ts)} M_\ell^{i,ts}\delta_{X_t}N_\ell^{i,ts}\delta_{X_s}T_\ell^{i,ts} + \sum_{\ell=1+w_1(i,ts)}^{w_2(i,ts)} M_\ell^{i,ts}\delta_{X_t}^T N_\ell^{i,ts}\delta_{X_s}T_\ell^{i,ts} \\ &+ \sum_{\ell=1+w_2(i,ts)}^{w_3(i,ts)} M_\ell^{i,ts}\delta_{X_t}N_\ell^{i,ts}\delta_{X_s}^T T_\ell^{i,ts} + \sum_{\ell=1+w_3(i,ts)}^{w_4(i,ts)} M_\ell^{i,ts}\delta_{X_t}^T N_\ell^{i,ts}\delta_{X_s}^T T_\ell^{i,ts}\Bigg\}\Bigg\}\end{aligned}$$

To proceed with the derivation, let us partition $\mathrm{D}^2\,\Theta(\overrightarrow{X})[\delta_{X_t},\delta_{X_s}]$ in five terms, H_1, H_2, H_3, H_4, and H_5, so that we can apply the directional derivative in each one of the terms separately:

$$\mathsf{H}_1 = \mathrm{Tr}\left\{\sum_{i=1}^{m} F_i^{-1}\,\mathrm{sym}\left\{\sum_{\ell=1}^{k(i,s)} A_\ell^{i,s}\delta_{X_s}B_\ell^{i,s}\right\}F_i^{-1}\,\mathrm{sym}\left\{\sum_{\eta=1}^{k(i,t)} A_\eta^{i,t}\delta_{X_t}B_\eta^{i,t}\right\}\right\}$$

$$\mathsf{H}_2 = -\,\mathrm{Tr}\left\{\sum_{i=1}^{m} F_i^{-1}\,\mathrm{sym}\left\{\sum_{\ell=1}^{w_1(i,ts)} M_\ell^{i,ts}\delta_{X_t}N_\ell^{i,ts}\delta_{X_s}T_\ell^{i,ts}\right\}\right\}$$

$$\mathsf{H}_3 = -\,\mathrm{Tr}\left\{\sum_{i=1}^{m} F_i^{-1}\,\mathrm{sym}\left\{\sum_{\ell=1+w_1(i,ts)}^{w_2(i,ts)} M_\ell^{i,ts}\delta_{X_t}^T N_\ell^{i,ts}\delta_{X_s}T_\ell^{i,ts}\right\}\right\}$$

$$\mathsf{H}_4 = -\,\mathrm{Tr}\left\{\sum_{i=1}^{m} F_i^{-1}\,\mathrm{sym}\left\{\sum_{\ell=1+w_2(i,ts)}^{w_3(i,ts)} M_\ell^{i,ts}\delta_{X_t}N_\ell^{i,ts}\delta_{X_s}^T T_\ell^{i,ts}\right\}\right\}$$

$$\mathsf{H}_5 = -\,\mathrm{Tr}\left\{\sum_{i=1}^{m} F_i^{-1}\,\mathrm{sym}\left\{\sum_{\ell=1+w_3(i,ts)}^{w_4(i,ts)} M_\ell^{i,ts}\delta_{X_t}^T N_\ell^{i,ts}\delta_{X_s}^T T_\ell^{i,ts}\right\}\right\}$$

For this derivation, we have omitted the fraction 1/2 that multiplies the Hessian. At the end, we just need to multiply the final result by 1/2.

B.2.1 The Term H_1

Expanding the first term H_1 we obtain

$$\begin{aligned}\mathsf{H}_1 = \mathrm{Tr}\Bigg\{&\sum_{i=1}^{m} F_i^{-1}\sum_{\ell=1}^{k(i,s)} A_\ell^{i,s}\delta_{X_s}B_\ell^{i,s}F_i^{-1}\sum_{\eta=1}^{k(i,t)} A_\eta^{i,t}\delta_{X_t}B_\eta^{i,t}\\ &+\sum_{i=1}^{m} F_i^{-1}\sum_{\ell=1}^{k(i,s)} A_\ell^{i,s}\delta_{X_s}B_\ell^{i,s}F_i^{-1}\sum_{\eta=1}^{k(i,t)} (B_\eta^{i,t})^T\delta_{X_t}^T(A_\eta^{i,t})^T\\ &+\sum_{i=1}^{m} F_i^{-1}\sum_{\ell=1}^{k(i,s)} (B_\ell^{i,s})^T\delta_{X_s}^T(A_\ell^{i,s})^T F_i^{-1}\sum_{\eta=1}^{k(i,t)} A_\eta^{i,t}\delta_{X_t}B_\eta^{i,t}\\ &+\sum_{i=1}^{m} F_i^{-1}\sum_{\ell=1}^{k(i,s)} (B_\ell^{i,s})^T\delta_{X_s}^T(A_\ell^{i,s})^T F_i^{-1}\sum_{\eta=1}^{k(i,t)} (B_\eta^{i,t})^T\delta_{X_t}^T(A_\eta^{i,t})^T\Bigg\}\end{aligned}$$

We should now apply the directional derivative of the above term as a function of δ_{X_t} along the direction δ_{V_t}. Doing so, we have

$$\begin{aligned}
D\mathsf{H}_1(\delta_{X_t})[\delta_{V_t}] = \mathrm{Tr}\Bigg\{ & \sum_{i=1}^{m} F_i^{-1} \sum_{\ell=1}^{k(i,s)} A_\ell^{i,s} \delta_{X_s} B_\ell^{i,s} F_i^{-1} \sum_{\eta=1}^{k(i,t)} A_\eta^{i,t} \delta_{V_t} B_\eta^{i,t} \\
& + \sum_{i=1}^{m} F_i^{-1} \sum_{\ell=1}^{k(i,s)} A_\ell^{i,s} \delta_{X_s} B_\ell^{i,s} F_i^{-1} \sum_{\eta=1}^{k(i,t)} (B_\eta^{i,t})^T \delta_{V_t}^T (A_\eta^{i,t})^T \\
& + \sum_{i=1}^{m} F_i^{-1} \sum_{\ell=1}^{k(i,s)} (B_\ell^{i,s})^T \delta_{X_s}^T (A_\ell^{i,s})^T F_i^{-1} \sum_{\eta=1}^{k(i,t)} A_\eta^{i,t} \delta_{V_t} B_\eta^{i,t} \\
& + \sum_{i=1}^{m} F_i^{-1} \sum_{\ell=1}^{k(i,s)} (B_\ell^{i,s})^T \delta_{X_s}^T (A_\ell^{i,s})^T F_i^{-1} \sum_{\eta=1}^{k(i,t)} (B_\eta^{i,t})^T \delta_{V_t}^T (A_\eta^{i,t})^T \Bigg\}
\end{aligned}$$

Finally, using the cyclic property of the trace operator we can collect on δ_{V_t} and $\delta_{V_t}^T$, obtaining

$$\begin{aligned}
D\mathsf{H}_1(\delta_{X_t})[\delta_{V_t}] = \mathrm{Tr}\Bigg\{ & \delta_{V_t} \Bigg[\sum_{i=1}^{m} \sum_{\ell=1}^{k(i,s)} \sum_{\eta=1}^{k(i,t)} \Big(B_\eta^{i,t} F_i^{-1} A_\ell^{i,s} \delta_{X_s} B_\ell^{i,s} F_i^{-1} A_\eta^{i,t} \\
& \qquad + B_\eta^{i,t} F_i^{-1} (B_\ell^{i,s})^T \delta_{X_s}^T (A_\ell^{i,s})^T F_i^{-1} A_\eta^{i,t} \Big) \Bigg] \\
& + \Bigg[\sum_{i=1}^{m} \sum_{\ell=1}^{k(i,s)} \sum_{\eta=1}^{k(i,t)} (A_\eta^{i,t})^T F_i^{-1} A_\ell^{i,s} \delta_{X_s} B_\ell^{i,s} F_i^{-1} (B_\eta^{i,t})^T \\
& \qquad + (A_\eta^{i,t})^T F_i^{-1} (B_\ell^{i,s})^T \delta_{X_s}^T (A_\ell^{i,s})^T F_i^{-1} (B_\eta^{i,t})^T \Big) \Bigg] \delta_{V_t}^T \Bigg\}
\end{aligned}$$

