University of Liège
Faculty of applied sciences
Department of electrical engineering and computer science

Geometric optimization algorithms for linear regression on fixed-rank matrices

PhD thesis of Gilles Meyer

Advisor: Prof. Rodolphe Sepulchre
Co-advisor: Dr. Silvère Bonnabel

June 2011

Jury members:

L. Wehenkel	(President)	Professor, University of Liège
R. Sepulchre	(Advisor)	Professor, University of Liège
S. Bonnabel	(Co-advisor)	Maître-assistant, Mines ParisTech
P. Geurts		FNRS research associate, University of Liège
P.-A. Absil		Associate professor, Université Catholique de Louvain
F. Bach		INRIA researcher, Ecole Normale Supérieure, Paris
I. Dhillon		Professor, University of Texas, Austin

Abstract

Nowadays, large and rapidly evolving data sets are commonly encountered in many modern applications. Efficiently mining and exploiting these data sets generally results in the extraction of valuable information and therefore appears as an important challenge in various domains including network security, computer vision, internet search engines, bioinformatics, marketing systems, online advertisement, social networks, just to name a few.

The rapid development of these modern computer science applications sustains an ever-increasing demand for efficient machine learning algorithms that can cope with large-scale problems, characterized by a large number of samples and a large number of variables.

The research reported in the present thesis is devoted to the design of efficient machine learning algorithms for large-scale problems. Specifically, we adopt a geometric optimization viewpoint to address the problem of linear regression in nonlinear and high-dimensional matrix search spaces. Our purpose is to efficiently exploit the geometric structure of the search space in the design of scalable linear regression algorithms.

Our search space of main interest will be the set of low-rank matrices. Learning a low-rank matrix is a typical approach to cope with high-dimensional problems. The low-rank constraint is expected to force the learning algorithm to capture a limited number of dominant factors that mostly influence the sought solution. We consider both the learning of a fixed-rank symmetric positive semidefinite matrix and of a fixed-rank non-symmetric matrix.

A first contribution of the thesis is to show that many modern machine learning problems can be formulated as linear regression problems on the set of fixed-rank matrices. For example, the learning of a low-rank distance, low-rank matrix completion and the learning on data pairs are cast into the considered linear regression framework. For these problems, the low-rank constraint is either part of the original problem formulation or is a sound approximation that significantly reduces the original problem size and complexity, resulting in a dramatic decrease in the computational complexity of algorithms.

Our main contribution is the development of novel efficient algorithms for learning a linear regression model parameterized by a fixed-rank matrix. The resulting algorithms preserve the underlying geometric structure of the problem, scale to high-dimensional problems, enjoy local convergence properties and confer a geometric basis to recent contributions on learning fixed-rank matrices. We thereby show that the considered geometric optimization framework offers a solid and versatile framework for the design of rank-constrained machine learning algorithms.

The efficiency of the proposed algorithms is illustrated on several machine learning applications. Numerical experiments suggest that the proposed algorithms compete favorably with the state-of-the-art in terms of achieved performance and required computational time.

Résumé

A l'heure actuelle, des bases de données de grande taille et soumises à des modifications régulières sont rencontrées dans la plupart des applications modernes. Une exploitation efficace de ces données constitue un enjeu essentiel dans de nombreux domaines d'activités tels que la sécurité des réseaux, la vision par ordinateur, les moteurs de recherche internet, la bioinformatique, les systèmes de marketing, la publicité en ligne ou encore les réseaux sociaux.

Le développement rapide de ces applications entretient une demande sans cesse croissante pour des algorithmes d'apprentissage automatique efficaces, capables de traiter des problèmes de grande taille en termes du nombre d'échantillons et du nombre de variables.

La recherche présentée dans cette thèse de doctorat est dédiée à l'élaboration d'algorithmes d'apprentissage automatique capables de traiter efficacement des problèmes de grande taille. Plus spécifiquement, nous adoptons un cadre géométrique d'optimisation pour aborder le problème de la regression linéaire dans des espaces de recherche non linéaires et de grande dimension. L'approche proposée repose sur une exploitation adéquate de la géométrie du problème pour élaborer de nouveaux algorithmes d'optimisation efficaces appliqués à la régression linéaire.

L'espace de recherche sur lequel porte principalement notre intérêt est l'ensemble des matrices réelles de rang faible et fixé. L'apprentissage d'une matrice de rang faible est une approche typique pour aborder des problèmes de grande dimension. La contrainte de rang faible tend à forcer l'algorithme d'apprentissage à capturer un nombre limité de facteurs dominants qui influences la solution recherchée. Nous abordons l'apprentissage des matrices réelles symétriques semidefinies positives de rang fixé puis nous généralisons les résultats obtenus aux matrices réelles non symétriques de rang fixé.

Une première contribution de cette thèse de doctorat est de montrer que de nombreux problèmes récents en apprentissage automatique peuvent être formulés comme un problème de régression linéaire sur l'espace des matrices de rang fixé. Par exemple, l'apprentissage d'une mesure de distance, la complétion de matrices, l'apprentissage sur des paires de données sont des problèmes qui se formulent naturellement dans le cadre de travail considéré. Pour ces problèmes, la contrainte de rang fixé apparaît naturellement, soit parce qu'elle fait partie de la formulation originale du problème, soit parce qu'elle permet une approximation raisonable de la solution, tout en réduisant significativement la taille et la complexité du problème original et la complexité des algorithmes.

Notre contribution principale est la conception de nouveaux algorithmes efficaces pour apprendre un modèle de régression linaire paramétré par une matrice de rang fixé. Les algorithmes résultants préservent la géométrie du problème, sont applicables à des problèmes de grande dimension, présentent des propriétés de convergence locales et confèrent une base géométrique à des contributions récentes sur l'apprentissage des matrices de rang fixé. Nous montrons de ce fait que les techniques d'optimisation considérées offrent un cadre de travail solide et flexible pour la conception de nouveaux algorithmes pour apprendre une matrices de rang fixé.

L'efficacité des algorithmes proposés est illustrée sur plusieurs problèmes d'aprentissage automatique, dont l'apprentissage d'une distance, la complétion de matrices et l'apprentissage sur des paires de données. Les simulations numériques effectuées suggèrent que les algorithmes proposés rivalisent favorablement avec l'état de l'art, à la fois en terme des performances obtenues et du temps de calcul requis.

Acknowledgements

Completing this thesis has been a rich, positive and often overwhelming personal experience. However, the work that I have carried out during the last four years would have been impossible without the help and support from many people to whom I would like to express my sincere appreciation.

First and foremost, my hearty thanks go to my advisor, Rodolphe Sepulchre, for his ever pertinent guidance and unconditionally enthusiastic support. I am very grateful to Rodolphe for his precious advice and for sharing with me his expertise and enthusiasm. Rodolphe has been for me a marvelous consultant and an outstanding project manager; who always succeeded in providing me optimism and motivation, especially when I needed it the most.

I warmly thank my closest collaborators Silvère Bonnabel (co-advisor of this thesis), Michel Journée and Bamdev Mishra who directly contributed to the present work. They have been of great help both scientifically and personally. Special thanks go to Silvère and Michel for coaching me with patience and talent during the beginning of my thesis.

I also owe warm thanks to Pierre Geurts and Alain Sarlette for inspiring research discussions and precious advice on subjects related to the present work.

These four years spent at Montefiore Institute have been extremely pleasant for many reasons. One of the most important reasons is great friends and colleagues from the research unit, past and present. I enjoyed very much working and exchanging with them during the last four years. Special thanks go to the "Coffee break" group for entertaining breaks and lunches.

I thank the administrative staff of the Montefiore Institute, and in particular Charline for her constant kindness and precious help.

I would also like to express my genuine appreciation to all members of the jury, Pierre-Antoine Absil, Francis Bach, Silvère Bonnabel, Inderjit Dhillon, Pierre Geurts, Rodolphe Sepulchre and Louis Wehenkel for devoting time and interest to the reading and evaluation of this manuscript.

I gratefully acknowledge the financial support from the Belgian National Fund for Scientific Research (FNRS) as well as from the Belgian Network DYSCO (Dynamical Systems, Control, and Optimization), funded by the Interuniversity Attraction Poles Programme, initiated by the Belgian State, Science Policy Office.

Last but not least, I want to express my deepest gratitude to my family and friends for their constant support and for helping me disconnect from research when needed. Super special thanks to my wonderful and lovely wife Estelle, who has been of invaluable patience and support.

Contents

Chapter 1

Introduction

With the advent of recent information technologies, data are collected, updated and stored at an ever-increasing rate. For instance,[1] the number of indexed web pages in the Google search engine is around 40 billions in June 2011. The online video broadcast system Youtube sees several hours of video uploaded on its website every minute. The social network Facebook counts more than 500 million active users in June 2011 and more than 30 billion pieces of content (web links, news stories, blog posts, notes, photo albums, etc.) are shared by the users each month. The movie rental system Netflix counts more than 23 million members in the United States and Canada in early 2011. Netflix members rate about four million movies a day. In bioinformatics and genomics also, high-throughput experiments on genetic material allow biologists to collect a large amount of high-dimensional data. As the cost of biological data acquisition becomes cheaper, the number of collected samples is expected to grow rapidly in the near future.

Efficiently mining large and evolving data sets has become an important challenge for many modern computer science applications that arise from various domains including network security, internet search engines, bioinformatics, computer vision, marketing systems, online advertisement, social networks, just to name a few. Over the last few years, a significant amount of research efforts have therefore been devoted to the design of efficient data mining algorithms whose aim is to extract relevant and up to date information from large data sets.

A central problem in this context is the learning of effective predictive models. A particular example of a valuable predictive model for an e-commerce company is a model that predicts the next items that are likely to be purchased by a given customer. This model can be exploited to suggest items to customers in accordance with their preferences. Another example is a predictive model of distance between any two objects of a data set. The computed distance can then be used for automatic similarity-based annotation or labeling. Another example, in bioinformatics, is a predictive model of pairwise interactions between entities of a biological network.

Although these problems might appear different in nature, they all fall in the scope of machine learning algorithms and can be solved by a particular instance of linear regression. This is not so surprising as linear regression is a fundamental building block in the design and computation of predictive models. In its basic form, the linear regression problem amounts to finding a set of linear predictors (or coefficients) that explain as much as possible the dependence between observations and input data. Linear regression admits many variants and refinements that are described by most introductory textbooks (see Hastie et al., 2009, and references therein).

In the present thesis, we address a variant of the linear regression problem for which the parameter of the linear regression model lies on a nonlinear and high-dimensional matrix space.

Our focus is on the learning of a linear regression model parameterized by a fixed-rank matrix. We consider the learning of a fixed-rank symmetric positive semidefinite matrix and the learning of a fixed-rank non-symmetric matrix. In our formulation, the rank of the matrix is fixed a priori and is typically chosen very small compared to the original problem dimension.

[1] Statistics are collected from the online press releases of the corresponding companies.

Learning a low-rank matrix is a typical approach to reduce the dimensionality of problems involving a large number of interrelated variables. The low-rank assumption is expected to force the learning algorithm to capture the dominant factors that mostly influence the sought solution.

A contribution of this thesis is to reformulate a number of modern large-scale machine learning applications as a linear regression problem on the set of fixed-rank matrices. For example, the learning of a low-rank distance is cast as a linear regression problem on the set of fixed-rank symmetric positive semidefinite matrices. The low-rank matrix completion problem is cast as a linear regression problem on the set of fixed-rank non-symmetric matrices.

The underlying regression problem is turned into an optimization problem which amounts to finding the optimal model parameter $\mathbf{W}^\star$ that minimizes an expected prediction error,

$$\mathbf{W}^\star = \operatorname*{arg\,min}_{\mathbf{W} \in \mathcal{W}} \mathbb{E}\{\ell(\hat{y}, y)\},$$

where the set $\mathcal{W}$ is the search space of interest and $\ell(\hat{y}, y)$ is a loss function that penalizes the discrepancy between an observation y and the value $\hat{y}$ that is predicted by the regression model. In the considered linear regression setting, the value $\hat{y}$ is a linear function of the parameter $\mathbf{W}$.

The formulation of linear regression problems as an optimization problem over a given search space motivates us to adopt a proper optimization framework, generalizing the classical optimization framework for linear regression on linear spaces. In this thesis, we adopt the geometric optimization framework of *optimization on matrix manifolds* (Absil et al., 2008).

The main objective of this thesis is to demonstrate that the adopted framework is effective in the design of efficient rank-constrained machine learning algorithms. For this purpose, we show throughout the present dissertation that the considered geometric optimization framework can be exploited to interpret existing algorithms but also to develop novel ones.

In addition to the problem of learning a fixed-rank matrix, the adopted framework applies to other linear regression problems defined on nonlinear matrix search spaces including the learning of rotation matrices (Arora, 2009), subspaces of $\mathbb{R}^d$ (Oja, 1992; Warmuth, 2007) or positive definite matrices (Tsuda et al., 2005; Davis et al., 2007).

1.1 Research context

The research presented in this thesis takes its roots in complementary research fields that evolved rapidly over the last few years. In the sequel, we relate the present work to the recent research developments in these areas and motivate the followed research leads.

Online learning algorithms

Online learning algorithms (Bottou, 1998) offer an appealing approach to cope with learning problems involving a large number of samples. The approach is also known as stochastic gradient descent when each iteration of the algorithm is a gradient step for the instantaneous cost function. The mathematics of stochastic gradient descent algorithms are presented in Section 2.3.

As opposed to batch learning algorithms that exploit the entire set of samples at each iteration, online learning algorithms consider each sample one at a time. The online learning approach is appealing because learning algorithms usually involve the minimization of a cost function that is expressed as the average prediction error evaluated over the complete set of available samples. Hence, the cost function is represented as a sum of a large number of individual contributions to the prediction error. If n is the total number of samples in the data set, online learning algorithms reduce the computational cost of each iteration by a factor n Bottou (1998). This is an important complexity reduction for problems with a large number of samples.

Online learning algorithms work well in practice because the averaged effect of updates is the same as for a batch algorithm. Although online algorithms have a much slower convergence

rate than batch algorithms, they generally decrease faster the prediction error in the large-scale regime (Bottou and Bousquet, 2007) and yield solutions with good generalization performance.

The online learning approach is a particular case of stochastic approximation. Stochastic approximation algorithms were originally studied by Robbins and Monro (1951) as a method to solve nonlinear equations. The method was later exploited by Kiefer and Wolfowitz (1952) for optimization purposes. Online learning algorithms have recently gained popularity in the machine learning community thanks to the work of Bottou (1998) and the advent of large-scale problems. Today, it is still an active area of research with the development of approximate second-order methods (Bordes et al., 2009; Yu et al., 2010), gradient free algorithms (Nesterov, 2011), hybrid deterministic-stochastic methods (Friedlander and Schmidt, 2011) and parallelization techniques (Zinkevich et al., 2010; Louppe and Geurts, 2010; Agarwal and Duchi, 2011).

Learning a low-rank matrix

The learning of a low-rank matrix is an active research problem that has a lot of applications in machine learning (see Chapter 2). Most of the recent algorithmic contributions in this field have focused on the design of efficient algorithms that scale to high-dimensional problems.

The growing interest in this problem has been partly stimulated by the $1 million Netflix prize (Netflix, 2006),[2] as this competition essentially amounts to solving a low-rank matrix completion problem. Moreover, the field has also benefited from the recent advances in closely related research areas such as compressed sensing (Candès and Wakin, 2008) and sparse methods (Tibshirani, 1996). The close connection with compressed sensing and sparse methods becomes clear when low-rank matrices are interpreted as matrices with a sparse singular value spectrum.

Two typical situations motivate the introduction of a low-rank constraint. The low-rank constraint might be part of the original problem formulation. For example, in matrix completion problems, it models the redundancy in the data and prevents the problem to be ill-posed. Another example is concerned with convex relaxations of combinatorial problems for which the final solution must be of low rank (Dhillon et al., 2007; d'Aspremont et al., 2007).

The low-rank constraint is also associated with dimensionality reduction when it is used as a sound approximation for reducing the original problem size. The approximation results in a dramatic decrease in the computational complexity of algorithms. Indeed, whereas algorithms based on full-rank matrices typically scale as $O(d^3)$, algorithms based on low-rank matrices typically scale as $O(dr^2)$ (Fine et al., 2001; Bach and Jordan, 2005). It is a significant complexity reduction since the matrix rank r is typically chosen very small compared to the problem size d.