This expression is equivalently written as

$$D\mathsf{H}_1(\delta_{X_t})[\delta_{V_t}] = \mathrm{Tr}\left\{ \delta_{V_t} \mathbb{H}_1(\delta_{X_s})^T + \mathbb{H}_1(\delta_{X_s}) \delta_{V_t}^T \right\} \tag{B.5}$$

with

$$\begin{aligned}
\mathbb{H}_1(\delta_{X_s}) = & \sum_{i=1}^{m} \sum_{\ell=1}^{k(i,s)} \sum_{\eta=1}^{k(i,t)} \Big((A_\eta^{i,t})^T F_i^{-1} A_\ell^{i,s} \delta_{X_s} B_\ell^{i,s} F_i^{-1} (B_\eta^{i,t})^T \Big) \\
& + \sum_{i=1}^{m} \sum_{\ell=1}^{k(i,s)} \sum_{\eta=1}^{k(i,t)} \Big((A_\eta^{i,t})^T F_i^{-1} (B_\ell^{i,s})^T \delta_{X_s}^T (A_\ell^{i,s})^T F_i^{-1} (B_\eta^{i,t})^T \Big)
\end{aligned}$$

B.2.2 The Term H_2

Expanding the second term H_2 we obtain

$$\mathsf{H}_2 = -\operatorname{Tr}\left\{\sum_{i=1}^{m} F_i^{-1}\left(\sum_{\ell=1}^{w_1(i,ts)} M_\ell^{i,ts}\delta_{X_t}N_\ell^{i,ts}\delta_{X_s}T_\ell^{i,ts} + \sum_{\ell=1}^{w_1(i,ts)} (T_\ell^{i,ts})^T\delta_{X_s}{}^T(N_\ell^{i,ts})^T\delta_{X_t}{}^T(M_\ell^{i,ts})^T\right)\right\}$$

Applying the directional derivative of the above term as a function of δ_{X_t} along the direction δ_{V_t} gives

$$D\mathsf{H}_2(\delta_{X_t})[\delta_{V_t}] = -\operatorname{Tr}\left\{\sum_{i=1}^{m} F_i^{-1}\left(\sum_{\ell=1}^{w_1(i,ts)} M_\ell^{i,ts}\delta_{V_t}N_\ell^{i,ts}\delta_{X_s}T_\ell^{i,ts} + \sum_{\ell=1}^{w_1(i,ts)} (T_\ell^{i,ts})^T\delta_{X_s}{}^T(N_\ell^{i,ts})^T\delta_{V_t}{}^T(M_\ell^{i,ts})^T\right)\right\}$$

Collecting the above expression on δ_{V_t} and $\delta_{V_t}^T$, we obtain

$$D\mathsf{H}_2(\delta_{X_t})[\delta_{V_t}] = -\operatorname{Tr}\left\{\delta_{V_t}\left[\sum_{i=1}^{m}\sum_{\ell=1}^{w_1(i,ts)} N_\ell^{i,ts}\delta_{X_s}T_\ell^{i,ts}F_i^{-1}M_\ell^{i,ts}\right] + \left[\sum_{i=1}^{m}\sum_{\ell=1}^{w_1(i,ts)} (M_\ell^{i,ts})^T F_i^{-1}(T_\ell^{i,ts})^T\delta_{X_s}^T(N_\ell^{i,ts})^T\right]\delta_{V_t}^T\right\}$$

This expression is equivalently written as

$$D\mathsf{H}_2(\delta_{X_t})[\delta_{V_t}] = \operatorname{Tr}\left\{\delta_{V_t}\mathbb{H}_2(\delta_{X_s})^T + \mathbb{H}_2(\delta_{X_s})\delta_{V_t}^T\right\} \tag{B.6}$$

with

$$\mathbb{H}_2(\delta_{X_s}) = -\sum_{i=1}^{m}\sum_{\ell=1}^{w_1(i,ts)} (M_\ell^{i,ts})^T F_i^{-1}(T_\ell^{i,ts})^T\delta_{X_s}^T(N_\ell^{i,ts})^T$$

B.2.3 The Term H_3

Expanding the second term H_3 we obtain

$$\mathsf{H}_3 = -\operatorname{Tr}\left[\sum_{i=1}^{m} F_i^{-1}\left(\sum_{\ell=1+w_1(i,ts)}^{w_2(i,ts)} M_\ell^{i,ts}\delta_{X_t}{}^T N_\ell^{i,ts}\delta_{X_s}T_\ell^{i,ts} + \sum_{\ell=1+w_1(i,ts)}^{w_2(i,ts)} (T_\ell^{i,ts})^T\delta_{X_s}{}^T(N_\ell^{i,ts})^T\delta_{X_t}(M_\ell^{i,ts})^T\right)\right]$$

Applying the directional derivative of the above term as a function of δ_{X_t} along the direction δ_{V_t} gives

$$D\mathsf{H}_3(\delta_{X_t})[\delta_{V_t}] = -\operatorname{Tr}\Bigg[\sum_{i=1}^{m} F_i^{-1}\Bigg(\sum_{\ell=1+w_1(i,ts)}^{w_2(i,ts)} M_\ell^{i,ts}{\delta_{V_t}}^T N_\ell^{i,ts}\delta_{X_s}T_\ell^{i,ts} + \sum_{\ell=1+w_1(i,ts)}^{w_2(i,ts)} (T_\ell^{i,ts})^T{\delta_{X_s}}^T(N_\ell^{i,ts})^T\delta_{V_t}(M_\ell^{i,ts})^T\Bigg)\Bigg]$$

Collecting the above expression on δ_{V_t} and $\delta_{V_t}^T$, we obtain

$$D\mathsf{H}_3(\delta_{X_t})[\delta_{V_t}] = -\operatorname{Tr}\Bigg\{\delta_{V_t}\left[\sum_{i=1}^{m}\sum_{\ell=1+w_1(i,ts)}^{w_2(i,ts)} (M_\ell^{i,ts})^T F_i^{-1}(T_\ell^{i,ts})^T\delta_{X_s}^T(N_\ell^{i,ts})^T\right] + \left[\sum_{i=1}^{m}\sum_{\ell=1+w_1(i,ts)}^{w_2(i,ts)} N_\ell^{i,ts}\delta_{X_s}T_\ell^{i,ts}F_i^{-1}M_\ell^{i,ts}\right]\delta_{V_t}^T\Bigg\}$$

This expression is equivalently written as

$$D\mathsf{H}_3(\delta_{X_t})[\delta_{V_t}] = \operatorname{Tr}\left\{\delta_{V_t}\mathbb{H}_3(\delta_{X_s})^T + \mathbb{H}_3(\delta_{X_s})\delta_{V_t}^T\right\} \tag{B.7}$$

with

$$\mathbb{H}_3(\delta_{X_s}) = \sum_{i=1}^{m}\sum_{\ell=1+w_1(i,ts)}^{w_2(i,ts)} N_\ell^{i,ts}\delta_{X_s}T_\ell^{i,ts}F_i^{-1}M_\ell^{i,ts}$$

B.2.4 The Term H_4

Expanding the second term H_4 we obtain

$$\mathsf{H}_4 = -\operatorname{Tr}\Bigg\{\sum_{i=1}^{m} F_i^{-1}\Bigg(\sum_{\ell=1+w_2(i,ts)}^{w_3(i,ts)} M_\ell^{i,ts}\delta_{X_t}N_\ell^{i,ts}{\delta_{X_s}}^T T_\ell^{i,ts} + \sum_{\ell=1+w_2(i,ts)}^{w_3(i,ts)} (T_\ell^{i,ts})^T\delta_{X_s}(N_\ell^{i,ts})^T{\delta_{X_t}}^T(M_\ell^{i,ts})^T\Bigg)\Bigg\}$$

Applying the directional derivative of the above term as a function of δ_{X_t} along the direction δ_{V_t} gives

$$D\mathsf{H}_4(\delta_{X_t})[\delta_{V_t}] = -\operatorname{Tr}\Bigg\{\sum_{i=1}^{m} F_i^{-1}\Bigg(\sum_{\ell=1+w_2(i,ts)}^{w_3(i,ts)} M_\ell^{i,ts}\delta_{V_t}N_\ell^{i,ts}{\delta_{X_s}}^T T_\ell^{i,ts} + \sum_{\ell=1+w_2(i,ts)}^{w_3(i,ts)} (T_\ell^{i,ts})^T\delta_{X_s}(N_\ell^{i,ts})^T{\delta_{V_t}}^T(M_\ell^{i,ts})^T\Bigg)\Bigg\}$$

Collecting the above expression on δ_{V_t} and $\delta_{V_t}^T$, we obtain

$$D\mathsf{H}_4(\delta_{X_t})[\delta_{V_t}] = -\operatorname{Tr}\left\{\delta_{V_t}\left[\sum_{i=1}^{m}\sum_{\ell=1+w_2(i,ts)}^{w_3(i,ts)} N_\ell^{i,ts}\delta_{X_s}^T T_\ell^{i,ts} F_i^{-1} M_\ell^{i,ts}\right]\right\}$$
$$-\operatorname{Tr}\left\{\left[\sum_{i=1}^{m}\sum_{\ell=1+w_2(i,ts)}^{w_3(i,ts)} (M_\ell^{i,ts})^T F_i^{-1}(T_\ell^{i,ts})^T \delta_{X_s}(N_\ell^{i,ts})^T\right]\delta_{V_t}^T\right\}$$

This expression is equivalently written as

$$D\mathsf{H}_4(\delta_{X_t})[\delta_{V_t}] = \operatorname{Tr}\left\{\delta_{V_t}\mathbb{H}_4(\delta_{X_s})^T + \mathbb{H}_4(\delta_{X_s})\delta_{V_t}^T\right\} \tag{B.8}$$

with

$$\mathbb{H}_4(\delta_{X_s}) = \sum_{i=1}^{m}\sum_{\ell=1+w_2(i,ts)}^{w_3(i,ts)} (M_\ell^{i,ts})^T F_i^{-1}(T_\ell^{i,ts})^T \delta_{X_s}(N_\ell^{i,ts})^T$$

B.2.5 The Term H_5

Expanding the second term H_5 we obtain

$$\mathsf{H}_5 = -\operatorname{Tr}\left\{\sum_{i=1}^{m} F_i^{-1}\left(\sum_{\ell=1+w_3(i,ts)}^{w_r(i,ts)} M_\ell^{i,ts}{\delta_{X_t}}^T N_\ell^{i,ts}{\delta_{X_s}}^T T_\ell^{i,ts}\right.\right.$$
$$\left.\left.+\sum_{\ell=1+w_3(i,ts)}^{w_4(i,ts)} (T_\ell^{i,ts})^T\delta_{X_s}(N_\ell^{i,ts})^T\delta_{X_t}(M_\ell^{i,ts})^T\right)\right\}$$

Applying the directional derivative of the above term as a function of δ_{X_t} along the direction δ_{V_t} gives

$$D\mathsf{H}_5(\delta_{X_t})[\delta_{V_t}] = -\operatorname{Tr}\left\{\sum_{i=1}^{m} F_i^{-1}\left(\sum_{\ell=1+w_3(i,ts)}^{w_r(i,ts)} M_\ell^{i,ts}{\delta_{V_t}}^T N_\ell^{i,ts}{\delta_{X_s}}^T T_\ell^{i,ts}\right.\right.$$
$$\left.\left.+\sum_{\ell=1+w_3(i,ts)}^{w_4(i,ts)} (T_\ell^{i,ts})^T\delta_{X_s}(N_\ell^{i,ts})^T\delta_{V_t}(M_\ell^{i,ts})^T\right)\right\}$$

Collecting the above expression on δ_{V_t} and $\delta_{V_t}^T$, we obtain

$$D\mathsf{H}_5(\delta_{X_t})[\delta_{V_t}] = -\operatorname{Tr}\left\{\delta_{V_t}\left[\sum_{i=1}^{m}\sum_{\ell=1+w_3(i,ts)}^{w_4(i,ts)} (M_\ell^{i,ts})^T F_i^{-1}(T_\ell^{i,ts})^T\delta_{X_s}(N_\ell^{i,ts})^T\right]\right\}$$
$$-\operatorname{Tr}\left\{\left[\sum_{i=1}^{m}\sum_{\ell=1+w_3(i,ts)}^{w_4(i,ts)} N_\ell^{i,ts}\delta_{X_s}^T T_\ell^{i,ts} F_i^{-1} M_\ell^{i,ts}\right]\delta_{V_t}^T\right\}$$

This expression is equivalently written as

$$D\mathsf{H}_5(\delta_{X_t})[\delta_{V_t}] = \mathrm{Tr}\left\{\delta_{V_t}\mathbb{H}_5(\delta_{X_s})^T + \mathbb{H}_5(\delta_{X_s})\delta_{V_t}^T\right\} \tag{B.9}$$

with

$$\mathbb{H}_5(\delta_{X_s}) = \sum_{i=1}^{m}\sum_{\ell=1+w_3(i,ts)}^{w_4(i,ts)} N_\ell^{i,ts}\delta_{X_s}^T T_\ell^{i,ts} F_i^{-1} M_\ell^{i,ts}$$

B.2.6 The Final Term $\mathbb{H}_{ts}(\delta_{X_s})$

The final term for the Hessian map $\mathbb{H}_{ts}(\delta_{X_s})$, which is obtained from the second directional derivative of the potential function, $\mathrm{D}^2\,\Theta(\overrightarrow{X})[\delta_{X_t}, \delta_{X_s}]$, is now readily available from the expressions given in (B.5)–(B.9). Thus, we have

$$\begin{aligned} D\left(\frac{1}{2}\,\mathrm{D}^2\,\Theta(\overrightarrow{X})[\delta_{X_t}, \delta_{X_s}]\right)[\delta_{V_t}] &= \frac{1}{2}\sum_{i=1}^{5} D\mathsf{H}_i(\delta_{X_t})[\delta_{V_t}] \\ &= \mathrm{Tr}\left\{\delta_{V_t}\mathbb{H}_{ts}(\delta_{X_s})^T + \mathbb{H}_{ts}(\delta_{X_s})\delta_{V_t}^T\right\} \end{aligned}$$

with

$$\mathbb{H}_{ts}(\delta_{X_s}) = \frac{1}{2}\sum_{i=1}^{5}\mathbb{H}_i(\delta_{X_s})$$

Thus, the term $\mathbb{H}_{ts}(\delta_{X_s})$ is naturally given by

$$\begin{aligned} \mathbb{H}_{ts}(\delta_{X_s}) = {} & \frac{1}{2}\sum_{i=1}^{m}\sum_{\ell=1}^{k(i,s)}\sum_{\eta=1}^{k(i,t)}\left((A_\eta^{i,t})^T F_i^{-1} A_\ell^{i,s}\delta_{X_s} B_\ell^{i,s} F_i^{-1}(B_\eta^{i,t})^T\right) \\ & + \frac{1}{2}\sum_{i=1}^{m}\sum_{\ell=1}^{k(i,s)}\sum_{\eta=1}^{k(i,t)}\left((A_\eta^{i,t})^T F_i^{-1}(B_\ell^{i,s})^T\delta_{X_s}^T(A_\ell^{i,s})^T F_i^{-1}(B_\eta^{i,t})^T\right) \\ & - \frac{1}{2}\sum_{i=1}^{m}\sum_{\ell=1}^{w_1(i,ts)}(M_\ell^{i,ts})^T F_i^{-1}(T_\ell^{i,ts})^T\delta_{X_s}^T(N_\ell^{i,ts})^T \\ & - \frac{1}{2}\sum_{i=1}^{m}\sum_{\ell=1+w_1(i,ts)}^{w_2(i,ts)} N_\ell^{i,ts}\delta_{X_s} T_\ell^{i,ts} F_i^{-1} M_\ell^{i,ts} \\ & - \frac{1}{2}\sum_{i=1}^{m}\sum_{\ell=1+w_2(i,ts)}^{w_3(i,ts)}(M_\ell^{i,ts})^T F_i^{-1}(T_\ell^{i,ts})^T\delta_{X_s}(N_\ell^{i,ts})^T \\ & - \frac{1}{2}\sum_{i=1}^{m}\sum_{\ell=1+w_3(i,ts)}^{w_4(i,ts)} N_\ell^{i,ts}\delta_{X_s}^T T_\ell^{i,ts} F_i^{-1} M_\ell^{i,ts} \end{aligned}$$

as stated in Proposition 4.5.1.