Learning a low-rank matrix requires to solve a difficult nonconvex optimization problem. Convex relaxations of the problem have been proposed (Fazel, 2002). The most popular one relies on the minimization of the *nuclear norm*, a matrix norm that promotes low-rank solutions. The nuclear norm of a matrix is defined as the sum of its singular values. The nuclear norm is the largest convex lower bound of the rank function over the spectral norm unit ball.

Nuclear norm based approaches enjoy nice theoretical performance guarantees such as consistency (Bach, 2008), and sufficient conditions for the recovery of the minimum rank solution are available (Recht et al., 2010). The approach allowed for the first theoretical guarantees on exact reconstruction for low-rank matrix completion problems (Candès and Recht, 2008).

Several algorithms for nuclear norm minimization have been proposed in the literature (Cai et al., 2008; Toh and Yun, 2010; Mazumder et al., 2010; Ma et al., 2010). However, an intrinsic limitation of the approach is that the rank of intermediate solutions cannot be bounded a priori. For large-scale problems, memory requirement may thus become prohibitively large.

A different yet complementary approach that resolves this issue, assumes a fixed-rank factorization of the solution and optimizes the corresponding non-convex optimization problem (Rennie and Srebro, 2005; Keshavan et al., 2010; Jain et al., 2010; Shalit et al., 2010).

[2]The Neflix prize has been awarded in 2009 to a group of researchers that developed a recommendation algorithm improving the performance of the existing Netflix movie recommendation system by at least 10%.

This approach is complementary to nuclear norm based approaches in the sense that the most efficient algorithms for nuclear norm minimization rely on a low-rank factorization of the solution to reduce the computational cost and memory requirement.

Despite the potential introduction of local minima, algorithms based on fixed-rank factorizations achieve very good performance in practice. Moreover, Keshavan et al. (2010) and Jain et al. (2010) show that performance guarantees are also possible when a good heuristic is available for the initialization of the algorithm.

In this thesis, we pursue the research on fixed-rank factorizations and study the geometry of the underlying search space. We will show that fixed-rank factorizations admit intrinsic invariance properties that induce a geometry as a quotient space. The geometry as a quotient space stems from the fact that a given matrix can be represented by an entire equivalence class of matrices. The adopted optimization framework will be exploited to take advantage of the quotient structure of the search space in the design of efficient optimization algorithms.

Optimization on Riemannian matrix manifolds

Optimization on Riemannian matrix manifolds (Absil et al., 2008) is a natural and principled geometric optimization framework to search a Riemannian domain. The framework deals with unconstrained optimization algorithms defined over matrix manifold search spaces. It provides the tools to exploit those matrix search spaces in the design of efficient numerical algorithms. The main concepts associated with this framework are introduced in Chapter 3.

The idea of treating optimization problems defined on Riemannian manifolds was originally presented by Luenberger (1972), but has raised significant interest in the control systems community with the work of Brockett (1972, 1993) on differential equations whose solutions evolve on a manifold. The book of Helmke and Moore (1996) also deals with optimization and dynamical systems from a differential geometric viewpoint and paves the way for the derivation of numerical algorithms. The paper of Edelman et al. (1998) addresses optimization problems subject to orthogonality constraints from a differential geometric perspective. Recent advances in the field are presented in the book of Absil et al. (2008), that focuses on algorithmic developments and addresses the general problem of optimizing a smooth function on a manifold.

Recently, differential geometric optimization methods have been successfully applied to a broad variety of computational problems such as motion and structure recovery for computer vision (Ma et al., 2001), invariant subspace computation (Absil, 2003), component analysis and analysis of gene expression data (Journée, 2009), distributed consensus algorithms (Sarlette, 2009), computation of low-rank solutions of Lyapunov equations (Vandereycken and Vandewalle, 2010), low multilinear rank approximation of high-order tensors (Ishteva et al., 2011).

1.2 Contributions

This thesis applies the framework of optimization on matrix manifolds to the problem of learning a linear regression model parameterized by a fixed-rank matrix. The contributions are threefold.

A first contribution is to cast several modern machine learning applications as a linear regression problem on the set of fixed-rank matrices. This includes the learning of a low-rank distance, low-rank matrix completion and low-rank distance matrix completion, learning on data pairs, similarity based ranking and multi-task regression problems (see Chapter 2).

A second contribution is the development of novel line-search algorithms for learning a linear regression model parameterized by a fixed-rank matrix. We first develop novel line-search algorithms for learning a fixed-rank symmetric positive semidefinite matrix (Chapter 4). We then generalize the obtained algorithms to fixed-rank non-symmetric matrices (Chapter 5). The proposed generalization demands the development of novel geometries for the set of non-symmetric matrices. The proposed line-search algorithms scale to high-dimensional problems, preserve the

geometric structure of the problem, enjoy local convergence properties and confer a geometric basis to a number of recent contributions on the learning of fixed-rank matrices.

A third contribution is to equip the considered geometries with the necessary material for the design of rank-constrained second-order optimization algorithms (Chapter 6). This material is exploited in the design of efficient trust-region algorithms for two matrix completion problems. The sparse structure of these matrix completion problems is exploited to maintain a linear complexity in the number of available samples and in the leading matrix dimension. The proposed trust-region algorithms come with a well-characterized convergence theory and converge superlinearly to a local minimum of the considered cost function.

Chapter specific contributions are listed below, following the organization of the manuscript.

- Chapter 4 addresses the problem of linear regression on the set of fixed-rank symmetric positive semidefinite matrices. We show that learning a d-by-d fixed-rank symmetric positive semidefinite matrix of rank r amounts to jointly learn a r-dimensional subspace in $\mathbb{R}^d$ and a positive definite linear operator of size r in this subspace. We derive line-search algorithms exploiting two recently proposed quotient geometries of the set of fixed-rank symmetric positive semidefinite matrices (Bonnabel and Sepulchre, 2009; Journée et al., 2010). In contrast with previous contributions in the literature, no restrictions are imposed on the range space of the learned matrix. The resulting algorithms scale to high-dimensional problems, enjoy important invariance properties and connect with existing contributions in the literature. We apply the proposed algorithms to the problem of learning a quadratic distance from data. We illustrate the good performance of the algorithms on classical benchmarks and show the practical advantages of simultaneously learning the subspace and a quadratic distance within that subspace.

  This work is published in the *Journal of Machine Learning Research* (Meyer et al., 2011a). Related publications are (Meyer et al., 2009; Bonnabel et al., 2010).

- Chapter 5 addresses the problem of linear regression on the set of fixed-rank non-symmetric matrices. We first develop novel quotient geometries for the set of fixed-rank non-symmetric matrices, generalizing the work of Bonnabel and Sepulchre (2009) and Journée et al. (2010) on the quotient geometry of fixed-rank symmetric positive semidefinite matrices. We then derive novel line-search algorithms based on the proposed geometries. The resulting algorithms scale to high-dimensional problems, enjoy local convergence properties, connect to recent contributions on learning fixed-rank non-symmetric matrices, and apply to a broad range of machine learning problems. Numerical experiments on benchmarks suggest that the proposed algorithms compete with the state-of-the-art.

  This work is published in the *Proceedings of the 28th International Conference on Machine Learning* (Meyer et al., 2011b). An extended version of this conference paper is in preparation for the *Journal of Machine Learning Research*.

- Chapter 6 provides the necessary material to turn the considered first-order optimization algorithms into second-order optimization algorithms. Novel trust-region algorithms for matrix completion and distance matrix completion are proposed. The proposed trust-region algorithms enjoy superlinear convergence properties and maintain a linear complexity in the problem size. Numerical experiments on matrix completion benchmarks illustrate the good performance of the trust-region algorithms.

  This work is part of a conference paper submitted to the *50th Conference on Decision and Control* (Mishra et al., 2011a) and also part of a journal paper in preparation for the *SIAM Journal on Optimization*.

1.3 Outline

The present document is organized as follows. Chapter 2 presents the problem of linear regression on nonlinear spaces and cast a number of machine learning applications into the considered regression framework. Chapter 3 introduces the key concepts and notations related to optimization algorithms on matrix manifolds. Chapter 4 addresses the problem of linear regression on fixed-rank symmetric positive semidefinite matrices. Chapter 5 deals with the same problem but on fixed-rank non-symmetric matrices. Chapter 6 presents the necessary material to extend the proposed first-order optimization algorithms to second-order optimization algorithms. Chapter 7 summarizes the objectives and achievements of this thesis and also raises some perspectives and directions for future research. Proofs of propositions and theorems are not provided in the main body of the dissertation to lighten the exposition, but are deferred to Appendix A. Omitted derivations are provided in Appendix B.

Notations

The following conventions and notations are used throughout the thesis.

$\mathbb{R}^d$	the set of d-dimensional real column vectors.
$\mathbb{R}^{d\times r}$	the set of d-by-r real matrices with d rows and r columns.
$\mathbb{R}_*^{d\times r}$	the set of full-rank d-by-r real matrices with d rows and r columns.
$\mathcal{S}^{d-1}$	the unit sphere in $\mathbb{R}^d$, the set of unit norm vectors in $\mathbb{R}^d$
$\mathbb{RP}^{d-1}$	the real projective plane in $\mathbb{R}^d$, the set of vector directions in $\mathbb{R}^d$.
$\mathrm{St}(r,d)$	the Stiefel manifold, the set of matrices in $\mathbb{R}^{d\times r}$ with orthonormal columns.
$\mathcal{O}(d)$	the orthogonal group, the set of orthogonal matrices of $\mathbb{R}^{d\times d}$.
$S_{++}(d)$	the set of positive definite matrices of $\mathbb{R}^{d\times d}$.
$S_{+}(d)$	the set of positive semidefinite matrices of $\mathbb{R}^{d\times d}$.
$S_{+}(r,d)$	the set of d-by-d symmetric positive semidefinite matrices of rank equal to r.
$\mathcal{F}(r,d_1,d_2)$	the set of d_1-by-d_2 matrices of rank equal to r.
$\mathbb{E}\{x\}$	expectation of the random variable x.
$\mathbf{X}$	input data matrix of a regression model.
y	scalar observation associated to input data $\mathbf{X}$.
$\hat{y}$	prediction of a regression model.
$\ell(\hat{y},y)$	loss function between observation y and prediction $\hat{y}$.
$\{(\mathbf{X}_i,y_i)\}_{i=1}^n$	data set containing n data and observation pairs.

Given a vector $\mathbf{w}\in\mathbb{R}^d$ and a matrix $\mathbf{W}\in\mathbb{R}^{d\times r}$, we define the notations,

w_i	the i-th component of vector $\mathbf{w}$.
$\mathbf{w}_i$	the i-th vector of a collection of vectors.
$\|\mathbf{w}\|$	the Euclidean ℓ_2 norm of the vector $\mathbf{w}$, $\|\mathbf{w}\| = (\sum_{i=1}^d w_i^2)^{1/2}$.
$\mathbf{W}_{ij}$	the element at row i and column j in the matrix $\mathbf{W}$.
$\mathbf{W}^T$	the transpose of the matrix $\mathbf{W}$.
$\mathrm{Tr}(\mathbf{W})$	the trace of the square matrix $\mathbf{W}\in\mathbb{R}^{d\times d}$, $\mathrm{Tr}(\mathbf{W}) = \sum_{i=1}^d \mathbf{W}_{ii}$.
$\|\mathbf{W}\|_F$	the Frobenius norm of the matrix $\mathbf{W}$, $\|\mathbf{W}\|_F = \mathrm{Tr}(\mathbf{W}^T\mathbf{W})^{1/2}$.
$\mathbf{W}\succ 0$	the matrix is (symmetric) positive definite, all its eigenvalues are strictly positive.
$\mathbf{W}\succeq 0$	the matrix is (symmetric) positive semidefinite, all its eigenvalues are nonnegative.
$\odot$	binary operator between vectors or matrices that denotes element wise multiplication.
$\mathrm{Sym}(\mathbf{W})$	the symmetric part $(\mathbf{W}+\mathbf{W}^T)/2$ of $\mathbf{W}\in\mathbb{R}^{d\times d}$.
$\mathrm{Skew}(\mathbf{W})$	the skew-symmetric part $(\mathbf{W}-\mathbf{W}^T)/2$ of $\mathbf{W}\in\mathbb{R}^{d\times d}$.
$\exp(\mathbf{W})$	the matrix exponential of $\mathbf{W}\in\mathbb{R}^{d\times d}$.
$\log(\mathbf{W})$	the matrix logarithm of the positive definite matrix $\mathbf{W}\in\mathbb{R}^{d\times d}$.
$\mathrm{rank}(\mathbf{W})$	the rank of the matrix $\mathbf{W}$.
$\mathrm{Diag}(\mathbf{W})$	a vector whose elements are the diagonal elements of $\mathbf{W}$.
$\mathrm{qf}(\mathbf{W})$	$\mathbf{Q}$ factor of the QR decomposition of $\mathbf{W}$ as $\mathbf{W}=\mathbf{QR}$, where $\mathbf{Q}\in\mathbb{R}^{d\times r}$ has orthonormal columns and $\mathbf{R}\in\mathbb{R}^{r\times r}$ is upper triangular.
$\mathrm{uf}(\mathbf{W})$	$\mathbf{U}$ factor of the polar decomposition of $\mathbf{W}$ as $\mathbf{W}=\mathbf{US}$, where $\mathbf{U}\in\mathbb{R}^{d\times r}$ has orthonormal columns and $\mathbf{S}\in\mathbb{R}^{r\times r}$ is positive definite.

Given a manifold $\mathcal{W}$ and a point $\mathbf{W} \in \mathcal{W}$, we use the following notations,

$T_{\mathbf{W}}\mathcal{W}$	tangent space of $\mathcal{W}$ at point $\mathbf{W}$.
$N_{\mathbf{W}}\mathcal{W}$	normal space of $\mathcal{W}$ at point $\mathbf{W}$.
$\xi_{\mathbf{W}}, \zeta_{\mathbf{W}}$	tangent vectors at $\mathbf{W}$, that is, elements of the tangent space $T_{\mathbf{W}}\mathcal{W}$.
$g_{\mathbf{W}}(\xi_{\mathbf{W}}, \zeta_{\mathbf{W}})$	Riemannian metric (inner product) between $\xi_{\mathbf{W}}$ and $\zeta_{\mathbf{W}}$.
$\nabla_{\xi}\zeta$	Riemannian connection of the vector field ζ in the direction ξ.

Given a function $f : \mathbb{R}^{d\times r} \to \mathbb{R} : \mathbf{W} \mapsto f(\mathbf{W})$, we use the following notations,

$Df(\mathbf{W})[\xi]$	directional derivative of f with respect to $\mathbf{W}$ in a direction ξ, evaluated at $\mathbf{W}$.
$\mathrm{grad} f(\mathbf{W})$	the differential-geometric gradient or Riemannian gradient of f at $\mathbf{W}$.
$\mathrm{Hess} f(\mathbf{W})[\xi]$	the application of the Riemannian Hessian of f at $\mathbf{W}$ in a direction ξ.

The following acronyms are also used in the thesis,

PCA	principal component analysis
SVD	singular value decomposition
PSD	positive semidefinite
SDP	semidefinite programming

Chapter 2

Linear regression on nonlinear spaces

Chapter abstract: In this chapter, we cast several modern machine learning applications as a linear regression problem on a nonlinear matrix search space. Our main driving application is the learning of low-rank matrix which we will present in Section 2.1.

Following a classical approach, the linear regression problem is turned into the minimization of a quadratic cost function on the considered nonlinear matrix search space (Section 2.2).

The reformulation of machine learning problems as optimization problems defined on nonlinear matrix search spaces appeals for a proper optimization framework generalizing the classical optimization framework for linear regression on linear spaces (Section 2.3). In this thesis, we adopt the framework of optimization on Riemannian matrix manifolds (Absil et al., 2008).

The present chapter motivates the adopted framework by means of several applications of linear regression on vector and matrix search spaces (Sections 2.4 and 2.5). The presented examples demonstrate that the adopted framework has solid foundations and that it offers a rich and flexible framework in the design of linear regression algorithms on nonlinear spaces. In particular, the flexibility of the adopted framework is exploited to interpret existing algorithms but also to develop novel ones. Table 2.1 on page 24 summarizes connections with previous works.

2.1 Motivation and applications

The learning of a low-rank matrix is a fundamental problem arising in many modern machine learning applications such as low-rank matrix completion (Candès and Recht, 2008), collaborative filtering (Rennie and Srebro, 2005), learning of low-rank distances (Kulis et al., 2009) and low-rank similarity measures (Shalit et al., 2010), dimensionality reduction (Cai et al., 2007), classification with multiple classes (Amit et al., 2007), learning on data pairs (Abernethy et al., 2009), multi-task learning (Evgeniou et al., 2005), just to name a few.

This problem has attracted a lot of research attention over the last few years and is still actively researched today. Indeed, the learning of a low-rank matrix is a challenging problem that requires the resolution of a difficult nonconvex optimization problem.

With the growing size and number of large-scale problems, most of the recent research efforts have focused on the design of efficient optimization algorithms that can cope with large matrices.

Two main approaches have been proposed in the literature to efficiently address this problem.

The first approach is based on a convex relaxation of the original problem (Fazel, 2002). The approach relies on the trace (or nuclear) norm heuristic, that consists in adding a convex regularization term to the considered objective function in order to promote low-rank solutions.

The second approach assumes a low-rank parametrization of the solution and solves the resulting nonconvex optimization problem (Burer and Monteiro, 2003; Rennie and Srebro, 2005). Despite the potential introduction of local minima, this approach gives good results in practice and ensures a low computational cost and small memory requirement of the algorithms.

The approach proposed in this thesis is along the line of the second approach as we exploit the geometry of fixed-rank matrix factorizations to address the nonconvex optimization problem.

A crucial difference with the existing approach is however that the rank-constrained optimization problem is viewed as an unconstrained optimization problem over a constrained matrix search space. Our purpose is to exploit the geometry of the constrained matrix search space in the design of efficient optimization algorithms. To achieve this goal, we build on recent advances in the field of optimization on Riemannian matrix manifolds (Absil et al., 2008).

The foundations in this field are not new (Luenberger, 1972), and there has been an increasing body of research on the subject over the recent years (Helmke and Moore, 1996; Edelman et al., 1998; Absil et al., 2008). The use of optimization algorithms on matrix manifolds to solve machine learning problems is however a new and ongoing research topic (Nishimori and Akaho, 2005; Arora, 2009; Shalit et al., 2010; Meyer et al., 2011c,b). One of the main objectives of this thesis is to demonstrate that manifold based optimization provides a proper and solid framework for the design of efficient machine learning algorithms. This objective is pursued by showing throughout the thesis that the adopted framework can be exploited to interpret earlier algorithms and to design novel ones.

A first contribution of this thesis is to reformulate existing modern machine learning applications as linear regression problems on a nonlinear matrix search space. In this section, we focus on two particular applications that involve the learning of a low-rank matrix.

2.1.1 Learning of low-rank distances

Selecting an appropriate distance measure is a central issue for many distance-based classification and clustering algorithms such as nearest neighbor classifiers, support vector machines or k-means. Because this choice is highly problem-dependent, numerous methods have been proposed to learn a distance function directly from data.

Consider n data samples $\mathbf{x}_1, ..., \mathbf{x}_n \in \mathbb{R}^d$ and the associated class labels $l_1, ..., l_n \in \{1, ..., c\}$, where c is the number of possible classes. The goal of distance learning algorithms is to compute a distance function that is optimized for the classification or clustering task at hand. A classical approach (e.g. Davis et al., 2007; Weinberger and Saul, 2009) is to compute a distance $d_{\mathbf{W}}(\mathbf{x}_i, \mathbf{x}_j)$ that is a linear function of a symmetric positive semidefinite matrix $\mathbf{W} \in S_+(d)$, where

$$S_+(d) = \{\mathbf{W} \in \mathbb{R}^{d \times d} : \mathbf{W} = \mathbf{W}^T \succeq 0\}.$$

In that case, learning the distance amounts to learning the matrix $\mathbf{W}$ that parametrize the distance. Our focus will be on the learning of Mahalanobis distances (Mahalanobis, 1936),

$$d_{\mathbf{W}}(\mathbf{x}_i, \mathbf{x}_j) = (\mathbf{x}_i - \mathbf{x}_j)^T \mathbf{W} (\mathbf{x}_i - \mathbf{x}_j),$$

and kernel distances (Shawe-Taylor and Cristianini, 2004)

$$d_{\mathbf{W}}(\mathbf{x}_i, \mathbf{x}_j) = (\phi(\mathbf{x}_i) - \phi(\mathbf{x}_j))^T \mathbf{W} (\phi(\mathbf{x}_i) - \phi(\mathbf{x}_j)).$$

These two distance functions will be presented in more details the sequel. In this setting, a standard approach for learning the distance is to optimize $\mathbf{W}$ such that the resulting distance satisfies as much as possible a set of given constraints. The constraints are constructed in order to map close together samples having the same class label and to send further apart samples having different class labels. Following Davis et al. (2007), the constraints can be computed as

$$\begin{aligned} d_{\mathbf{W}}(\mathbf{x}_i, \mathbf{x}_j) &\leq y_u \quad \text{if} \quad l_i = l_j, \\ d_{\mathbf{W}}(\mathbf{x}_i, \mathbf{x}_j) &\geq y_l \quad \text{if} \quad l_i \neq l_j, \end{aligned} \tag{2.1}$$

where y_u and y_l are bounds on the distance that are chosen such that y_l is much larger than y_u.

The distance learning problem can be formulated as a linear regression problem on the set of symmetric positive definite matrices $S_+(d)$. The input data of this regression problem are

sample pairs $(\mathbf{x}_i, \mathbf{x}_j)$ and the corresponding observations are the right hand sides y_u and y_l in the definition (2.1). Provided that the distance function $d_{\mathbf{W}}(\mathbf{x}_i, \mathbf{x}_j)$ is a linear function of $\mathbf{W}$, it defines a predictive model $\hat{y}_{ij}$ of distance between samples $\mathbf{x}_i$ and $\mathbf{x}_j$. Learning the distance then amounts to solving a regression problem

$$\min_{\mathbf{W} \in S_+(d)} \sum_{(i,j) \in \mathcal{S}} \ell(\hat{y}_{ij}, y_u) + \sum_{(i,j) \in \mathcal{D}} \ell(\hat{y}_{ij}, y_l), \tag{2.2}$$

where we have defined the sets $\mathcal{S} = \{(i,j) : d_{\mathbf{W}}(\mathbf{x}_i, \mathbf{x}_j) \leq y_u\}$ and $\mathcal{D} = \{(i,j) : d_{\mathbf{W}}(\mathbf{x}_i, \mathbf{x}_j) \geq y_l\}$.

The loss function ℓ penalizes the discrepancy between an observation y_u or y_l and the value $\hat{y}_{ij}$ that is predicted by the model. A common choice for a loss function ℓ that enforces inequalities is the continuously differentiable cost function

$$\ell(\hat{y}, y) = \frac{1}{2} \max(0, \rho(\hat{y} - y))^2, \tag{2.3}$$

where $\rho = +1$ if $\hat{y} \leq y$ is required and $\rho = -1$ if $\hat{y} \geq y$ is required.

Most of the early work on distance learning has focused on the learning of a full-rank distance (e.g. Xing et al., 2002; Davis et al., 2007), that is, a distance $d_{\mathbf{W}}(\mathbf{x}_i, \mathbf{x}_j)$ is parametrized by a full-rank symmetric positive definite matrix $\mathbf{W} \in S_{++}(d)$, where

$$S_{++}(d) = \{\mathbf{W} \in \mathbb{R}^{d \times d} : \mathbf{W} = \mathbf{W}^T \succ 0\}.$$

The learning of full-rank positive definite matrix requires to search for a parameter in a space of dimension $O(d^2)$, which prevents the use of full-rank distance learning algorithms in a growing number of high-dimensional problems. To address this issue, recent algorithmic work has focused on the learning of low-rank distances (Davis and Dhillon, 2008; Kulis et al., 2009).

Indeed, learning a low-rank (or identity plus low-rank) matrix instead of a full-rank one reduces the dimension of the search space to $O(dr)$, where r is the rank of the matrix. This dramatically reduces the computational cost and memory requirement of distance learning algorithms since the rank r is typically chosen very small compared to the problem size d.

The learning of a low-rank distance can be formulated as (2.2), where the search space $S_+(d)$ is now replaced by the set of rank r symmetric positive semidefinite matrices

$$S_+(r, d) = \{\mathbf{W} \in \mathbb{R}^{d \times d} : \mathbf{W}^T = \mathbf{W} \succeq 0, \ \text{rank}(\mathbf{W}) = r\}.$$

We now present two important distance models $d_{\mathbf{W}}(\mathbf{x}_i, \mathbf{x}_j)$ that are compatible with the considered linear regression model and review some literature on the learning of these distances.

Kernel learning

In kernel-based methods (Shawe-Taylor and Cristianini, 2004), the data samples $\mathbf{x}_1, ..., \mathbf{x}_n$ are first transformed by a nonlinear mapping $\phi : \mathbf{x} \in \mathcal{X} \mapsto \phi(\mathbf{x}) \in \mathcal{F}$, where $\mathcal{F}$ is a new feature space that is expected to facilitate pattern detection into the data.

The kernel function is then defined as the dot product between any two samples in $\mathcal{F}$,

$$\kappa(\mathbf{x}_i, \mathbf{x}_j) = \phi(\mathbf{x}_i) \cdot \phi(\mathbf{x}_j).$$

In practice, the kernel function is represented by a positive semidefinite matrix $\mathbf{K} \in \mathbb{R}^{n \times n}$ whose entries are defined as $\mathbf{K}_{ij} = \phi(\mathbf{x}_i) \cdot \phi(\mathbf{x}_j)$. This inner product information is used solely to compute the relevant quantities needed by the algorithms based on the kernel. For instance, a distance is implicitly defined by any kernel function as the Euclidean distance between the samples in the new feature space

$$d_\phi(\mathbf{x}_i, \mathbf{x}_j) = \|\phi(\mathbf{x}_i) - \phi(\mathbf{x}_j)\|^2 = \kappa(\mathbf{x}_i, \mathbf{x}_i) + \kappa(\mathbf{x}_j, \mathbf{x}_j) - 2\kappa(\mathbf{x}_i, \mathbf{x}_j).$$

The previous distance function can be evaluated using only the elements of the kernel matrix by the formula

$$d_\phi(\mathbf{x}_i, \mathbf{x}_j) = \mathbf{K}_{ii} + \mathbf{K}_{jj} - 2\mathbf{K}_{ij} = \mathrm{Tr}\left(\mathbf{K}(\mathbf{e}_i - \mathbf{e}_j)(\mathbf{e}_i - \mathbf{e}_j)^T\right) = (\mathbf{e}_i - \mathbf{e}_j)^T\mathbf{K}(\mathbf{e}_i - \mathbf{e}_j),$$

which is a linear function of $\mathbf{K}$ and thus fits into the considered linear regression framework.

Learning a kernel consists in computing the kernel (or Gram) matrix from scratch or improving an existing kernel matrix based on side-information (in a semi-supervised setting for instance). Data samples and class labels are generally exploited by means of equality or inequality constraints involving pairwise distances or inner products.

Most of the numerous kernel learning algorithms that have been proposed work in the so-called transductive setting, that is, it is not possible to generalize the learned kernel function to new data samples (Kwok and Tsang, 2003; Lanckriet et al., 2004; Tsuda et al., 2005; Zhuang et al., 2009; Kulis et al., 2009). In that setting, the total number of considered samples is known in advance and determines the size of the learned matrix. Recently, algorithms have been proposed to learn a kernel function that can be extended to new points (Davis et al., 2007; Chatpatanasiri et al., 2010; Jain et al., 2010). In this thesis, we only consider the kernel learning problem in the transductive setting. When low-rank matrices are considered, kernel learning algorithms can be then regarded as dimensionality reduction methods. Depending on the chosen cost function, several dimensionality reduction criterion are obtained. Very popular unsupervised algorithms in that context are kernel principal component analysis (Schölkopf et al., 1998) and multidimensional scaling (Cox and Cox, 2001; Borg and Groenen, 2005). Other kernel learning techniques include the maximum variance unfolding algorithm (Weinberger et al., 2004) and its semi-supervised version (Song et al., 2007), and the kernel spectral regression framework (Cai et al., 2007) which encompasses many reduction criterion (for example, linear discriminant analysis (LDA), locality preserving projection (LPP), neighborhood preserving embedding (NPE)). See the survey of Yang (2006) for a more complete state-of-the-art in this area.

Since our algorithms are able to compute a low-rank kernel matrix from data, they can be used for unsupervised or semi-supervised dimensionality reduction, depending on whether or not the class labels are exploited through the imposed constraints.

Mahalanobis distance learning

Mahalanobis distances generalize the usual Euclidean distance as it allows to transform the data with an arbitrary rotation and scaling before computing the distance. Let $\mathbf{x}_i, \mathbf{x}_j \in \mathbb{R}^d$ be two data samples, the (squared) Mahalanobis distance between these two samples is parameterized by a positive definite matrix $\mathbf{W} \in \mathbb{R}^{d\times d}$ and writes as

$$d_{\mathbf{W}}(\mathbf{x}_i, \mathbf{x}_j) = (\mathbf{x}_i - \mathbf{x}_j)^T\mathbf{W}\ (\mathbf{x}_i - \mathbf{x}_j). \tag{2.4}$$

The previous distance function is again a linear function of its parameter $\mathbf{W}$ and thus qualifies as a valid distance for the considered linear regression framework.

When $\mathbf{W}$ is equal to the identity matrix, the standard Euclidean distance is obtained. Another frequently used matrix is $\mathbf{W} = \mathbf{\Sigma}^{-1}$, the inverse of the sample covariance matrix. For centered data features, computing this Mahalanobis distance is equivalent to performing a whitening of the data before computing the Euclidean distance.

For low-rank Mahalanobis matrices, computing the distance is equivalent to first performing a linear data reduction step before computing the Euclidean distance on the reduced data.[1] Learning a low-rank Mahalanobis matrix can thus be seen as learning a linear projector that is used for dimension reduction.

[1] In the low-rank case, one should rigorously refer to (2.4) as a pseudo-distance. Indeed, one has $d_{\mathbf{W}}(\mathbf{x}_i, \mathbf{x}_j) = 0$ with $\mathbf{x}_i \neq \mathbf{x}_j$ whenever $(\mathbf{x}_i - \mathbf{x}_j)$ lies in the null space of $\mathbf{W}$.

In contrast to kernel functions, Mahalanobis distances easily generalize to new data samples since the sole knowledge of $\mathbf{W}$ determines the distance function.

In recent years, Mahalanobis distance learning algorithms have been the subject of many contributions that cannot be all enumerated here. We review a few of them, most relevant for the present work. The first proposed methods have been based on successive projections onto a set of large margin constraints (Xing et al., 2002; Shalev-Shwartz et al., 2004). The method proposed by Globerson and Roweis (2005) seeks a Mahalanobis matrix that maximizes the between classes distance while forcing to zero the within classes distance. A simpler objective is pursued by the algorithms that optimize the Mahalanobis distance for the specific k-nearest neighbor classifier (Goldberger et al., 2004; Torresani and Lee, 2006; Weinberger and Saul, 2009). Bregman projection based methods minimize a particular Bregman divergence under distance constraints. Both batch (Davis et al., 2007) and online (Jain et al., 2008) formulations have been proposed for learning full-rank matrices. Low-rank matrices have also been considered with Bregman divergences but only when the range space of the matrix is fixed in the first place (Davis and Dhillon, 2008; Kulis et al., 2009).

2.1.2 Low-rank matrix completion

The problem of low-rank matrix completion amounts to estimating the missing entries of a matrix from a very limited number of its entries. There has been a large number of research contributions on this subject over the last years, addressing the problem both from a theoretical (e.g. Candès and Recht, 2008; Gross, 2011) and from algorithmic point of view (e.g. Rennie and Srebro, 2005; Cai et al., 2008; Lee and Bresler, 2009; Meka et al., 2009; Keshavan et al., 2010; Simonsson and Eldén, 2010; Jain et al., 2010; Mazumder et al., 2010).