Appendix C

Matlab Codes for the Riccati inequality

This appendix provides a copy of the Matlab codes used to solve the Riccati feasibility problem and the Riccati trace minimization problem presented in Section 4.4.4 of Chapter 4. These two codes are very simple and are mainly intended to illustrate the proposed methodology. The major code NCSDP implemented by the author to solve constrained optimization problems over matrix functions is not provided, since it would take excessively many pages.

C.1 Code: Riccati Feasibility Problem

This section provides a copy of the code used to solve the following Riccati feasibility problem:

find $\alpha^* = \min \alpha$ *subject to*

$$AX + XA^T - XRX + Q + \alpha I > 0$$

The constraint $X > 0$ is added by setting the Option(2).

```
function [X,Alpha,GammaSet] = RicFeaspCode(A,R,Q,Options)
% function [X,Alpha,GammaSet] = RicFeaspCode(A,R,Q,Options)
%
% Based on the method of centers
```

```
%
% Find Alpha and X such that
%
%     AX+XA'-X*R*X+Q + Alpha*I > 0
%
% Input:
%   A, R, Q   system matrices.
%   OPTIONS   optional:   five-entry vector of control parameters.
%             OPTIONS(1): value of centralization parameter
%                         THETA in (0,1). (Default=0.3)
%             OPTIONS(2): when nonzero, enforce the constraint X > 0
%                         (Default=0)
%             OPTIONS(3): when zero, code stops as soon as
%                         Alpha < 0.     (Default=0)
%             OPTIONS(4): Termination tolerance on objective.
%                         (Default=1e-3)
%             OPTIONS(5): Termination tolerance on the inner loop.
%                         (Default=1e-6)
%
% Output:
%   X         corresponding minimizer.
%   Alpha     value of Alpha upon termination.
%
% Author: Juan Camino
% Date  : 05/10/2003

if  nargin<3 | nargin>4 ,
  error('usage:[X,Alpha]=RicFeaspCode(A,R,Q,[options])');
elseif nargin==3,
  Options=zeros(1,5);
else
  if ~isnumeric(Options) | length(Options)~=5,
      error('OPTIONS must be a five-entry vector');
  end
end

% DEFAULTS  parameters
Ipin(1) = 0.3;       % Value of THETA
Ipin(2) = 0;         % No Constraint on X
Ipin(3) = 0;         % TARGET
```

```
Ipin(4) = 1E-3;      % stopping criteria for the objective.
Ipin(5) = 1E-6;      % stopping criteria for the inner loop.

if Options(1)~=0,  Ipin(1)=Options(1); end
if Options(2)~=0,  Ipin(2)=Options(2); end
if Options(3)~=0,  Ipin(3)=Options(3); end
if Options(4)~=0,  Ipin(4)=Options(4); end
if Options(5)~=0,  Ipin(5)=Options(5); end

n=size(Q,1);
X = eye(n);
Alpha = norm(A*X+X*A'-X*R*X+Q)+1;
Gamma= Alpha+1;
k=1;
GammaOld=0; GammaSet=[Gamma];
neg = 1;
while  neg & norm(Gamma-GammaOld)>Ipin(4) & k < 200
  GammaOld=Gamma;
  Gamma=(1-Ipin(1))*Alpha+Ipin(1)*Gamma;
  fprintf('FeaspCode: Iteration = %2i,  Gamma = %7.7f\n',k,Gamma);
  [X,Alpha,H,g] = Newton(A,R,Q,X,Alpha,Gamma,Ipin(2),Ipin(5));
  if ~Ipin(3), neg = (Gamma > 0); end
  GammaSet=[GammaSet;Gamma];
  k=k+1;
end
fclose('all');
return

function [X,Alpha,H,g] = Newton(A,R,Q,X,Alpha,Gamma,flag,tol)
% Apply Modified Newton Methods to compute the  analytic center
n=size(X);
Xset=[];Alphaset=[];
fprintf('-------------------------------------------------------\n');
str1=strcat('  k,    |g|,       T,      S,       Alpha,',...
      '            |X|,           lH,       uH\n');
str2=strcat(' %2i,  %3.1E,  %2.1E,  %2.1f,  % 7.7E,',...
      '  % 7.7E,  %2.1E,  %2.1E\n');
g = 1; Tau=1;
k = 0;
fprintf(str1);
```

```
while Tau>tol & k < 50
 [dX,dAlpha,H,g] = NCdirection(A,R,Q,X,Alpha,Gamma,flag);
 Tau = sqrt(g'/H*g);
 Sigma = ( Tau > 1/4 ) * 1/(1+Tau) + ( Tau <= 1/4  ) * 1.0;
 X=X+Sigma*dX;
 Alpha=Alpha+Sigma*dAlpha;
 Xset{k+1}=X;
 Alphaset=[Alphaset; Alpha];
 k=k+1;
 mi=min(eig(H)); ma=max(eig(H));
 fprintf(str2,k,norm(g),Tau,Sigma,Alpha,norm(X),mi,ma);
end
fprintf('-----------------------------------------------------------\n');
return

function [dX,dAlpha,H,g]=NCdirection(A,R,Q,X,Alpha,Gamma,flag)
% Return: Hessian H; Gradient g;  update directions dX and dAlpha
% A's, B's, and QQ's from  NCAlg/Mathematica
n = size(X,1);
I = eye(n);
invF1=inv(A*X+X*A'-X*R*X+Q+Alpha*I); invF1=(invF1+invF1')/2;
invF2=1/(Gamma-Alpha)*I;
A11{1}=invF1*(A-X*R);
B11{1}=invF1*(A-X*R);
A11{2}=invF1;
B11{2}=(A-X*R)'*invF1*(A-X*R); B11{2}=(B11{2}+B11{2}')/2;
A11{3}=invF1;
B11{3}=R;
A11{4}=inv(X);   % the constraint X > 0
B11{4}=inv(X);
A12{1}=invF1;
B12{1}=invF1*(A-X*R);
A21{1}=invF1*(A-X*R);
B21{1}=invF1;
A22{1}=invF1;
B22{1}=invF1;
A22{2}=invF2;
B22{2}=invF2;
% Hessian Matrix
H11 = 0;
```

```
N=3;
if flag, N=4; end
for i = 1:N;
  H11 = H11 + kron(B11{i}',A11{i})+kron(A11{i},B11{i}');
end
H21 = vec(B21{1}*A21{1}+A21{1}'*B21{1}')';
H22 = trace(A22{1}*B22{1}+A22{2}*B22{2});
H = [H11, H21'; H21, H22];
% Gradient
QQ{1}=invF1*(A-X*R) + (invF1*(A-X*R))';
if flag,
 QQ{1}=QQ{1}+inv(X);    % the constraint X > 0
 end
QQ{2}=invF1-invF2;
% Gradient vector
g = [vec(QQ{1}) ; trace(QQ{2})];
% Solves the linear system H.v = g
v = H\g;
dX = reshape(v(1:n^2),n,n);
dX = (dX+dX')/2;
dAlpha = v(n^2+1);
return

function [x] = vec(X)
     x = X(:);
return
```

C.2 Code: Riccati Trace Minimization Problem

This section provides a copy of the code used to solve the following Riccati trace minimization Problem:

find $X^* = \operatorname{argmin} \operatorname{Tr}\{X\}$, *such that*

$$AX + XA^T - XRX + Q > 0$$

The constraint $X > 0$ is added by setting Options(2).

```
function [X,GammaSet] = RicTraceCode(A,R,Q,Options)
% function [X,GammaSet] = RicTraceCode(A,R,Q,Options)
%
% Based on the method of centers
%
% Minimize Trace(X) subject to
%
%     AX+XA'-X*R*X+Q > 0
%
% Input:
%   A, R, Q   system matrices.
%   OPTIONS   optional:   five-entry vector of control parameters.
%             OPTIONS(1): value of centralization parameter
%                         THETA in (0,1). (Default=0.3)
%             OPTIONS(2): when nonzero, enforce the constraint X > 0
%                         (Default=0)
%             OPTIONS(3): not used.
%             OPTIONS(4): Termination tolerance on objective.
%                         (Default=1e-3)
%             OPTIONS(5): Termination tolerance on the inner loop.
%                         (Default=1e-6)
%
% Output:
%   X         corresponding minimizer.
%
% Author: Juan Camino
% Date  : 05/10/2003

if  nargin<3 | nargin>4 ,
```

```
  error('usage:[X]=RicTraceCode(A,R,Q,[options])');
elseif nargin==3,
  Options=zeros(1,5);
else
  if ~isnumeric(Options) | length(Options)~=5,
      error('OPTIONS must be a five-entry vector');
  end
end

% DEFAULTS  parameters
Ipin(1) = 0.3;       % Value of THETA
Ipin(2) = 0;         % No Constraint on X
Ipin(3) = 0;         % Not used
Ipin(4) = 1E-3;      % Stopping criteria for the objective.
Ipin(5) = 1E-6;      % Stopping criteria for the inner loop.

if Options(1)~=0,  Ipin(1)=Options(1); end
if Options(2)~=0,  Ipin(2)=Options(2); end
if Options(3)~=0,  Options(3)=0; end
if Options(4)~=0,  Ipin(4)=Options(4); end
if Options(5)~=0,  Ipin(5)=Options(5); end

n=size(Q,1);
% Call to the feasibility solver
disp('Feasibility Phase');
[X,Alpha] = RicFeaspCode(A,R,Q,Options);
if Alpha > 0,
  error('Problem seems to be infeasible');
end
disp('Minimization Phase');

Gamma = trace(X)+.1;
k=1;
GammaOld=0; GammaSet=[Gamma];
while  norm(Gamma-GammaOld)>Ipin(4) & k < 400
  GammaOld=Gamma;
  Gamma=(1-Ipin(1))*trace(X)+Ipin(1)*Gamma;
  fprintf('Iteration = %2i,  Gamma = %7.7f\n',k,Gamma);
  [X,H,g] = Newton(A,R,Q,X,Gamma,Ipin(2),Ipin(5));
  k=k+1;
```

```
  GammaSet=[GammaSet;Gamma];
end
fclose('all');
return

function [X,H,g] = Newton(A,R,Q,X,Gamma,flag,tol)
% Apply modified Newton's method to compute  the  analytic center
n=size(X);
Xset=[];
fprintf('-----------------------------------------------------------\n');
str1=strcat('  k,     |g|,       T,       S,     Trace(X),',...
      '        lH,         uH\n');
str2=strcat(' %2i,  %3.2E,  %2.1E,  %2.2f,  % 7.7f,',...
      '  %2.2E,  %2.2E\n');
g = 1; Tau=1;
k = 0;
fprintf(str1);
while Tau>tol & k < 50
 [dX,H,g] = NCdirection(A,R,Q,X,Gamma,flag);
 Tau = sqrt(g'/H*g);
 Sigma = ( Tau > 1/4 ) * 1/(1+Tau) + ( Tau <= 1/4  ) * 1.0;
 X=X+Sigma*dX;
 Xset{k+1}=X;
 k=k+1;
 mi=min(eig(H)); ma=max(eig(H));
 fprintf(str2,k,norm(g),Tau,Sigma,trace(X),mi,ma);
end
fprintf('-----------------------------------------------------------\n');
return

function [dX,H,g]=NCdirection(A,R,Q,X,Gamma,flag)
% Return: Hessian H; Gradient g;  update directions dX
n = size(X,1);
I = eye(n);
invF1=inv(A*X+X*A'-X*R*X+Q); invF1=(invF1+invF1')/2;
invF2=1/(Gamma-trace(X))*I;
A11{1}=invF1*(A-X*R);
B11{1}=invF1*(A-X*R);
A11{2}=invF1;
B11{2}=(A-X*R)'*invF1*(A-X*R); B11{2}=(B11{2}+B11{2}')/2;
```

```
A11{3}=invF1;
B11{3}=R;
A11{4}=inv(X);            % the constraint X > 0
B11{4}=inv(X);
% Hessian Matrix
H11 = 0;
N=3; if flag,  N=4; end
for i = 1:N;
  H11 = H11 + kron(B11{i}',A11{i})+kron(A11{i},B11{i}');
end
% Add the Cost Function
H = H11 + 1/(Gamma-trace(X))^2*vec(I)*vec(I)';
% Gradient
QQ{1}=invF1*(A-X*R) + (invF1*(A-X*R))' - invF2;
if flag,
  QQ{1}=QQ{1}+inv(X);    % the constraint X > 0
end
% Gradient vector
g = vec(QQ{1});
% Solves the linear system H.v = g
v = H\g;
dX = reshape(v(1:n^2),n,n);
dX = (dX+dX')/2;
return

function [x] = vec(X)
     x = X(:);
return
```

Appendix D

Collection of Experiments Used to Check the Code

This chapter presents a collection of minimization problems used to check the implementation of the proposed methodology. Most of the problems here presented are convex and convertible to LMI. So that, their solutions were easily verified by others SDP codes.

D.1 List of Successful Convex Experiments

The NCSDP solver run successfully all the convex experiments we have tried, for a variety of different dimensions. The set of feasibility problems used to check the code basically has the form:

$$\min \alpha \qquad \text{subject to}$$
$$F(X_1, \ldots, X_r) + \alpha I > 0, \qquad G_i(X_1, \ldots, X_r) > 0$$

We have also run an inner product version of all the feasibilities problems presented. In this test, the constraints remained the same, but the cost function was the trace of one of the unknowns. Thus, we do not repeat them here.

D.1.1 Symmetric variables

Example D.1.1

$$\min_X \alpha \qquad \text{subject to}$$
$$\alpha I - B^T X B > 0$$
$$AX + XA^T - R + XQ_1X < 0$$
$$X > Q_2$$

Example D.1.2

$$\min_{X,T} \alpha \qquad \text{subject to}$$
$$\alpha I - T > 0$$
$$AX + XA^T + Q_1 - XRX + T > 0$$

Example D.1.3

$$\min_{X,Y} \alpha \qquad \text{subject to}$$
$$\alpha I - (B_1^T X B_1 + B_2^T Y B_2) > 0$$
$$AX + X^T A^T - R + XQX + AY + YA^T + YQY < 0$$

Example D.1.4

$$\min_{X,Y,Z,K} \alpha \qquad \text{subject to}$$
$$\alpha I - (X + Y + Z + K) > 0$$
$$X - Y^{-1} > 0$$
$$Y - ZX^{-1}Z > 0$$
$$X + Z - Y - K > 0$$
$$X > 0, \qquad Y > 0, \qquad Z > 0, \qquad K > 0$$
$$X < Q, \qquad Z < Q, \qquad Y < Q$$

Example D.1.5

$$\min_{X_1,X_2,X_3,X_4} \alpha \qquad \text{subject to}$$
$$\alpha I - (X_1 + X_2 + X_3 + X_4) > 0$$
$$-X_1^2 + Q > 0$$
$$-X_2^2 + Q > 0$$
$$-X_3^2 + Q > 0$$
$$-X_4^2 + Q > 0$$

Example D.1.6

$$\min_{X,T} \alpha \qquad \text{subject to}$$
$$\alpha I - T > 0$$
$$Q - X^2 - X^{-1} + T > 0$$
$$X > 0, \qquad Q - X > 0$$

D.1.2 Not symmetric variables

Example D.1.7

$$\min_{X,F,U} \alpha \qquad \text{subject to}$$
$$\alpha I - U > 0$$
$$AX + XA^T + BF + F^T B + F^T RF + Q < 0$$
$$U - FX^{-1}F > 0$$
$$X > 0, \qquad CXC^T < \Omega$$

Example D.1.8

$$\min_{X,Y,Z} \alpha \qquad \text{subject to}$$
$$\alpha I - (X + Y + B^T Z^T + ZB) > 0$$
$$X - Y^{-1} > 0$$
$$Y - B^T Z^T X^{-1} ZB > 0$$
$$ZB + B^T Z^T - X > 0$$
$$I - ZZ^T > 0$$

D.2 Nonconvex Experiments

In this thesis, we have focused on convex problems solely, since we have made no effort in providing a reliable implementation of a nonconvex code based on the proposed methodology. However, with a simple modification of our convex code, basically by implementing a rudimentary line search and a "strategy" for dealing with indefinite Hessian, we were able to successfully run a few nonconvex examples.