Let $\mathbf{W}^\star \in \mathbb{R}^{d_1 \times d_2}$ be a matrix whose entries $\mathbf{W}^\star_{ij}$ are only given for some indices $(i,j) \in \Omega$, where Ω is a subset of the complete set of indices $\{(i,j) : i \in \{1,...,d_1\} \text{ and } j \in \{1,...,d_2\}\}$.

Low-rank matrix completion amounts to solving the following optimization problem

$$\min_{\mathbf{W} \in \mathbb{R}^{d_1 \times d_2}} \|\mathcal{P}_\Omega(\mathbf{W}) - \mathcal{P}_\Omega(\mathbf{W}^\star)\|_F^2, \quad \text{subject to } \operatorname{rank}(\mathbf{W}) = r, \tag{2.5}$$

where the function $\mathcal{P}_\Omega(\cdot)$ is a projection onto the set of known entries,

$$\mathcal{P}_\Omega(\mathbf{W})_{ij} = \begin{cases} \mathbf{W}_{ij}, & \text{if } (i,j) \in \Omega, \\ 0, & \text{otherwise.} \end{cases}$$

The rank constraint is part of the problem formulation. It models the fact that the rows and columns of $\mathbf{W}^\star$ contain redundant patterns. The number of given entries $|\Omega|$ is typically much smaller than $d_1 d_2$, the total number of entries in $\mathbf{W}^*$. Recent contributions provide conditions on $|\Omega|$ under which exact reconstruction is possible from a set of entries sampled uniformly and at random (Candès and Recht, 2008; Cai et al., 2008; Keshavan et al., 2010).

Collaborative filtering (Rennie and Srebro, 2005; Abernethy et al., 2009) is an important application of the low-rank matrix completion problem. Given a matrix containing the preference ratings of users about some items (say movies or songs), the goal of collaborative filtering algorithms is to compute automatic and personalized recommendations of these items. The rating matrix has as many rows as users and as many columns as items. The element (i,j) of the rating matrix represents the rating of user i for item j. Only a few ratings per user are available and one expects redundant patterns (e.g. users with similar tastes, related items). This problem is naturally formulated as a low-rank matrix completion problem for which the predicted ratings are exploited in the recommendation of items to users.

We now show that low-rank matrix completion can be formulated as a linear regression problem over the set of non-symmetric fixed-rank matrices

$$\mathcal{F}(r, d_1, d_2) = \{\mathbf{W} \in \mathbb{R}^{d_1 \times d_2} : \operatorname{rank}(\mathbf{W}) = r\}.$$

Indeed, provided each known entry $\mathbf{W}^\star_{ij}$ with $(i,j) \in \Omega$ is regarded as an observation y_{ij} and the data are given by $\mathbf{X}_{ij} = \mathbf{e}_j \mathbf{e}_i^T$, where $\mathbf{e}_j \in \mathbb{R}^{d_2}$ and $\mathbf{e}_i \in \mathbb{R}^{d_1}$ are canonical basis vector, we can define the linear regression model

$$\hat{y}_{ij} = \mathrm{Tr}(\mathbf{W}\mathbf{X}_{ij}) = \mathrm{Tr}(\mathbf{W}\mathbf{e}_j\mathbf{e}_i^T) = \mathbf{e}_i^T\mathbf{W}\mathbf{e}_j = \mathbf{W}_{ij}.$$

With a quadratic cost function $(\hat{y}_{ij} - y_{ij})^2$, the following linear regression problem:

$$\min_{\mathbf{W}\in\mathcal{F}(r,d_1,d_2)} \sum_{(i,j)\in\Omega} (\hat{y}_{ij} - y_{ij})^2 \tag{2.6}$$

is equivalent to the optimization problem (2.5). The next section presents a more general formulation of the linear regression problem in nonlinear search spaces.

2.2 Problem formulation

Given data $\mathbf{X}$ in an input vector space $\mathcal{X}$ and observations y in an output vector space $\mathcal{Y}$, the purpose of linear regression is to compute a linear predictive model $\hat{y} : \mathcal{X} \to \mathcal{Y}$ whose parameter $\mathbf{W}$ belongs to a search space $\mathcal{W}$, such that the optimal fit $\mathbf{W}^\star$ minimizes the *expected cost*

$$\mathbf{W}^\star = \arg\min_{\mathbf{W}\in\mathcal{W}} \mathbb{E}_{\mathbf{X},y}\{\ell(\hat{y}, y)\}, \tag{2.7}$$

where

$$\mathbb{E}_{\mathbf{X},y}\{\ell(\hat{y}, y)\} = \int \ell(\hat{y}, y)\ dP(\mathbf{X}, y). \tag{2.8}$$

The loss function $\ell(\hat{y}, y)$ penalizes the discrepancy between predictions $\hat{y}$ and observations y. Expectation (2.8) is taken with respect to the (unknown) joint probability distribution over data and observation pairs. In the context of regression, observations are typically scalar ($\mathcal{Y} = \mathbb{R}$) and a classical choice for the loss function is the quadratic loss

$$\ell(\hat{y}, y) = \frac{1}{2}(\hat{y} - y)^2. \tag{2.9}$$

Several variants of the problem are obtained depending on the input space $\mathcal{X}$ and the search space $\mathcal{W}$. In Section 2.4, we review examples of linear regression on vector search spaces. In Section 2.5, we focus on examples of matrix search spaces.

Our main interest will be on nonlinear search spaces $\mathcal{W}$ that have the structure of a matrix manifold. Matrix manifold search spaces lie at the interface between linear algebra and differential geometry. They can be intuitively thought of as a smoothly curved matrix spaces that locally look like a Euclidean space, but with a possibly more complex global structure.

We will consider the case of embedded submanifolds and quotient manifolds. Embedded submanifolds are matrix spaces in $\mathbb{R}^{d\times r}$ that are defined by means of an explicit set of algebraic constraints. They can be viewed as a generalization of the notion of surface in $\mathbb{R}^d$. Quotient manifolds are matrix spaces defined as sets of equivalence classes, that is, each point of the quotient space is an equivalence class defined in $\mathbb{R}^{d\times r}$.

These concepts will be described with more details in Chapter 3. In the next section, we turn our attention to optimization algorithms for solving the linear regression problem (2.7).

2.3 Gradient based learning

Because the joint probability distribution $P(\mathbf{X}, y)$ over data and observation pairs is typically unknown, it is generally not possible to compute the expected cost (2.8) explicitly. It is however possible to compute an approximation of the expected cost by considering the *empirical cost*

$$f_n(\mathbf{W}) = \frac{1}{n}\sum_{i=1}^{n} \ell(\hat{y}_i, y_i), \tag{2.10}$$

which is the average loss computed over a finite number of samples $\{(\mathbf{X}_i, y_i)\}_{i=1}^n$. Minimizing the empirical cost (2.10) can be achieved using a batch gradient descent algorithm. Successive estimates $\mathbf{W}_t$ of the optimal parameter are then computed using the formula

$$\mathbf{W}_{t+1} = \mathbf{W}_t - s_t \operatorname{grad} f_n(\mathbf{W}_t) = \mathbf{W}_t - s_t \frac{1}{n} \sum_{i=1}^{n} \operatorname{grad} \ell(\hat{y}_i, y_i).$$

In contrast with batch gradient descent algorithms, online gradient descent algorithms (Bottou, 2004) consider possibly infinite sets of samples $\{(\mathbf{X}_t, y_t)\}_{t\geq 1}$, received one at a time or by mini-batches. At time t, the online learning algorithm minimizes the *instantaneous cost*

$$f_t(\mathbf{W}) = \frac{1}{b} \sum_{\tau=t}^{t+b} \ell(\hat{y}_\tau, y_\tau),$$

where b is the mini-batch size. When $b = 1$, one recovers plain stochastic gradient descent.

In the sequel, we only present plain stochastic gradient descent versions of algorithms to shorten the exposition. The single necessary change to convert a stochastic gradient descent algorithm into its batch counterpart is to perform, at each iteration, the minimization of the empirical cost f_n instead of the minimization of the instantaneous cost f_t. We will omit the index t and denote by f the (instantaneous) cost function that is minimized at each iteration.

A typical iteration of the learning algorithm is therefore written as

$$\mathbf{W}_{t+1} = \mathbf{W}_t - s_t \operatorname{grad} f(\mathbf{W}_t), \tag{2.11}$$

where $\operatorname{grad} f(\mathbf{W}_t)$ is the gradient of the considered cost function and $s_t > 0$ is the step size.

The previous iteration is only valid provided that the search space $\mathcal{W}$ is a vector space. However, for the nonlinear search spaces considered in this dissertation, update (2.11) cannot generally be applied. The adopted optimization framework resolves this issue by formulating the problem directly on the considered manifold search space $\mathcal{W}$. As a consequence, each step of the optimization algorithm ensures that the next iterate remains a feasible solution.

Following Absil et al. (2008), an abstract gradient descent algorithm generalizing (2.11) to Riemannian matrix manifold search spaces is given by the update formula

$$\mathbf{W}_{t+1} = R_{\mathbf{W}_t}(-s_t \operatorname{grad} f(\mathbf{W}_t)). \tag{2.12}$$

The meaning of formula (2.12) is as follows. The Riemannian gradient $\operatorname{grad} f(\mathbf{W}_t)$ is an element of the tangent space $T_{\mathbf{W}_t}\mathcal{W}$ at $\mathbf{W}_t$, that locally encodes the set of search directions that are consistent with the geometry of the search space. The Riemannian gradient depends on the chosen Riemannian metric $g_{\mathbf{W}}$, which is a smoothly varying inner product between elements of the tangent space $T_{\mathbf{W}}\mathcal{W}$ at $\mathbf{W}$. The scalar $s_t > 0$ is the step size. The retraction $R_{\mathbf{W}_t}$ is a mapping from the tangent space $T_{\mathbf{W}_t}\mathcal{W}$ to the Riemannian manifold $\mathcal{W}$. The retraction allows us to efficiently update the search variable $\mathbf{W}$ and to maintain it within the search space of interest. Under mild conditions on the retraction $R_{\mathbf{W}_t}$, the classical convergence theory of line-search algorithms in vector spaces generalizes to Riemannian manifolds (Absil et al., 2008).

A conceptual illustration of the update formula (2.12) is provided in Figure 2.1. For more details, we refer the reader to Chapter 3 that explains these concepts in more details and covers the necessary material that allows us to design optimization algorithms.

2.4 Linear regression in vector search spaces

We first consider the case where the parameter of the linear regression model is a d-dimensional vector $\mathbf{w}$. The data input space $\mathcal{X}$ is $\mathbb{R}^d$ and the considered regression model writes as

$$\hat{y} = \mathbf{w}^T \mathbf{x}. \tag{2.13}$$

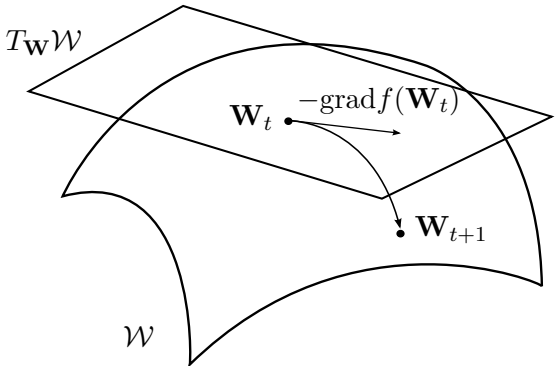


Figure 2.1: Conceptual illustration of a gradient descent step on a Riemannian manifold $\mathcal{W}$. The search direction $\text{grad}f(\mathbf{W}_t)$ belongs to the tangent space $T_{\mathbf{W}_t}\mathcal{W}$ at point $\mathbf{W}_t$. The retraction mapping automatically maintains the updated point $\mathbf{W}_{t+1}$ inside the manifold.

2.4.1 Linear regression in $\mathbb{R}^d$

The simplest example of linear regression problem in vector spaces is obtained when the search space $\mathcal{W}$ is $\mathbb{R}^d$. Linear regression in $\mathbb{R}^d$ is a fundamental problem, central to many science disciplines: biology, social and behavioral sciences, finance and economics, just to name a few. It was originally discussed by Legendre (1805) and Gauss (1809) who were both interested in determining comet orbits from astronomical observations. The statistical foundations of the method date back from the early 1900's and were pioneered by Yule (1897) and Pearson et al. (1903). The corresponding least square optimization problem is typically formulated as

$$\min_{\mathbf{w}\in\mathbb{R}^d} \mathbb{E}_{\mathbf{x},y}\{f(\mathbf{w})\}, \text{ with } f(\mathbf{w}) = \frac{1}{2}(\mathbf{w}^T\mathbf{x} - y)^2, \tag{2.14}$$

and an online gradient descent algorithm for solving (2.14) is then given by,

$$\mathbf{w}_{t+1} = \mathbf{w}_t - s_t(\mathbf{w}_t^T\mathbf{x}_t - y_t)\mathbf{x}_t. \tag{2.15}$$

This algorithm is both known as the Widrow-Hoff algorithm or the Least Mean Square (LMS) algorithm. As an aside, observe that (2.15) can be interpreted as a particular case of (2.12). The Euclidean metric turns $\mathbb{R}^d$ into a (flat) Riemannian manifold. For a scalar function $f : \mathbb{R}^d \to \mathbb{R}$ of $\mathbf{w}$, and the usual metric $g_{\mathbf{w}}(\boldsymbol{\delta}_1, \boldsymbol{\delta}_2) = \boldsymbol{\delta}_1^T\boldsymbol{\delta}_2$, the gradient satisfies

$$Df(\mathbf{w})[\boldsymbol{\delta}] = \boldsymbol{\delta}^T\text{grad}f(\mathbf{w}),$$

where $Df(\mathbf{w})[\boldsymbol{\delta}]$ is the directional derivative of f in the direction $\boldsymbol{\delta}$,

$$Df(\mathbf{w})[\boldsymbol{\delta}] = \lim_{t\to 0} \frac{f(\mathbf{w} + t\boldsymbol{\delta}) - f(\mathbf{w})}{t}.$$

In the case of interest, the gradient of the cost function is given by

$$\text{grad}f(\mathbf{w}) = (\hat{y}_t - y_t)\mathbf{x}_t = (\mathbf{w}_t^T\mathbf{x}_t - y_t)\mathbf{x}_t, \tag{2.16}$$

and the natural retraction

$$\mathbf{w}_{t+1} = R_{\mathbf{w}_t}(-s_t\ \text{grad}f(\mathbf{w}_t)) = \mathbf{w}_t - s_t\ \text{grad}f(\mathbf{w}_t), \tag{2.17}$$

induces a line-search along "straight lines", which are paths of shortest length in Euclidean spaces. Combining the gradient (2.16) with the retraction (2.17), one arrives at (2.15).

2.4.2 Linear regression with positive weights

A variant of the previous regression problem is obtained when the parameter is restricted to be a vector with positive entries, that is, $w_i > 0$, $i = 1, ..., d$. The search space is then $\mathcal{W} = \mathbb{R}^d_+$.

To handle this variant of the problem, Kivinen and Warmuth (1997) propose the exponentiated gradient descent algorithm

$$\mathbf{w}_{t+1} = \mathbf{w}_t \odot \exp(-s_t(\mathbf{w}_t^T \mathbf{x}_t - y_t)\mathbf{x}_t), \tag{2.18}$$

where $\odot$ denotes element-wise multiplication. Vector exponential is also performed element-wise. Obviously, this iteration preserves positive entries, $\mathbf{w}_t > 0 \Rightarrow \mathbf{w}_{t+1} > 0$. The variant where $\mathbf{w}$ is a probability vector, that is, $w_i > 0$ and $\sum_{i=1}^d w_i = 1$, can also be handled by normalizing the parameter $\mathbf{w}$ at each iteration so that the components add up to one.

The exponentiated gradient algorithm is well-known in the context of boosting algorithms, where it appeared in the first version of the AdaBoost algorithm (Freund and Schapire, 1997). Boosting and exponentiated gradient also have a nice geometric interpretation in terms of entropy based projections (see Kivinen and Warmuth, 1999).