D.3 List of "Successful" Nonconvex Experiments

We have tested the solver for a variety of matrix sizes. The solver was able to reached an optimal solution within an accuracy of at least 10^{-4} on the objective value, for most of the dimensions and initial conditions. In this sense we say the solver runs successfully. However, for some initial conditions, the solver fails to attain a solution. Basically, in the ongoing implementation, the nonconvex code stops whenever the algorithm is not able to compute a feasible direction.

D.3.1 Symmetric variables

Example D.3.1 Let $A \in \mathbb{R}^{n\times n}$, $X, Y, M, N \in \mathbb{S}^n$, $R \in \mathbb{S}^n_{++}$, and $Q_i \in \mathbb{S}^n$ for $i = 1, \ldots, 4$. Let us define the following Riccati in X

$$\mathrm{Ric}(X) := AX + XA^T - XR^{-1}X + M$$

and the following polynomial in X

$$\mathrm{Poly}(X) := (X - Q_1)(X - Q_2)(X - Q_3)(X - Q_4) + N$$

The optimization problem is

$$\min_{X,Y} \alpha \qquad \text{subject to}$$

$$\alpha I > X, \qquad Y < \mathrm{Ric}(X), \qquad \text{and} \qquad Y > \mathrm{Poly}(X)$$

First experiment

The data are taken to be scalars: $Q_1 = 1$, $Q_2 = 4$, $Q_3 = 10$, $Q_4 = 15$, $N = 100$, $A = 120$, $R = 0.0625$, $M = 100$. The initial feasible guess is $X = 12.5$ and $Y = 0$. The feasibility

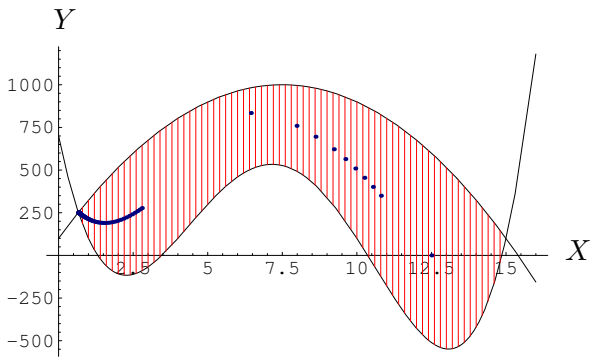

Figure D.1: Feasibility region and solution path

region together with the solution path (dotted line) is shown in Figure D.1. The optimal point was found to be $X^* = 0.6609$ and $Y^* = 251.6233$.

Second experiment

Now, we change the values for $A = 100$ and $M = -100$. All the other values remains the same. The result is presented in Figure D.2. For this experiment, if we take the initial guess to be $X = 4.5$ and $Y = 400$, the optimal solution is $X^* = 1.0296$ and $Y^* = 88.9659$. On the other hand, if we take the initial guess to be $X = 11$ and $Y = 0$, the optimal solution is $X^* = 8.2965$ and $Y^* = 457.987$.

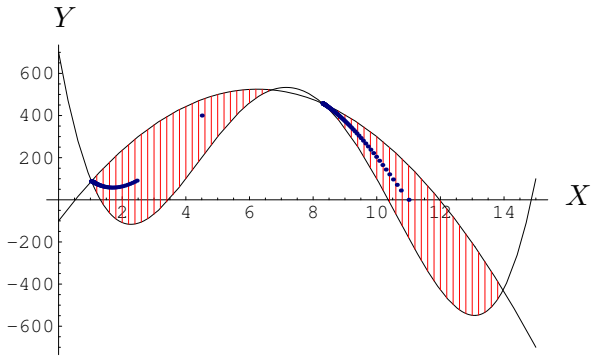

Figure D.2: Feasibility region and solution path

Third experiment

We also run the above nonconvex optimization problem for a variety of matrices of different size with the data generated randomly. Whenever we had a initial feasible guess, the code

run successfully. For example, let $n = 3$ and the date be given by

$$Q_1 = \begin{pmatrix} 0.1600 & 0.1000 & 0.1500 \\ 0.1000 & 0.2200 & 0.2200 \\ 0.1500 & 0.2200 & 0.2400 \end{pmatrix}, \qquad M = \begin{pmatrix} 2800 & 5100 & 7900 \\ 5100 & 16000 & 19000 \\ 7900 & 19000 & 27000 \end{pmatrix}$$

with $Q_2 = Q_3 = Q_4 = Q_1$ and $N = -M$, and

$$A = \begin{pmatrix} 0.3600 & 0.0900 & 0.4500 \\ 0.3900 & 0.4900 & 0.0340 \\ 0.7500 & 0.6900 & 0.7200 \end{pmatrix}, \qquad R = \begin{pmatrix} 0.8300 & 0.6500 & 0.0940 \\ 0.6500 & 0.9300 & 0.0300 \\ 0.0940 & 0.0300 & 0.0160 \end{pmatrix}.$$

The initial guess is $Y = 0$ and X given by

$$X = \begin{pmatrix} 1.2000 & 0.2500 & 0.3700 \\ 0.2500 & 1.5000 & 0.8300 \\ 0.3700 & 0.8300 & 1.7000 \end{pmatrix}.$$

For these values, the optimal solution, found after 85 iterations, is given by

$$X^* = \begin{pmatrix} -4.3306 & -1.2405 & -2.2693 \\ -1.2405 & -4.5240 & -1.8789 \\ -2.2693 & -1.8789 & -5.1755 \end{pmatrix} \quad \text{and} \quad Y^* = \begin{pmatrix} -0.0571 & -0.1135 & -0.3483 \\ -0.1135 & -0.1808 & -0.5147 \\ -0.3483 & -0.5147 & -1.3106 \end{pmatrix}$$

The maximum eigenvalue of X^* is $\alpha^* = -2.3876$.

Example D.3.2 All the dimensions we have tried for this problem provided the optimal solution $X^* = I$.

$$\min_{X=X^T} \alpha \qquad \text{subject to}$$
$$\alpha I - X > 0$$
$$X^2 - I > 0$$
$$X - 0.5I > 0$$

Example D.3.3 Powell's Function, Powell (1964). In the scalar case, the typical testing point is $(X_1, X_2, X_3, X_4) = (3, -1, 0, 1)$. The unknowns $X_i \in \mathbb{S}^n$ for $i = 1, \ldots, 4$. The optimal solution is $X_i^* = 0$, for $i = 1, \ldots, 4$.

$$\min_{X_1, X_2, X_3, X_4} \alpha \qquad \text{subject to}$$
$$\alpha I - ((X_1 + 10X_2)^2 + 5(X_3 - X_4)^2 + (X_2 - 2X_3)^4 + (10X_1 - X_4)^4) > 0$$

We have test the solver for the dimensions $n = 1, \ldots, 7$. For this dimension the solver reached the optimal solution $X^* = 0$ with the initial guesses, which were randomly generated, as (Matlab notation):

$$X1 = \text{randn}(n); X1 = X1 + X1';$$
$$X2 = \text{randn}(n); X2 = X2 + X2';$$
$$X3 = \text{randn}(n); X3 = X3 + X3';$$
$$X4 = \text{randn}(n); X4 = X4 + X4';$$

For a few initial condition, the solver failed.

Example D.3.4 Rosenbrock's Function - Banana valley, Rosenbrock (1960).

$$\min_{X_1, X_2} \alpha \qquad \text{subject to}$$
$$\alpha I - (100(X_2 - X_1^2)^2 + (I - X_1)^2) > 0$$

For the scalar case, the typical testing point is $(X_1, X_2) = (-1.2, 1)$. The solver was successful for the scalar case, and we also test it for the range of dimensions $n = 1, \ldots, 9$ with the initial guess been randomly generated as (Matlab notation):

$$X = 2 * \text{randn}(n); X = (X * X')/n;$$
$$Y = 2 * \text{randn}(n); Y = (Y * Y')/n;$$

All the test provided the optimal solution $X^* = I$.

D.3.2 Not symmetric variables

Example D.3.5 We generalize the Rosenbrock's Function such that $Y \in \mathbb{S}^n$, $X \in \mathbb{R}^{p \times q}$, $A \in \mathbb{R}^{n \times p}$, and $B \in \mathbb{R}^{q \times n}$.

$$\min_{X, Y} \alpha \qquad \text{subject to}$$
$$\alpha I - (100(Y - B^T X^T X B)^2 + (I - B^T X^T A^T)(I - AXB)) > 0$$

The solver converged to a solution for a large variety of dimensions (for matrices of size less than 10×10), and initial guesses generated randomly. Although for some initial guess, the solver failed.

Example D.3.6 For this example X and T are square matrices. The solver reached the optimal values for many different size and initial guess. It also failed for some initial

conditions.

$$\min_{X,T} \alpha \qquad \text{subject to}$$

$$\alpha I - T > 0$$

$$2I - X^3 - X^{-3} + T > 0$$

$$X > 0, \qquad 2I - X > 0$$

When successful the optimal solution $X^* = I$ and $T^* = 0$ is found.

Bibliography

Alizadeh, F., Haeberly, J.-P. A., and Overton, M. L. (1998). Primal-dual interior-point methods for semidefinite programming: convergence rates, stability and numerical results. *SIAM J. Optim.*, 8(3):746–768.

Boyd, S. and El Ghaoui, L. (1993). Method of centers for minimizing generalized eigenvalues. *Linear Algebra and its Applications, special issue on Numerical Linear Algebra Methods in Control, Signals and Systems*, 188:63–111.

Boyd, S., El Ghaoui, L., Feron, E., and Balakrishnan, V. (1994). *Linear Matrix Inequalities in Systems and Control Theory*, volume 15 of *Studies in Applied Mathematics*. SIAM, Philadelphia, PA, USA.

Boyd, S. and Vandenberghe, L. (2003). Convex optimization. To be published by Cambridge University Press.

Camino, J. F., Zampieri, D. E., and Peres, P. L. D. (1999). Design of A vehicular suspension controller by static output feedback. In *Proc. American Control Conf.*, pages 3168–3172, San Diego, CA.

Canon, M., Cullum, C., and Polak, E. (1966). Constrained minimization problems in finite-dimensional spaces. *SIAM J. Control.*, 4(3):528–547.

Chalasani, R. M. (1987). Ride performance of active suspension systems - Part I: Simplified analysis based on a quarter-car model. *ASME, AMD*, 80:187–204.

Colaneri, P., Geromel, J. C., and Locatelli, A. (1997). *Control Theory and Design*. Academic Press, San Diego, CA, USA.

de Oliveira, M. C., Camino, J. F., and Skelton, R. E. (2000). A convexifying algorithm for the design of structured linear controller. In *Proc. IEEE Conf. on Decision and Control*, pages 2781–2786, Sydney, Australia.

El-Ghaoui, L. and Niculescu, S. (1999). *Advances in Linear Matrix Inequality Methods in Control*. Advances in Design and Control. SIAM, Philadelphia, PA, USA.

Fiacco, A. V. and McCormick, G. P. (1990). *Nonlinear Programming: Sequential Unconstrained Minimization Techniques*. Classics in Applied Mathematics. SIAM, Philadelphia, PA, USA.

Fröberg, R. (1997). *An Introduction to Gröbner Bases*. Pure and Applied Mathematics. John Wiley & Sons, Inc., Chichester, UK, and New York, USA.

Fujisawa, K., Fukuda, M., Kojima, M., and Nakata, K. (1997). Numerical Evaluation of SDPA (Semidefinite Programming Algorithm). Technical Report B-330, Department of Mathematical and Computing Sciences, Tokyo Institute of Technology, Oh-Okayama, Meguro-ku, Tokyo 152.

Gahinet, P., Nemirovskii, A., Laub, A. J., and Chilali, M. (1995). *LMI Control Toolbox*. The Math Works, Inc., USA.

Gill, P. E., Murray, W., and Wright, M. H. (1999). *Practical Optimization*. Academic Press, San Diego, CA, USA.

Golub, G. and Loan, C. V. (1983). *Matrix Computation*. The Johns Hopkins University Press, Baltimore, USA, and London, UK.

Grandhi, R. V. (1989). Structural and control optimization of space structures. *Computers & Structures*, 31:139–150.

Graves, L. M. (1935). Topics in the functional calculus. *Bulletin of the American Mathematical Society*, 41:641–662.

Grigoriadis, K. M. and Skelton, R. E. (1998). Integrated structural and control design for vector second-order systems via LMIs. In *Proc. American Control Conf.*, pages 1625–1629, Philadelphia, Pennsylvania.

Grigoriadis, K. M. and Wu, F. (1997). Integrated H_∞ plant/controller design via linear matrix inequalities. In *Proc. IEEE Conf. on Decision and Control*, pages 789–790, San Diego, California.

Grigoriadis, K. M., Zhu, G., and Skelton, R. E. (1996). Optimal redesign of linear systems. *ASME Journal of Dynamic Systems, Measurement, and Control*, 118:598–605.

Helton, J. W. (2002). Manipulating matrix inequalities automatically. In *Plenary Talks at the MTNS 2002*, volume 134 of *Inst. Math Analysis, Series on Math and Appl.*, pages 237–257. Springer Verlag.

Helton, J. W. and Merino, O. (1997). Coordinate optimization for bi-convex matrix inequalities. In *Proc. IEEE Conf. on Decision and Control*, volume 4, pages 3609–3613.

Helton, J. W. and Merino, O. (1998). Sufficient conditions for optimization of matrix functions. In *Proc. IEEE Conf. on Decision and Control*, volume 3, pages 3361–3365.

Helton, J. W., Stankus, M., and Wavrik, J. J. (1998). Computer simplification of formulas in linear system theory. *IEEE Trans. Aut. Control*, 4(3):302–314.

Hildebrandt, T. H. and Graves, L. M. (1927). Implicit functions and their differentials in general analysis. *Transaction of the American Mathematical Society*, 29:127–153.

Horn, R. A. and Johnson, C. R. (1996). *Matrix Analysis*. Cambridge University Press, New York, USA.

Horn, R. A. and Johnson, C. R. (1999). *Topics in Matrix Analysis*. Cambridge University Press, New York, USA.

Housner, G. W., Bergman, L. A., Caughey, T. K., Chassiakos, A. G., Claus, R. O., Masri, S. F., Skelton, R. E., Soong, T. T., Spencer, B. F., and Yao, J. T. P. (1997). Structural control: Past, present and future. *ACSE Journal of Engineering Mechanics*, 123(9):897–971.

Hrovat, D. (1991). Optimal suspension performance for 2-D vehicle models. *Journal of Sound and Vibration*, 146(1):93–110.

Hrovat, D. (1993). Applications of optimal control to advanced automotive suspension design. *ASME Journal of Dynamic Systems, Measurement, and Control*, 115(2B):328–342.

Hsieh, C. (1992). Robust output variances constrained controller design. In *Proc. American Control Conf.*, pages 2871–2875, Chicago, IL.

Huard, P. (1967). Resolution of mathematical programming with nonlinear constraints by the method of centers. In Abadie, J., editor, *Nonlinear Programming*, chapter 8, pages 209–219. John Wiley & Sons, Inc., Chichester, UK, and New York, USA.