The exponentiated gradient update (2.18) can be interpreted as a particular case of (2.12). Indeed, the log-Euclidean representation

$$\mathbf{w} = \exp(\mathbf{s}), \quad \mathbf{s} \in \mathbb{R}^d,$$

where exponentiation is performed element wise, is a global diffeomorphism from $\mathbb{R}^d$ to $\mathbb{R}^d_+$. With the usual Euclidean metric $g_{\mathbf{w}}(\boldsymbol{\delta}_1, \boldsymbol{\delta}_2) = \boldsymbol{\delta}_1^T \boldsymbol{\delta}_2$, the gradient of the considered quadratic cost function is again given by (2.16), and the retraction

$$R_{\mathbf{w}_t}(-s_t \operatorname{grad} f(\mathbf{w}_t)) = \mathbf{w}_t \odot \exp(-s_t \operatorname{grad} f(\mathbf{w}_t)), \tag{2.19}$$

induces a line-search in $\mathbb{R}^d_+$. Combining (2.16) with (2.19), one recovers update (2.18).

2.5 Linear regression in matrix search spaces

We now turn to the case where the parameter of the linear regression model is a matrix.

2.5.1 Linear regression on orthogonal matrices

Given data $\mathbf{x} \in \mathbb{R}^d$, observations $\mathbf{y} \in \mathbb{R}^d$, and a regression model $\hat{\mathbf{y}} = \mathbf{Q}\mathbf{x}$ the purpose is to compute an orthogonal matrix $\mathbf{Q}$ that solves the optimization problem

$$\min_{\mathbf{Q} \in \mathcal{O}(d)} \mathbb{E}_{\mathbf{x},\mathbf{y}}\{f(\mathbf{Q})\}, \text{ with } f(\mathbf{Q}) = \frac{1}{2}\|\mathbf{Q}\mathbf{x} - \mathbf{y}\|_2^2,$$

where $\mathcal{O}(d)$ is the orthogonal group, that is, the set of d-by-d orthogonal matrices

$$\mathcal{O}(d) = \{\mathbf{Q} \in \mathbb{R}^{d \times d} : \mathbf{Q}\mathbf{Q}^T = \mathbf{Q}^T\mathbf{Q} = \mathbf{I}\}.$$

The learning of an orthogonal matrix has applications in various fields (see Arora, 2009, and references therein). As a linear transformation, an orthogonal matrix acts as a rotation. Therefore, the set $\mathcal{O}(d)$ is often referred to as the set of rotation matrices.

In the batch setting, that is, when all the samples $(\mathbf{x}_1, \mathbf{y}_1), ...(\mathbf{x}_n, \mathbf{y}_n)$ are available up front, the problem is known as the orthogonal Procrustes problem (Schonemann, 1966),

$$\min_{\mathbf{Q} \in \mathcal{O}(d)} ||\mathbf{Q}\mathbf{X} - \mathbf{Y}||_F^2,$$

where $\mathbf{X} = [\mathbf{x}_1 \ ... \ \mathbf{x}_n] \in \mathbb{R}^{d \times n}$ and $\mathbf{Y} = [\mathbf{y}_1 \ ... \ \mathbf{y}_n] \in \mathbb{R}^{d \times n}$. This problem admits the closed-form solution

$$\mathbf{Q} = \text{uf}(\mathbf{X}^T \mathbf{Y}),$$

where $\text{uf}(\cdot)$ extracts the orthogonal factor of the polar decomposition of its argument and is computed as $\text{uf}(\mathbf{A}) = \mathbf{U}\mathbf{V}^T$, where $\mathbf{A} = \mathbf{U}\mathbf{\Sigma}\mathbf{V}^T$ is the thin singular value decomposition of $\mathbf{A}$.

In a recent paper, Arora (2009) considers the problem in the online setting and proposes the online learning update

$$\mathbf{Q}_{t+1} = \mathbf{Q}_t \exp(-s_t \, \text{Skew}(\mathbf{Q}_t^T (\hat{\mathbf{y}}_t - \mathbf{y}_t)\mathbf{x}_t^T)), \tag{2.20}$$

where $\text{Skew}(\cdot)$ extracts the skew-symmetric part of its argument, $\text{Skew}(\mathbf{A}) = (\mathbf{A} - \mathbf{A}^T)/2$.

Arora (2009) motivates update (2.20) as a line-search algorithm exploiting the geometry of $\mathcal{O}(d)$ as a Lie group (Boothby, 1986). He further provides an interpretation of the update in terms of the von Neumann divergence, which is a particular case of Bregman divergence (Bauschke and Borwein, 1997; Dhillon and Tropp, 2007).

Equivalently, update (2.20) can be regarded as a line-search algorithm on the orthogonal group $\mathcal{O}(d)$, viewed as an embedded submanifold of $\mathbb{R}^{d \times d}$ (Absil et al., 2008). The main concepts related to optimization on embedded submanifolds are explained in Chapter 3.

In particular, we will show in Section 3.4.1 that the Riemannian gradient of $f(\mathbf{Q})$ for the Euclidean metric $g_{\mathbf{Q}}(\mathbf{\Delta}_1, \mathbf{\Delta}_2) = \text{Tr}(\mathbf{\Delta}_1^T \mathbf{\Delta}_2)$ is given by

$$\text{grad} \, f(\mathbf{Q}) = \mathbf{Q}\text{Skew}(\mathbf{Q}^T (\hat{\mathbf{y}} - \mathbf{y})\mathbf{x}^T), \tag{2.21}$$

and is an element of the tangent space $T_{\mathbf{Q}}\mathcal{O}(d)$ at $\mathbf{Q}$. Combining (2.21) with the retraction

$$R_{\mathbf{Q}_t}(-s_t \ \text{grad} f(\mathbf{Q}_t)) = \mathbf{Q}_t \exp(\mathbf{Q}_t^T (-s_t \text{grad} f(\mathbf{Q}_t))),$$

induces a line-search algorithm on $\mathcal{O}(d)$, and yields update (2.20) of Arora (2009).

2.5.2 Linear regression on positive definite matrices

This section presents linear regression on the set of positive definite matrices, which is the matrix generalization of linear regression on positive vectors (Section 2.4.2). An application of this problem in machine learning is the learning of full-rank distance from data (Section 2.1.1).

Positive definite matrices are fundamental objects arising in many applications of engineering and applied mathematics including statistics (Smith, 2005), optimization (Boyd and Vandenberghe, 2004), systems and control theory (Boyd et al., 1994), machine learning (Shawe-Taylor and Cristianini, 2004), graph theory (Merris, 1994), just to name a few.

The set of positive definite matrices forms a convex cone that has a rich and well-studied Riemannian geometry (Faraut and Koranyi, 1994; Arsigny et al., 2007).

Linear regression on positive definite matrices is formulated as follows. Given input data matrices $\mathbf{X} \in \mathbb{R}^{d \times d}$, observations $y \in \mathbb{R}_+$, and a linear regression model $\hat{y} = \text{Tr}(\mathbf{W}\mathbf{X})$, the problem amounts to solving the optimization problem

$$\min_{\mathbf{W} \in S_{++}(d)} \mathbb{E}_{\mathbf{X},y}\{f(\mathbf{W})\}, \text{ with } f(\mathbf{W}) = \frac{1}{2}(\text{Tr}(\mathbf{W}\mathbf{X}) - y)^2, \tag{2.22}$$

where the search space $S_{++}(d)$ is the cone of d-by-d symmetric positive definite matrices,

$$S_{++}(d) = \{\mathbf{W} \in \mathbb{R}^{d \times d} : \mathbf{W}^T = \mathbf{W} \succ 0\}.$$

Since $\mathbf{W}$ is symmetric, only the symmetric part $\text{Sym}(\mathbf{X})$ of $\mathbf{X}$ contributes to the trace and the linear regression model can be equivalently written as $\hat{y} = \text{Tr}(\mathbf{W}\text{Sym}(\mathbf{X}))$.

Two algorithms have been recently proposed to solve this problem. Tsuda et al. (2005) propose a generalization of the exponentiated gradient algorithm (2.18),

$$\mathbf{W}_{t+1} = \exp(\log \mathbf{W}_t - s_t(\hat{y}_t - y_t)\mathrm{Sym}(\mathbf{X}_t)), \tag{2.23}$$

where $\exp(\cdot)$ and $\log(\cdot)$ are matrix exponential and matrix logarithm respectively. The update automatically maintains the iterate within the search space of interest, $\mathbf{W}_t \succ 0 \Rightarrow \mathbf{W}_{t+1} \succ 0$. Davis et al. (2007) propose a different algorithm for learning positive definite matrices,

$$\mathbf{W}_{t+1} = \mathbf{W}_t - s_t(\hat{y}_t - y_t)\mathbf{W}_t\mathrm{Sym}(\mathbf{X}_t)\mathbf{W}_t. \tag{2.24}$$

With a careful selection of the step size, the iterate also remains in the search space of interest.

Sections 4.2.2 and 4.2.3 show how these two algorithms can be interpreted as particular cases of the general update (2.12) for particular choices of the retraction and the Riemannian metric.

2.5.3 Linear regression on subspaces

Subspace learning (Oja, 1992; Crammer, 2006; Warmuth, 2007) is a recurring problem arising in various adaptive signal processing tasks (Delmas and Cardoso, 1998; Balzano et al., 2010).

In the online learning setting, this problem amounts to tracking the r-dimensional dominant subspace spanned by a given sequence of d-dimensional input vectors $\{\mathbf{x}_t\}_{t\geq 0}$.

Following Smith (1993), this problem can be naturally formulated as a learning problem on the Grassmann manifold $\mathrm{Gr}(r, d)$, the set of r-dimensional subspaces in $\mathbb{R}^d$.

Riemannian optimization algorithms on the Grassmann manifold $\mathrm{Gr}(r, d)$ have been well studied in the litterature (Edelman et al., 1998; Absil et al., 2004).

As a subspace in $\mathrm{Gr}(r, d)$ is an abstract concept, a concrete representation must be chosen for numerical optimization purposes. Following Edelman et al. (1998), we choose to represent subspaces by means of orthonormal matrices $\mathbf{U} \in \mathrm{St}(r, d)$ in the Stiefel manifold

$$\mathrm{St}(r, d) = \{\mathbf{U} \in \mathbb{R}^{d\times r} : \mathbf{U}^T\mathbf{U} = \mathbf{I}\}.$$

For a given subspace $\mathrm{range}(\mathbf{U}) \in \mathrm{Gr}(r, d)$, the representation $\mathbf{U} \in \mathrm{St}(r, d)$ is however not unique since $\mathrm{range}(\mathbf{U}) = \mathrm{range}(\mathbf{UQ})$ for any rotation matrix $\mathbf{Q} \in \mathcal{O}(r)$. This observation naturally leads to a representation of the Grassmann manifold as a quotient space

$$\mathrm{Gr}(r, d) \simeq \mathrm{St}(r, d)/\mathcal{O}(r), \tag{2.25}$$

representing the set of equivalence classes $[\mathbf{U}] = \{\mathbf{UQ} : \mathbf{Q} \in \mathcal{O}(r)\}$ in the matrix space $\mathbb{R}^{d\times r}$. The main concepts related to quotient manifolds will be further explained in Chapter 3.

With the chosen representation of $\mathrm{Gr}(r, d)$, the learning of a subspace can be formulated as

$$\min_{[\mathbf{U}]\in\mathrm{Gr}(r,d)} \mathbb{E}_{\mathbf{x}}\{f(\mathbf{U})\}, \text{ with } f(\mathbf{U}) = \frac{1}{2}\|\mathbf{U}\mathbf{U}^T\mathbf{x} - \mathbf{x}\|_2^2, \tag{2.26}$$

where $\mathbf{x} \in \mathbb{R}^d$ plays the role of both input data and observation. Observe that the cost function is invariant by rotation $f(\mathbf{UQ}) = f(\mathbf{U})$ and is thus a function of the equivalence class $[\mathbf{U}]$.

An online algorithm to solve (2.26) can be derived using the material presented by Edelman et al. (1998). The Euclidean metric $g_{\mathbf{U}}(\boldsymbol{\Delta}_1, \boldsymbol{\Delta}_2) = \mathrm{Tr}(\boldsymbol{\Delta}_1^T\boldsymbol{\Delta}_2)$ in $\mathrm{St}(r, d)$ induces a proper metric on the quotient space $\mathrm{Gr}(r, d) \simeq \mathrm{St}(r, d)/\mathcal{O}(r)$. We will show in Section 3.4.2 that the Riemannian gradient of the cost function $f(\mathbf{U})$ associated with this metric is given by

$$\mathrm{grad}\, f(\mathbf{U}) = -(\mathbf{I} - \mathbf{U}_t\mathbf{U}_t^T)\mathbf{x}_t\mathbf{x}_t^T\mathbf{U}_t.$$

Combining this gradient with the retraction

$$R_{\mathbf{U}}(-s_t\mathrm{grad}\, f(\mathbf{U})) = \mathrm{qf}(\mathbf{U}_t - s_t\mathrm{grad}\, f(\mathbf{U})),$$

where $\mathrm{qf}(\cdot)$ is a function that extracts the orthogonal factor of the QR-decomposition of its argument, gives us the online learning algorithm

$$\mathbf{U}_{t+1} = \mathrm{qf}(\mathbf{U}_t + s_t(\mathbf{I} - \mathbf{U}_t\mathbf{U}_t^T)\mathbf{x}_t\mathbf{x}_t^T\mathbf{U}_t), \tag{2.27}$$

which is the well-known algorithm of Oja for subspace learning (Oja, 1992).

For centered data $\mathbf{x}_t$, this algorithm globally converges to the dominant subspace of the data covariance matrix (Oja, 1992; Chen et al., 1998). Therefore, it can be regarded as an online PCA algorithm that extracts the p-dominant principal components of the data $\{\mathbf{x}_t\}_{t\geq 0}$.

2.5.4 Linear regression on fixed-rank positive semidefinite matrices

We consider the same setup as for linear regression on positive definite matrices (Section 2.5.2).

Given data $\mathbf{X} \in \mathbb{R}^{d\times d}$, observations $y \in \mathbb{R}_+$, and a linear regression model $\hat{y} = \mathrm{Tr}(\mathbf{W}\mathbf{X})$, the problem amounts to solving the optimization problem

$$\min_{\mathbf{W}\in S_+(r,d)} \mathbb{E}_{\mathbf{X},y}\{f(\mathbf{W})\}, \text{ with } f(\mathbf{W}) = \frac{1}{2}(\mathrm{Tr}(\mathbf{W}\mathrm{Sym}(\mathbf{X})) - y)^2, \tag{2.28}$$

and where

$$S_+(r,d) = \{\mathbf{W} \in \mathbb{R}^{d\times d} : \mathbf{W}^T = \mathbf{W} \succeq 0,\ \mathrm{rank}(\mathbf{W}) = r\}.$$

For $1 \leq r < d$, this problem is non convex because of the rank constraint. It becomes however convex in the case $r = d$ for which $S_+(r,d)$ coincides with $S_{++}(d)$ and (2.28) matches (2.22).

In a recent paper by Kulis et al. (2009), updates (2.23) and (2.24) for linear regression on positive definite matrices are generalized to the case of low-rank semidefinite matrices with a fixed range space. In practice, this means that the subspace of the matrix is fixed beforehand by the initial condition of the algorithm. The resulting optimization problem is convex, but the solution that is eventually obtained is not necessarily related to a minimum of the original cost function because it depends on the heuristic chosen for the initialization.

In contrast to the algorithms of Kulis et al. (2009), the novel algorithms that we will develop in Chapter 4 do not constrain the range space of the learned matrix. In addition, we will show in Chapter 4 how to interpret the algorithms of Kulis et al. (2009) in the proposed framework.

The proposed generalization rests on the adopted geometric optimization viewpoint as we view the learning of a matrix in $S_+(r,d)$ as the simultaneous learning of a r-dimensional subspace in $\mathbb{R}^d$ and of a r-by-r positive definite matrix inside that subspace. This crucial idea is implemented by means of two quotient geometries of $S_+(r,d)$ that have been developed recently.

The first quotient geometry has been developed by Journée et al. (2010). It relies on the fixed-rank factorization $\mathbf{W} = \mathbf{G}\mathbf{G}^T$, where $\mathbf{G} \in \mathbb{R}_*^{d\times r}$. This factorization is not unique because it is defined up to a rotation $\mathbf{G} \mapsto \mathbf{G}\mathbf{O}$, whenever $\mathbf{O} \in \mathcal{O}(r)$. This invariance property leads to a quotient geometry $S_+(r,d) \simeq \mathbb{R}_*^{d\times r}/\mathcal{O}(r)$ where $\mathbf{G}$ can be chosen as any square root of $\mathbf{W}$.

The second quotient geometry has been developed by Bonnabel and Sepulchre (2009). It is obtained by considering the polar factorization $\mathbf{G} = \mathbf{U}\mathbf{R}$, where $\mathbf{U} \in \mathrm{St}(r,d)$ and $\mathbf{R} \in S_{++}(r)$. This factorization always exists and, since $\mathbf{G}$ has full rank, it is unique. The resulting fixed-rank factorization is $\mathbf{W} = \mathbf{U}\mathbf{R}^2\mathbf{U}^T$, which is defined up to a rotation $(\mathbf{U}, \mathbf{R}^2) \mapsto (\mathbf{U}\mathbf{O}, \mathbf{O}^T\mathbf{R}^2\mathbf{O})$, for all $\mathbf{O} \in \mathcal{O}(r)$. This invariance property leads to a quotient geometry $S_+(r,d) \simeq (\mathrm{St}(r,d) \times S_{++}(r))/\mathcal{O}(r)$. By exploiting this geometry, we propose an algorithm that yields separate but coupled iterations for learning the subspace $\mathbf{U}$ and the positive definite scaling $\mathbf{R}^2$.

Alternative geometries for the set $S_+(r,d)$ have also been proposed. Indeed, the set $S_+(r,d)$ admits a embedded geometry as submanifold of $\mathbb{R}^{d\times d}$ (Orsi et al., 2006; Vandereycken and Vandewalle, 2010). The recent work of Vandereycken et al. (2011) presents a geometry of $S_+(r,d)$ as a homogenous space and demonstrates that this geometry can be equipped with complete geodesics available in closed-form. A rigorous comparison between the proposed algorithms and

the algorithms that would result from these geometries is not provided in the present thesis. However, such a comparison is certainly worthwhile and is left as a topic for future research.

The learning of a low-rank distance (Section 2.1) is a particular application of (2.28) with a cost function enforcing inequality constraints. We now present the problem of low-rank distance matrix completion (Fang and Oleary, 2010) that is another application of (2.28).

Low-rank distance matrix completion

A Euclidean distance matrix $\mathbf{D}$ of size n-by-n contains the (squared) pairwise distances between a set of n data points $\mathbf{p}_i \in \mathbb{R}^r$, $i = 1, ..., n$. This matrix is symmetric and has a zero diagonal. Its entries are non-negative and satisfy the triangle inequality. These properties are readily verified by examining the entries of the distance matrix,

$$\mathbf{D}_{ij} = \|\mathbf{p}_i - \mathbf{p}_j\|_2^2.$$

Given a set of pairwise dissimilarities $\widetilde{\mathbf{D}}_{ij} = \widetilde{\mathbf{D}}_{ji} \geq 0$ between n data points and the associated set of indices $(i, j) \in \mathcal{D}$ with $i < j$, distance matrix completion algorithms solve the problem

$$\min_{\mathbf{D} \in \mathrm{EDM}(n)} \|\mathbf{H} \odot (\mathbf{D} - \widetilde{\mathbf{D}})\|_F^2, \tag{2.29}$$

where $\mathrm{EDM}(n)$ is the set of n-by-n Euclidean distance matrices and $\mathbf{H}$ is defined as

$$\mathbf{H}_{ij} = \mathbf{H}_{ji} = \begin{cases} 1 & \text{if } (i, j) \in \mathcal{D}, \\ 0 & \text{otherwise.} \end{cases}$$

Dissimilarities potentially differ from distances in that they are not required to satisfy the triangle inequality. For instance, this takes into account the fact that observation noise could make $\widetilde{\mathbf{D}}$ different from a valid Euclidean distance matrix. The number of elements in the set $\mathcal{D}$ is denoted by $|\mathcal{D}|$. Although, $|\mathcal{D}|$ is at most equal to $n(n-1)/2$, in most applications, it is of order $O(nr^\star)$, where $r^\star$ is the optimal embedding dimension.

A convenient alternative formulation of (2.29) is to cast this problem into an optimization problem on the set of positive semidefinite matrices. The reformulation is given by

$$\min_{\mathbf{W} \in S_+(d)} \|\mathbf{H} \odot (\psi(\mathbf{W}) - \widetilde{\mathbf{D}})\|_F^2, \tag{2.30}$$

where ψ is a mapping from the set of positive semidefinite matrices to the set of Euclidean distance matrices

$$\psi(\mathbf{W}) = \mathrm{Diag}(\mathbf{W})\mathbf{1}_n^T + \mathbf{1}_n \mathrm{Diag}(\mathbf{W})^T - 2\mathbf{W}.$$

The function $\mathrm{Diag}(\cdot)$ extracts the diagonal of its argument, and $\mathbf{1}_n$ denotes a n-dimensional vector with all its entries equal to one. A practical advantage of (2.30) compared to (2.29) is that the rank of $\mathbf{W}$ identifies with the dimension of the embedding space. When no restriction is imposed on the rank of $\mathbf{W}$, problem (2.30) is convex and thus presents a global solution.

However, when the rank of $\mathbf{W}$ is fixed, the problem becomes non-convex and amounts to finding a matrix in the set $S_+(r, d)$. The formulation (2.30) can be equivalently written as

$$\min_{\mathbf{W} \in S_+(r,d)} \sum_{(i,j) \in \mathcal{D}} (\hat{y}_{ij} - y_{ij})^2,$$

where $\hat{y}_{ij} = (\mathbf{e}_i - \mathbf{e}_j)^T \mathbf{W} (\mathbf{e}_i - \mathbf{e}_j)$ and $y_{ij} = \widetilde{\mathbf{D}}_{ij}$. A closely related problem is multidimensional scaling (Borg and Groenen, 2005) for which all pairwise dissimilarities are available up front. The problem can also be regarded as kernel learning problem (Section 2.1.1) for which equality constraints between known pairwise dissimilarities are required.

2.5.5 Linear regression on fixed-rank non-symmetric matrices

Given data $\mathbf{X} \in \mathbb{R}^{d_2 \times d_1}$, observations $y \in \mathbb{R}$, and a linear regression model $\hat{y} = \mathrm{Tr}(\mathbf{W}\mathbf{X})$, linear regression on fixed-rank non-symmetric matrices amounts to solving the optimization problem

$$\min_{\mathbf{W} \in \mathcal{F}(r,d_1,d_2)} \mathbb{E}_{\mathbf{X},y}\{f(\mathbf{W})\}, \text{ with } f(\mathbf{W}) = \frac{1}{2}(\mathrm{Tr}(\mathbf{W}\mathbf{X}) - y)^2. \tag{2.31}$$

The search space is the set of fixed-rank non-symmetric matrices

$$\mathcal{F}(r, d_1, d_2) = \{\mathbf{W} \in \mathbb{R}^{d_1 \times d_2} : \mathrm{rank}(\mathbf{W}) = r\}.$$

In Chapter 5, we will propose novel linear regression algorithms for solving (2.31).

For this purpose, we will develop novel quotient geometries for the set $\mathcal{F}(r, d_1, d_2)$, generalizing the quotient geometries of $S_+(r, d)$ proposed by Bonnabel and Sepulchre (2009) and Journée et al. (2010). The proposed generalization hinges on the fixed-rank factorizations

$$\mathbf{W} = \mathbf{G}\mathbf{H}^T = \mathbf{U}\mathbf{B}\mathbf{V}^T, \tag{2.32}$$

where $\mathbf{G} \in \mathbb{R}_*^{d_1 \times r}$, $\mathbf{H} \in \mathbb{R}_*^{d_2 \times r}$, $\mathbf{U} \in \mathrm{St}(d_1, r)$, $\mathbf{B} \in S_{++}(r)$ and $\mathbf{V} \in \mathrm{St}(d_2, r)$.

We will show in Chapter 5 that the factorizations (2.32) admit intrinsic invariance properties and we will construct the associated quotient manifold geometries of $\mathcal{F}(r, d_1, d_2)$.

Despite the recent interest in the geometries of $S_+(r, d)$, we are only aware of two recent contributions dealing with alternative geometries of $\mathcal{F}(r, d_1, d_2)$. Shalit et al. (2010) propose a Riemannian geometry of $\mathcal{F}(r, d_1, d_2)$ as an embedded submanifold of $\mathbb{R}^{d_1 \times d_2}$. The geometry of Shalit et al. (2010) generalizes the work of Vandereycken and Vandewalle (2010) on the embedded geometry of $S_+(r, d)$ as a submanifold of $\mathbb{R}^{d \times d}$. Simonsson and Eldén (2010) propose a quotient geometry of $\mathcal{F}(r, d_1, d_2)$ that relies on the fixed-rank factorization $\mathbf{W} = \mathbf{U}\mathbf{Z}^T$, where $\mathbf{U} \in \mathrm{St}(d_1, r)$ and $\mathbf{Z} \in \mathbb{R}_*^{d_2 \times r}$. The use of these geometries for the design of rank-constrained linear regression algorithms will be discussed in Chapter 5.

As shown in Section 2.1, low-rank matrix completion is one application of (2.31). In the remainder of this section, we present additional machine learning applications that can be formulated as a linear regression problem on $\mathcal{F}(r, d_1, d_2)$.

Learning on data pairs

Given two types of data $\mathbf{x} \in \mathbb{R}^{d_1}$ and $\mathbf{z} \in \mathbb{R}^{d_2}$ associated with two types of samples, learning on pairs amounts to learning a regression model $\hat{y} : \mathcal{X} \times \mathcal{Z} \to \mathbb{R}$ from training examples $\{(\mathbf{x}_i, \mathbf{z}_i, y_i)\}_{i=1}^n$. When the chosen regression model $\hat{y}$ is the fixed-rank bilinear form $\hat{y} = \mathbf{x}^T\mathbf{W}\mathbf{z}$, then the problem boils down to the considered regression problem with the choice $\mathbf{X} = \mathbf{z}\mathbf{x}^T$.

A first application of this general setting is the inference of edges in bipartite or directed graphs. Such a problem arises in bioinformatics for the identification of interactions between drugs and target proteins (Yamanishi et al., 2008; Bleakley and Yamanishi, 2009), micro-RNA and genes or genes and diseases. Let $\mathcal{G} = (U, V, E)$ be a bipartite graph with two sets of nodes U, V and a set of edges E that connect the nodes in U to the nodes in V. Suppose that data is available for the nodes in U and the nodes in V. Suppose also that a similarity score between nodes is given along with some edges in E. The graph inference problem amounts to discovering new edges that are consistent with both the available data and the known interconnections.

For a similarity score between edges as $\hat{y} = \mathbf{x}^T\mathbf{W}\mathbf{z}$, this problem boils down to the considered regression framework. Extension to directed graphs can be seen as a particular case of bipartite graphs for which $U = V$. In that case, the parameter is a matrix $\mathbf{W} \in \mathbb{R}^{d \times d}$ with $d = |U| = |V|$.

A second application of this setting is domain adaptation using asymmetric transforms. In real-world applications, the type and dimension of data available for a given problem often

changes from one data set to the next. A model learned on a given data source domain might not be applicable to data that belong a different target domain. For instance, source and target domains might not share the same feature space or dimensionality. Domain adaptation algorithms address this issue by learning a transformation that transfers object models from one domain to another. This problem has been recently studied by Kulis et al. (2011) in the context of image domain adaptation for classification purposes.

The algorithm proposed by Kulis et al. (2011) learns a non-symmetric transformation $\mathbf{x}^T\mathbf{W}\mathbf{z}$ between labeled images $\mathbf{x}$ from the source domain $\mathcal{X}$ and labeled images $\mathbf{z}$ from the target domain $\mathcal{Z}$. The transformation $\mathbf{W}$ is then used on new input data to map the data from one domain to the other. This problem can be directly addressed in the proposed linear regression framework as a learning on data pairs problem. Imposing a low-rank constraint on the transformation $\mathbf{W}$ has potential interest in the context of high-dimensional problems and can be regarded as a means to simultaneously perform dimensionality reduction on the two domains.

Multi-task regression

Multi-task regression is a learning framework where several regression problems are solved simultaneously using a unique regression parameter shared between the different problems. More precisely, suppose that P regression problems have to be solved and that each problem consists in learning a linear regression model $\hat{y}_p = \mathbf{w}_p^T\mathbf{x}_{pi}$ over n_p examples $\{(\mathbf{x}_{pi}, y_{pi})\}_{i=1}^{n_p}$, with $\mathbf{x}_{pi}, \mathbf{w}_p \in \mathbb{R}^d$, $y_{pi} \in \mathbb{R}$, and $p = 1, ..., P$. A convenient way of constructing a global parameter is to stack the vectors $\mathbf{w}_p$ into a matrix $\mathbf{W} \in \mathbb{R}^{P\times d}$. The regression model for the p-th problem is

$$\hat{y}_{pi} = \mathrm{Tr}(\mathbf{W}\mathbf{e}_p\mathbf{x}_{pi}^T) = \mathbf{w}_p^T\mathbf{x}_{pi},$$

where $\mathbf{e}_p$ is a canonical basis vector of $\mathbb{R}^P$. The regression problem can then be formulated as

$$\min_{\mathbf{W}\in\mathcal{F}(r,P,d)} \mathbb{E}_{\mathbf{x},y}\{\ell(\hat{y}, y)\} + \gamma J(\mathbf{W}).$$

The function $J : \mathbb{R}^{P\times d} \to \mathbb{R}$ enforces prior knowledge on the dependence between parameters of the different regression problems. A possible choice for $J(\mathbf{W})$ is

$$J(\mathbf{W}) = \mathrm{Tr}(\mathbf{W}^T\mathbf{C}\mathbf{W}),$$

where $\mathbf{C}$ is positive definite. For example, the choice $\mathbf{C} = \mathbf{I}$ means that the task parameters are learned independently, and one recovers the classical regression setting. Other coupling functions are presented in Evgeniou et al. (2005). Besides, constraining the rank of $\mathbf{W}$ enforces a decomposition of the problems into a limited number of factors (Argyriou et al., 2007).

Learning to rank with a non-symmetric similarity measure

Ranking problems can be cast as regression problems over a similarity or relevance measure between data (Chechik et al., 2010; Shalit et al., 2010). The goal of this problem is to compute a similarity score $\hat{y} : \mathbb{R}^d \times \mathbb{R}^d \to \mathbb{R}$ between data inputs $\mathbf{x}_i, \mathbf{x}_j \in \mathbb{R}^d$.

The following approach has been taken by Chechik et al. (2010) and Shalit et al. (2010). Given a query point $\mathbf{x}_q$, ranking the data that are mostly related to $\mathbf{x}_q$ is performed by sorting the scores $\hat{y}(\mathbf{x}_q, \cdot)$ in descending order of magnitude. The relevance score is constructed from triplets $(\mathbf{x}_i, \mathbf{x}_i^+, \mathbf{x}_i^-)$ in order to impose $y(\mathbf{x}_i, \mathbf{x}_i^+) > y(\mathbf{x}_i, \mathbf{x}_i^-)$, whenever $\mathbf{x}_i^+$ is more relevant to $\mathbf{x}_i$ than $\mathbf{x}_i^-$. The resulting regression problem can be formulated as

$$\min_{\mathbf{W}\in\mathcal{F}(r,d,d)} \mathbb{E}_{(\mathbf{x}_i,\mathbf{x}_i^+,\mathbf{x}_i^-)}\{\ell(\hat{y}(\mathbf{x}_i, \mathbf{x}_i^+), \hat{y}(\mathbf{x}_i, \mathbf{x}_i^-))\}.$$

where the considered regression model is again the quadratic form $\hat{y}(\mathbf{x}_i, \mathbf{x}_j) = \mathbf{x}_i^T\mathbf{W}\mathbf{x}_j$, while the loss function enforces the constraint $y(\mathbf{x}_i, \mathbf{x}_i^+) > y(\mathbf{x}_i, \mathbf{x}_i^-)$.