Iwasaki, T. and Skelton, R. E. (1994). All controlled for the general $\mathcal{H}_\infty$ control problem: LMI existence conditions and state space formulas. *Automatica*, 30(8):1307–1317.

Jarre, F. (2000). A QQP-minimization methods for semidefinite and smooth nonconvex programs. Technical report, University of Notre Dame, Notre Dame, IN.

Jin, I. M. and Sepulveda, A. E. (1995). Structural/control system optimization with variable actuator masses. *AIAA Journal*, 33(9):1709–1714.

Konstantinov, M., Mehrmann, V., and Petkov, P. (2000). On properties of Sylvester and Lyapunov operators. *Linear Algebra and its Applications*, 312:35–71.

Kose, I. E., Jabbari, F., Schmitendorf, W. E., and Yang, J. N. (1998). Controllers for quadratic stability and performance of a benchmark problems. *Earthquake Engineering and Structural Dynamics*, 27(11):1385–1397.

Kreyszig, E. (1989). *Introductory Functional Analysis with Application*. John Wiley & Sons, Inc., Chichester, UK, and New York, USA.

Lancaster, P. and Rodman, L. (1995). *Algebraic Riccati Equations*. Oxford University Press.

Leibfritz, F. and Mostafa, E. M. (2002). An interior point constrained trust region method for a special class of nonlinear semidefinite programming problems. *SIAM J. Optim.*, 12(4):1048–1074.

Lieu, B. and Huard, P. (1965). La méthode des centres dans un espace topologique. *Numerische Mathematik*, 8:56–67.

Lu, J. and Skelton, R. E. (2000). Integrating structure and control design to achieve mixed H_2/H_∞ performance. *Int. J. Control*, 73(16):1449–1462.

Luenberger, D. G. (1969). *Optimization by Vector Space Methods*. John Wiley & Sons, Inc., Chichester, UK, and New York, USA.

Lusternik, L. A. and Sobolev, V. J. (1961). *Elements of Functional Analysis*. Gordon & Breach Publishing Group, New York.

MacLane, S. and Birkhoff, G. (1999). *Algebra*. AMS Chelsea Publishing.

Mangasarian, O. L. (1994). *Nonlinear Programming*. Classics in Applied Mathematics. SIAM, Philadelphia, PA, USA.

Maurer, H. and Zowe, J. (1979). First and second-order necessary and sufficient optimality conditions for infinite-dimensional programming problems. *Mathematical Programming*, 16:98–110.

Michel, A. N. and Herget, C. J. (1981). *Applied Algebra and Functional Analysis*. Dover.

Mora, T. (1986). Gröebner bases for noncommutative polynomial rings. *Lecture Notes in Computer Sci.*, 1(229):353–362.

Mora, T. (1994). An introduction to commutative and noncommutative Gröbner bases. *Theoret. Comput. Sci.*, 134(1):131–173.

Murray, W. (1971). Analytic expression for the eigenvalues and eigenvectors of the Hessian matrices of barrier and penalty functions. *Journal of Optimization Theory and Applications*, 7:189–196.

Nesterov, Y. and Nemirovskii, A. (1994). *Interior-Point Polynomial Algorithms in Convex Programming*, volume 13 of *Studies in Applied Mathematics*. SIAM, Philadelphia, PA, USA.

Onoda, J. and Haftka, R. T. (1987). An approach to structure/control simultaneous optimization for large flexible spacecraft. *AIAA Journal*, 25(8):1133–1139.

Ortega, J. M. and Rheinboldt, W. C. (2000). *Iterative Solution of Nonlinear Equations in Several Variables*. Classics in Applied Mathematics. SIAM, Philadelphia, PA, USA.

Overton, M. L. (1988). On minimizing the maximum eigenvalue of a symmetric matrix. *SIAM J. Matrix Anal. Appl.*, 9(2):256–268.

Parrilo, P. A. (2000). On A decomposition of multivariable forms via LMI methods. In *Proc. American Control Conf.*

Powell, M. J. D. (1964). An efficient method for finding the minimum of a function of several variables without calculating derivatives. *The Computer Journal*, 7:155–162.

Powers, V. and Wörmann, T. (1998). An algorithm for sums of squares of real polynomials. *Journal of Pure and Applied Algebra*, 127:99–104.

Ramalho, J. C., Johnson, E. A., and Spencer, B. F. (2000). "Smart" base isolation system. *ACSE Journal of Engineering Mechanics*, 128(10):1088–1099.

Reed, M. and Simon, B. (2000). *Methods of Modern Mathematical Physics I: Functional Analysis*. Academic Press.

Renegar, J. (2001). *A Mathematical View of Interior–Point Methods in Convex Optimization*. MPS-SIAM Series on Optimization. SIAM, Philadelphia, PA, USA.

Rockafellar, R. T. (1997). *Convex Analysis*. Princeton University Press, Princeton, N. J., USA.

Rosenbrock, H. H. (1960). An automatic method for finding the greatest or the least value of a function. *The Computer Journal*, 3:175–184.

Roubi, A. (2001). Some properties of methods of centers. *Computational Optimization and Applications*, 19:319–335.

Sharp, R. S. and Crolla, D. A. (1987). Road vehicle suspension system design - A review. *Vehicle System Dynamics*, 16:167–192.

Skelton, R. E., Hanks, B. R., and Smith, M. (1992). Structure redesign for improved dynamic response. *Journal of Guidance Control and Dynamics*, 15(5):1272–1278.

Skelton, R. E. and Iwasaki, T. (1995). Increased roles of linear algebra in control education. *IEEE Control Syst. Mag.*, 15(4):76–90.

Skelton, R. E., Iwasaki, T., and Grigoriadis, K. M. (1998). *A Unified Algebraic Approach to Linear Control Design*. Taylor & Francis, London.

Skelton, R. E. and Kim, J. H. (1992). The optimal mix of structure redesign and active dynamic controller. In *Proc. American Control Conf.*, pages 2775–2779.

Spencer Jr., B. F., Dyke, S. J., and Deoskar, H. S. (1998). Benchmark problems in structural control: Part I – active mass driver system. *Earthquake Engineering and Structural Dynamics*, 27(11):1127–1139.

Sturm, J. F. (1999). Using SeDuMi 1.02, a MATLAB toolbox for optimization over symmetric cones. *Optimization Methods and Software*, 11/12(1-4):625–653.

Takahashi, R. H. C., Camino, J. F., Zampieri, D. E., and Peres, P. L. D. (2000). Multiobjective Weighting Selection for Optimization-Based Control Design. *ASME Journal of Dynamic Systems, Measurement, and Control*, 122(3):567–570.

Thompson, A. G. (1976). An active suspension with optimal linear state feedback. *Vehicle System Dynamics*, 5:187–203.

Vandenberghe, L. and Balakrishnan, A. V. (1997). Algorithms and software for LMI problems in control. *IEEE Control Syst. Mag.*, pages 89–95.

Vandenberghe, L. and Boyd, S. (1995). A primal-dual potential reduction method for problems involving matrix inequalities. *Mathematical Programming, Series B*, 69:205–236.

Vandenberghe, L. and Boyd, S. (1996). Semidefinite Programming. *SIAM Review*, 38:49–95.

Wright, M. H. (1992a). Determining subspace information from the hessian of a barrier function. Technical Report NAM 92-02, AT&T Bell Laboratories.

Wright, M. H. (1992b). Interior methods for constrained optimization. *Acta Numerica*, pages 341–407.

Wright, S. J. (1997). *Primal-Dual Interior-Point Methods*. SIAM, Philadelphia, PA, USA.

Wright, S. J. and Orban, D. (2001). Properties of the log-barrier function on degenerate nonlinear programs. Technical Report TR/PA/99/36, CERFACS.

Yang, Y.-P. and Chen, Y.-A. (1996). Multiobjective optimization of hard disk suspension assemblies: Part II — integrated structure and control design. *Computers & Structures*, 59(4):771–782.

Zhou, K., Doyle, J. C., and Glover, K. (1996). *Robust and Optimal Control*. Prentice-Hall, Inc., New Jersey, USA.