The main motivation for introducing the rank constraint in this problem is to reduce the computational cost of the learning algorithm and to tackle high-dimensional problems.

Table 2.1: Summary of the connections with previous works

Regression on	Search space	Geometry	Metric	Connection with	See Section(s)
Positive vectors	$\mathbf{w} > 0$	Log-Euclidean	Euclidean	Boosting (Freund and Schapire, 1997)	2.4.2
Subspaces	$\mathrm{Gr}(r, d)$	Quotient	Euclidean	Subspace learning (Oja, 1992)	2.5.3, 3.4.2
Orthogonal matrices	$\mathcal{O}(d)$	Embedded	Euclidean	Learning of rotations (Arora, 2009)	2.5.1, 3.4.1
Positive definite matrices	$S_{++}(d)$	Embedded	Affine-invariant	Log-Det divergence (Davis et al., 2007)	4.2.2, 4.5.1
Positive definite matrices	$S_{++}(d)$	Log-Euclidean	Euclidean	von Neumann divergence (Tsuda et al., 2005)	4.2.3, 4.5.1
Fixed-rank non-symm. matrices	$\mathcal{F}(r, d_1, d_2)$	Quotient: $\mathbf{GH}^T$	Equation (5.7)	MMMF (Rennie and Srebro, 2005)	5.4.1
Fixed-rank non-symm. matrices	$\mathcal{F}(r, d_1, d_2)$	Quotient: $\mathbf{UBV}^T$	Equation (5.11)	SVP (Jain et al., 2010)	5.4.2
				OptSpace (Keshavan et al., 2010)	5.4.2

Chapter 3

A geometric optimization framework for regression

Chapter abstract: This chapter introduces the key concepts and notations related to optimization on matrix manifolds. The material presented in this chapter will be exploited in subsequent chapters to develop novel linear regression algorithms for fixed-rank matrices.

Section 3.1 presents the main ideas behind optimization on Riemannian matrix manifolds and introduces the concepts of embedded and quotient manifolds. Section 3.2 defines the tangent space of an embedded manifold. Section 3.3 defines the tangent space of a quotient manifold. Section 3.4 presents a systematic procedure to develop line-search algorithms on matrix manifolds. Section 3.5 presents the analogous procedure for trust-region algorithms.

We refer to Absil et al. (2008) for a more complete exposition and a state-of-the-art in this area.

3.1 Optimization on Riemannian matrix manifolds

Classical optimization algorithms generally deal with structured matrix search spaces by means of explicit constraints or penalty terms expressed as a function of the decision variable. This is the approach taken by many classical constrained optimization techniques such as penalty methods, barrier methods or augmented Lagrangian methods (Nocedal and Wright, 2006). These methods all turn a constrained optimization problem into a sequence of unconstrained optimization problems for which classical unconstrained optimization techniques can be applied.

An alternative approach is to embed the constraints into the search space and to solve an unconstrained optimization problem on the constrained search space. This alternative approach is the purpose of optimization algorithms defined on matrix manifolds (Absil et al., 2008). In a nutshell, a manifold $\mathcal{W}$ is a set endowed with a differentiable structure. Intuitively, a manifold can be regarded as a smoothly curved space. Locally, this space looks like a Euclidean space, but its global structure is possibly much richer. Once a manifold is endowed with a differentiable structure, computations can be performed, and the classical geometric objects of optimization such as derivatives, gradients or Hessians are naturally extended to manifolds.

Our focus will be on embedded submanifolds and quotient manifolds. Embedded submanifolds can be viewed as a generalization of the notion of surface in $\mathbb{R}^d$. They are defined by means of an explicit set of algebraic constraints in matrix space $\mathbb{R}^{d\times r}$. This general concept applies straight to the Stiefel manifold

$$\mathrm{St}(r,d) = \{\mathbf{U} \in \mathbb{R}^{d\times r} : \mathbf{U}^T\mathbf{U} = \mathbf{I}\},$$

which is a submanifold embedded in $\mathbb{R}^{d\times r}$. Another example is the d-dimensional unit sphere $\mathcal{S}^{d-1}$ embedded in $\mathbb{R}^d$, which coincides with $\mathrm{St}(1,d)$. The set $\mathcal{O}(d)$ of d-by-d orthogonal matrices is an embedded submanifold of the vector space of square real matrices and coincides with $\mathrm{St}(d,d)$. A subset of a manifold does not automatically define an embedded submanifold. To define an embedded submanifold, the subset of interest must be equipped with a differential

structure that is compatible with the differential structure of its embedding space. The reader is referred to Lee (2003); Absil et al. (2008) for a general definition of an embedded submanifold.

The concept of quotient manifold is more abstract. Each point of the quotient manifold is defined as an equivalence class of matrices. Since these equivalence classes are abstract objects, they cannot be explicitly used in numerical computations. Optimization algorithms on quotient manifolds work instead with representatives of these equivalence classes. Geometric objects on the quotient manifold can be defined by means of these representatives, provided their definition do not depend on a particular choice for the representative within an equivalence class. This general concept applies to the Grassmann manifold $\mathrm{Gr}(r, d)$, that is, the set of r-dimensional subspaces in $\mathbb{R}^d$. Indeed, each subspace can be defined by a r-dimensional orthogonal frame up to a rotation. The rotational variants of a given frame thus define an equivalence class which is identified as a single point on the quotient manifold. Another example is the real projective space, representing the set of straight lines passing through the origin. This space coincides with $\mathrm{Gr}(1, d)$. The real projective space can be regarded as a collection of vectors in $\mathbb{R}^d_* \triangleq \mathbb{R}^d \backslash \{0\}$ that are defined up to a change of length. Any two vectors $\mathbf{a} \in \mathbb{R}^d_*$ and $\mathbf{b} \in \mathbb{R}^d_*$ belong to the same equivalence class whenever $\mathbf{a} = \rho\mathbf{b}$ for $\rho \neq 0$. As with embedded submanifolds, a careful choice of the differentiable structure accompanying the quotient space is necessary to yield a quotient manifold. When each equivalence class is generated by the action of a group, the *quotient manifold theorem* (Lee, 2003, Theorem 9.16) states the conditions under which the group action yields a valid quotient manifold.

3.2 Tangent space of an embedded submanifold

We will only consider the simpler and prevalent case of submanifolds embedded in the Euclidean space $\mathbb{R}^{d\times r}$, which is sufficient for the needs of the present dissertation.

Consider a manifold $\mathcal{W}$ embedded in $\mathbb{R}^{d\times r}$, a point $\mathbf{W} \in \mathcal{W}$, and a smooth curve $\gamma : \mathbb{R} \to \mathcal{W}$ passing through $\gamma(0) = \mathbf{W}$. The tangent vector to γ at $\mathbf{W}$ is the matrix in $\mathbb{R}^{d\times r}$ defined as

$$\dot{\gamma}(0) = \lim_{t\to 0} \frac{\gamma(t) - \gamma(0)}{t}. \tag{3.1}$$

Since γ is included in $\mathcal{W}$, $\dot{\gamma}(0)$ defines a tangent vector $\xi_{\mathbf{W}}$ to the manifold.

The set of all tangent vectors $\xi_{\mathbf{W}}$ at $\mathbf{W}$ forms the tangent space $T_{\mathbf{W}}\mathcal{W}$ of $\mathcal{W}$ at $\mathbf{W}$. The tangent space thus admits a natural identification with the set

$$\{\dot{\gamma}(0) : \gamma \text{ is a curve in } \mathcal{W} \text{ with } \gamma(0) = \mathbf{W}\}. \tag{3.2}$$

The definition of tangent vectors to embedded submanifolds of $\mathbb{R}^{d\times r}$ is illustrated in Figure 3.1.

Tangent space of the Stiefel manifold

Let $\mathbf{U}(t)$ be a curve in $\mathrm{St}(r, d)$ passing through $\mathbf{U}_0$ at $t = 0$. We have

$$\mathbf{U}(t)^T\mathbf{U}(t) = \mathbf{I} \quad \forall t,$$

whose differential with respect to t yields

$$\dot{\mathbf{U}}(t)^T\mathbf{U}(t) + \mathbf{U}(t)^T\dot{\mathbf{U}}(t) = 0. \tag{3.3}$$

Since a tangent vector $\dot{\mathbf{U}}(t)$ can be represented as a matrix in $\mathbb{R}^{d\times r}$, it can be decomposed into

$$\dot{\mathbf{U}}(t) = \mathbf{U}(t)\boldsymbol{\Omega}(t) + \mathbf{U}_\perp(t)\mathbf{K}(t), \tag{3.4}$$

where $\boldsymbol{\Omega}(t) \in \mathbb{R}^{r\times r}$, $\mathbf{K}(t) \in \mathbb{R}^{(d-r)\times r}$ and $\mathbf{U}_\perp(t)$ is an orthonormal matrix that spans the orthogonal complement of $\mathbf{U}(t)$, that is, $\mathbf{U}_\perp(t)^T\mathbf{U}(t) = 0$ and $\mathbf{U}_\perp(t)^T\mathbf{U}_\perp(t) = \mathbf{I}$.

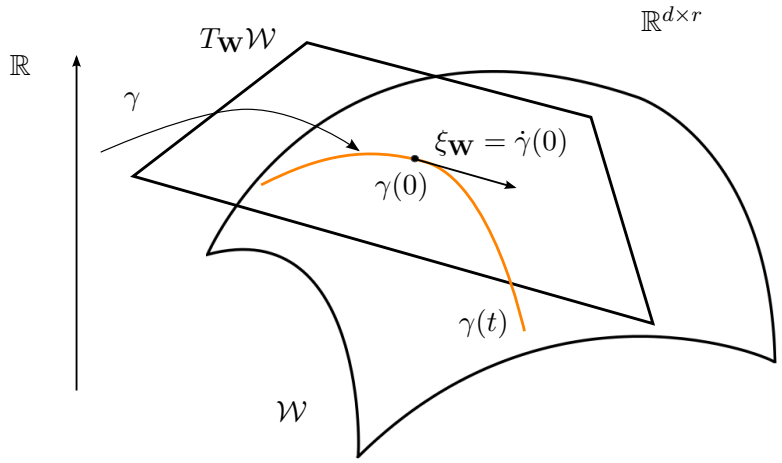


Figure 3.1: For a submanifold $\mathcal{W}$ embedded in $\mathbb{R}^{d\times r}$, a tangent vector $\xi_{\mathbf{W}} \in T_{\mathbf{W}}\mathcal{W}$ at $\mathbf{W}$ is defined as a tangent vector $\dot\gamma(0)$ to a curve passing through $\gamma(0) = \mathbf{W}$.

Substituting (3.4) into (3.3) yields the conditions

$$\mathbf{\Omega}(t)^T + \mathbf{\Omega}(t) = 0 \quad \text{and} \quad \mathbf{K}(t) \in \mathbb{R}^{(d-r)\times r},$$

that is, $\mathbf{\Omega}(t) \in \mathbb{R}^{r\times r}$ is skew-symmetric and $\mathbf{K}(t) \in \mathbb{R}^{(d-r)\times r}$ is not constrained. Therefore, the tangent space at $\mathbf{U}_0$ corresponds to the set

$$T_{\mathbf{U}_0}\mathrm{St}(r,d) = \{\mathbf{U}_0\mathbf{\Omega} + \mathbf{U}_{0,\perp}\mathbf{K} : \mathbf{\Omega}^T = -\mathbf{\Omega} \in \mathbb{R}^{r\times r},\ \mathbf{K} \in \mathbb{R}^{(d-r)\times r}\}. \tag{3.5}$$

A tangent vector to $\mathrm{St}(r,d)$ at a given point $\mathbf{U}_0$ thus has $r(r-1)/2+(d-r)r$ degrees of freedom. This number of degrees of freedom matches $\dim(\mathrm{St}(r,d)) = dr - r(r+1)/2$, the number of the Stiefel manifold itself, that can be decomposed as the dimension of the embedding space $\mathbb{R}^{d\times r}$ minus the number of orthogonality constraints imposed by $\mathbf{U}_0^T\mathbf{U}_0 = \mathbf{I}$.

Tangent space of the set of orthogonal matrices

As the set $\mathcal{O}(d)$ coincides with $\mathrm{St}(d,d)$, we have that the tangent space to $\mathcal{O}(d)$ at $\mathbf{Q}$ is given by

$$T_{\mathbf{Q}}\mathcal{O}(d) = \{\mathbf{Q}\mathbf{\Omega} : \mathbf{\Omega}^T = -\mathbf{\Omega} \in \mathbb{R}^{r\times r}\}. \tag{3.6}$$

Tangent space of the sphere

The tangent space to the d-dimensional sphere

$$\mathcal{S}^{d-1} = \{\mathbf{w} \in \mathbb{R}^d : \|\mathbf{w}\| = 1\},$$

at point $\mathbf{w}$ is obtained from the equivalence between $\mathcal{S}^{d-1}$ and $\mathrm{St}(1,d)$. We have

$$T_{\mathbf{w}}\mathcal{S}^{d-1} = \{\mathbf{z} \in \mathbb{R}^d : \mathbf{w}^T\mathbf{z} = 0\}.$$

This is in accordance with intuition: the most effective directions for moving on a sphere do not contain a component which is normal to the sphere.

3.3 Tangent space of a quotient manifold

Let $\overline{\mathcal{W}}$ be a manifold, either $\mathbb{R}^{d\times r}$ or a embedded submanifold of $\mathbb{R}^{d\times r}$, that is equipped with an equivalence relation $\sim$ (symmetric, reflexive and transitive). The equivalence class (or *fiber*) of a given point $\mathbf{W}$ is defined by the set

$$[\mathbf{W}] = \{\mathbf{X} \in \overline{\mathcal{W}} : \mathbf{X} \sim \mathbf{W}\}.$$

By extension, the set

$$\overline{\mathcal{W}}/\sim \triangleq \{[\mathbf{W}] : \mathbf{W} \in \overline{\mathcal{W}}\},$$

is a quotient manifold of $\overline{\mathcal{W}}$ by $\sim$. The mapping $\pi : \overline{\mathcal{W}} \to \overline{\mathcal{W}}/\sim: \mathbf{W} \mapsto [\mathbf{W}]$ is called the quotient map or canonical projection. Clearly, we have $\pi(\mathbf{W}) = \pi(\mathbf{X})$ if and only if $\mathbf{W} \sim \mathbf{X}$. The set $\overline{\mathcal{W}}$ is the *total space* of the quotient $\overline{\mathcal{W}}/\sim$. These concepts are illustrated in Figure 3.2.

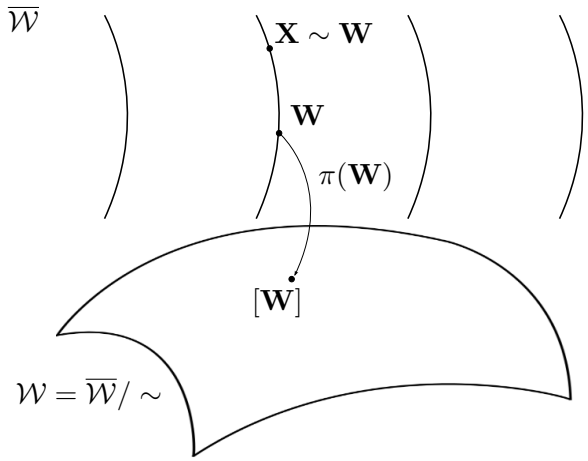


Figure 3.2: Quotient manifolds are defined such that equivalent points $\mathbf{X} \sim \mathbf{W}$ in the total space $\overline{\mathcal{W}}$ correspond to a single point $[\mathbf{W}] = \pi(\mathbf{W})$ on the quotient space $\overline{\mathcal{W}}/\sim$.

For example, following Edelman et al. (1998), the Grassmann manifold $\mathrm{Gr}(r, d)$ admits the quotient manifold representation

$$\mathrm{Gr}(r, d) \simeq \mathrm{St}(r, d)/\mathcal{O}(r). \tag{3.7}$$

It is based on the following equivalence relation: two orthogonal frames $\mathbf{U}_1, \mathbf{U}_2 \in \mathrm{St}(r, d)$ are such that $\mathrm{range}(\mathbf{U}_1) = \mathrm{range}(\mathbf{U}_2)$ whenever they are related by a rotation,

$$\mathbf{U}_1 \sim \mathbf{U}_2 \quad \Leftrightarrow \quad \exists \mathbf{Q} \in \mathcal{O}(r) : \mathbf{U}_1 = \mathbf{U}_2\mathbf{Q}. \tag{3.8}$$

The corresponding set of equivalence classes is

$$[\mathbf{U}] = \{\mathbf{U}\mathbf{Q} : \mathbf{Q} \in \mathcal{O}(r)\}. \tag{3.9}$$

The reader is referred to the papers of Edelman et al. (1998) or Absil et al. (2004) for alternative characterizations of the Grassmann manifold $\mathrm{Gr}(r, d)$ as a quotient manifold.

Tangent vectors to quotient manifolds are obtained from tangent vectors in the total space. Indeed, for a quotient manifold $\mathcal{W} = \overline{\mathcal{W}}/\sim$, a tangent vector $\xi_{[\mathbf{W}]} \in T_{[\mathbf{W}]}\mathcal{W}$ at $[\mathbf{W}]$ is restricted to the directions that do not induce a displacement along the set of equivalence classes $[\mathbf{W}]$.

This is achieved by decomposing the tangent space in the total space $T_{\mathbf{W}}\overline{\mathcal{W}}$ into complementary spaces

$$T_{\mathbf{W}}\overline{\mathcal{W}} = \mathcal{V}_{\mathbf{W}}\mathcal{W} \oplus \mathcal{H}_{\mathbf{W}}\mathcal{W}.$$

The *vertical space* $\mathcal{V}_{\mathbf{W}}\mathcal{W}$ is the set of directions that contains tangent vectors to the equivalence classes. The *horizontal space* $\mathcal{H}_{\mathbf{W}}\mathcal{W}$ is a complement of $\mathcal{V}_{\mathbf{W}}\mathcal{W}$ in $T_{\mathbf{W}}\overline{\mathcal{W}}$.

With such a decomposition, any element $\xi_{\mathbf{W}} \in T_{\mathbf{W}}\overline{\mathcal{W}}$ can be decomposed into

$$\xi_{\mathbf{W}} = P^{\mathcal{V}}_{\mathbf{W}}(\xi_{\mathbf{W}}) + P^{\mathcal{H}}_{\mathbf{W}}(\xi_{\mathbf{W}}), \quad \text{where} \quad P^{\mathcal{V}}_{\mathbf{W}}(\xi_{\mathbf{W}}) \in \mathcal{V}_{\mathbf{W}}\mathcal{W} \text{ and } P^{\mathcal{H}}_{\mathbf{W}}(\xi_{\mathbf{W}}) \in \mathcal{H}_{\mathbf{W}}\mathcal{W}.$$

The horizontal space $\mathcal{H}_{\mathbf{W}}\mathcal{W}$ provides a representation of tangent vectors to the quotient space. Indeed, displacements in the vertical space leave the point $\mathbf{W}$ unchanged, which suggests to restrict tangent vectors $\xi_{[\mathbf{W}]}$ to the horizontal space. This yields $T_{[\mathbf{W}]}\mathcal{W} \triangleq \mathcal{H}_{\mathbf{W}}\mathcal{W}$.

Once $\overline{\mathcal{W}}$ is endowed with a horizontal distribution $\mathcal{H}_{\mathbf{W}}\mathcal{W}$, a given tangent vector $\xi_{[\mathbf{W}]} \in T_{[\mathbf{W}]}\mathcal{W}$ at $[\mathbf{W}]$ is uniquely represented by a *horizontal tangent vector* $\bar{\xi}_{\mathbf{W}} \in \mathcal{H}_{\mathbf{W}}\mathcal{W}$ that satisfies

$$D\pi(\mathbf{W})[\bar{\xi}_{\mathbf{W}}] = \xi_{[\mathbf{W}]}.$$

The tangent vector $\bar{\xi}_{\mathbf{W}} \in \mathcal{H}_{\mathbf{W}}\mathcal{W}$ is called the *horizontal lift* of $\xi_{[\mathbf{W}]}$ at $\mathbf{W}$. Although a tangent vector to the quotient $\xi_{[\mathbf{W}]}$ is an abstract geometric object, its horizontal lift $\bar{\xi}_{\mathbf{W}}$ lends itself to representation in a computer as a matrix array.

The concept of tangent space of a quotient manifold is illustrated in Figure 3.3. The computation of the tangent space to the Grassmann manifold is deferred to Section 3.4.2.

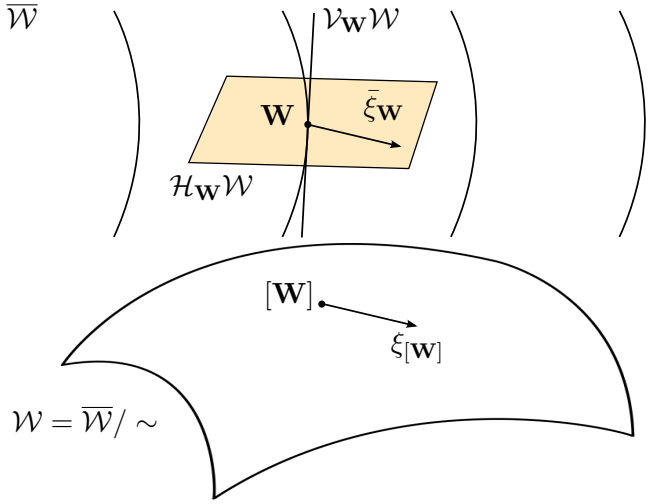


Figure 3.3: Tangent vectors to quotient manifolds are represented by their horizontal lift.

3.4 Line-search algorithms on matrix manifolds

A line-search algorithm on a manifold $\mathcal{W}$ is based on the update formula

$$\mathbf{W}_{t+1} = R_{\mathbf{W}_t}(s_t \xi_{\mathbf{W}_t}), \tag{3.10}$$

where the search direction $\xi_{\mathbf{W}_t}$ is an element of the tangent space $T_{\mathbf{W}_t}\mathcal{W}$ at $\mathbf{W}_t$ and $s_t > 0$ is the step size. The update is performed by means of the *retraction mapping* $R_{\mathbf{W}_t}$.

A first-order retraction $R_{\mathbf{W}} : T_{\mathbf{W}_t}\mathcal{W} \to \mathcal{W}$ is a local update mapping from the tangent space to the manifold which satisfies the following two conditions:

1. $R_{\mathbf{W}}(0) = \mathbf{W}$,
2. $\forall \xi_{\mathbf{W}} \in T_{\mathbf{W}}\mathcal{W}$, the curve $\gamma : t \mapsto R_{\mathbf{W}}(t\xi_{\mathbf{W}})$ realizes $\xi_{\mathbf{W}}$ at $\mathbf{W}$, that is, $\dot{\gamma}(0) = \xi_{\mathbf{W}}$.

Gradient descent algorithms on manifolds are a particular case of (3.10) for which $\xi_{\mathbf{W}_t}$ coincides with the gradient of the considered cost function.

To define the notion of gradient on the manifold, the manifold must be first endowed with a metric $g_{\mathbf{W}}(\cdot,\cdot)$, which is an inner product between any two elements $\xi_{\mathbf{W}}, \zeta_{\mathbf{W}} \in T_{\mathbf{W}}\mathcal{W}$. The metric induces a norm on $T_{\mathbf{W}}\mathcal{W}$,

$$\|\xi_{\mathbf{W}}\|_{\mathbf{W}} = \sqrt{g_{\mathbf{W}}(\xi_{\mathbf{W}}, \xi_{\mathbf{W}})}.$$

A manifold $\mathcal{W}$ endowed with such a smoothly varying metric is called a *Riemannian manifold*. The corresponding metric is called the *Riemannian metric*.

Let $f : \mathcal{W} \to \mathbb{R}$ be a smooth cost function on the manifold. The *Riemannian gradient* of f according to the chosen metric is defined as the unique element $\operatorname{grad} f(\mathbf{W}) \in T_{\mathbf{W}}\mathcal{W}$ that satisfies

$$Df(\mathbf{W})[\xi_{\mathbf{W}}] = g_{\mathbf{W}}(\operatorname{grad} f(\mathbf{W}), \xi_{\mathbf{W}}), \quad \forall \xi_{\mathbf{W}} \in T_{\mathbf{W}}\mathcal{W}, \tag{3.11}$$

where the quantity $Df(\mathbf{W})[\xi_{\mathbf{W}}]$ is the directional derivative of $f(\mathbf{W})$ in the direction $\xi_{\mathbf{W}}$,

$$Df(\mathbf{W})[\xi_{\mathbf{W}}] = \lim_{t \to 0} \frac{f(\mathbf{W} + t\xi_{\mathbf{W}}) - f(\mathbf{W})}{t}.$$

Natural displacements on the manifold are performed by following geodesics (paths of shortest length on the manifold) starting from $\mathbf{W} \in \mathcal{W}$ and tangent to $\xi_{\mathbf{W}} \in T_{\mathbf{W}}\mathcal{W}$. However, in most cases, the geodesics are expensive to compute or are not available in closed-form.

A more general update mapping is obtained if we relax the constraint of moving along geodesics. The retraction $R_{\mathbf{W}_t}(s_t\xi_{\mathbf{W}_t})$, locally approximates the geodesics and provides an attractive alternative to the geodesics in the design of optimization algorithms on manifolds, as it reduces the computational complexity of the update while retaining the essential properties that ensure convergence results. Line-search algorithms on manifolds come with a well-characterized convergence theory (see Absil et al., 2008, Theorem 4.3.1).

3.4.1 Line-search algorithms on embedded manifolds

Let $\mathcal{W}$ be a submanifold embedded in the Euclidean space $\mathbb{R}^{d\times r}$. Since the tangent space $T_{\mathbf{W}}\mathcal{W}$ can be regarded as a subspace of $\mathbb{R}^{d\times r}$, a Riemannian metric defined on $\mathbb{R}^{d\times r}$ induces a Riemannian metric on $\mathcal{W}$, and turns $\mathcal{W}$ into a *Riemannian submanifold* of $\mathbb{R}^{d\times r}$.

At a given point $\mathbf{W} \in \mathcal{W}$, the space $\mathbb{R}^{d\times r}$ can be decomposed into

$$\mathbb{R}^{d\times r} = T_{\mathbf{W}}\mathcal{W} \oplus N_{\mathbf{W}}\mathcal{W},$$

where $N_{\mathbf{W}}\mathcal{W}$ is the *normal space*, that is, the space that is orthogonal to the tangent space $T_{\mathbf{W}}\mathcal{W}$ according to the chosen metric $g_{\mathbf{W}}(\cdot,\cdot)$. Any element $\xi \in \mathbb{R}^{d\times r}$ can be decomposed into

$$\xi = P_{\mathbf{W}}(\xi) + P^{\perp}_{\mathbf{W}}(\xi),$$

where $P_{\mathbf{W}}(\xi) \in T_{\mathbf{W}}\mathcal{W}$ and $P^{\perp}_{\mathbf{W}}(\xi) \in N_{\mathbf{W}}\mathcal{W}$ are respectively the orthogonal projections of ξ onto the tangent space $T_{\mathbf{W}}\mathcal{W}$ and the normal space $N_{\mathbf{W}}\mathcal{W}$ (Figure 3.4).

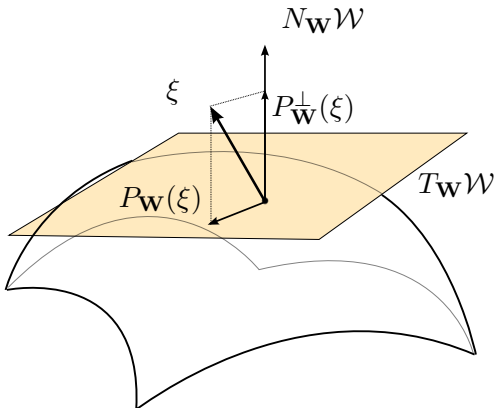


Figure 3.4: For a submanifold $\mathcal{W}$ embedded in $\mathbb{R}^{d\times r}$, a given $\xi \in \mathbb{R}^{d\times r}$ has a component in the tangent space $T_{\mathbf{W}}\mathcal{W}$ and a component in the normal space $N_{\mathbf{W}}\mathcal{W}$.

Given a function $\bar{f} : \mathbb{R}^{d\times r} \to \mathbb{R}$, and given $f = \bar{f}|_{\mathcal{W}} : \mathcal{W} \to \mathbb{R}$ the restriction of $\bar{f}$ to the submanifold $\mathcal{W}$, the Riemannian gradient of f is obtained thanks to the convenient formula

$$\mathrm{grad} f(\mathbf{W}) = P_{\mathbf{W}}(\mathrm{grad}\bar{f}(\mathbf{W})). \tag{3.12}$$

When the chosen metric coincides with the Euclidean metric $g_{\mathbf{W}}(\xi_{\mathbf{W}}, \zeta_{\mathbf{W}}) = \mathrm{Tr}(\xi_{\mathbf{W}}^T \zeta_{\mathbf{W}})$, the gradient in the embedding space $\mathrm{grad}\bar{f}(\mathbf{W})$ identifies with the usual Euclidean gradient,

$$\mathrm{grad}\bar{f}(\mathbf{W}) = \left.\frac{\partial \bar{f}}{\partial \mathbf{W}_{ij}}\right|_{\mathbf{W}} = \nabla_{\mathbf{W}}\bar{f}(\mathbf{W}).$$

Line-search on the Stiefel manifold

The tangent space to the Stiefel manifold

$$T_{\mathbf{U}}\mathrm{St}(r,d) = \{\mathbf{U}\boldsymbol{\Omega} + \mathbf{U}_\perp \mathbf{K} : \boldsymbol{\Omega}^T = -\boldsymbol{\Omega} \in \mathbb{R}^{r\times r},\ \mathbf{K} \in \mathbb{R}^{(d-r)\times r}\}$$

is endowed with the Euclidean metric $g_{\mathbf{U}}(\xi_{\mathbf{U}}, \zeta_{\mathbf{U}}) = \mathrm{Tr}(\xi_{\mathbf{U}}^T \zeta_{\mathbf{U}})$. The associated normal space is

$$N_{\mathbf{U}}\mathrm{St}(r,d) = \{\mathbf{U}\mathbf{S} : \mathbf{S}^T = \mathbf{S} \in \mathbb{R}^{r\times r}\},$$

which is obtained by considering the property $\mathrm{Tr}(\mathbf{S}\boldsymbol{\Omega}) = 0$ that holds for any symmetric matrix $\mathbf{S}$ and skew-symmetric matrix $\boldsymbol{\Omega}$. The projections are given by

$$P_{\mathbf{U}}(\xi) = (\mathbf{I} - \mathbf{U}\mathbf{U}^T)\xi + \mathbf{U}\mathrm{Skew}(\mathbf{U}^T\xi) \quad \text{and} \quad P_{\mathbf{U}}^{\perp}(\xi) = \mathbf{U}\mathrm{Sym}(\mathbf{U}^T\xi).$$

For a cost function $\bar{f} : \mathbb{R}^{d\times r} \to \mathbb{R}$ that induces a cost function $f : \mathrm{St}(r,d) \to \mathbb{R}$ on the Stiefel manifold, the Riemannian gradient is given by

$$\mathrm{grad}\, f(\mathbf{U}) = P_{\mathbf{U}}(\mathrm{grad}\, \bar{f}(\mathbf{U})) = P_{\mathbf{U}}(\nabla_{\mathbf{U}}\bar{f}(\mathbf{U}))$$

An efficient retraction is given by

$$R_{\mathbf{U}}(\xi_{\mathbf{U}}) = \mathrm{qf}(\mathbf{U} + \xi_{\mathbf{U}}).$$

Alternative choices for the retraction $R_{\mathbf{U}}$ are possible (Edelman et al., 1998; Absil et al., 2008).

Line-search on the set of orthogonal matrices

The tangent space to the set of orthogonal matrices $\mathcal{O}(d)$

$$T_{\mathbf{Q}}\mathcal{O}(d) = \{\mathbf{Q}\boldsymbol{\Omega} : \boldsymbol{\Omega}^T = -\boldsymbol{\Omega} \in \mathbb{R}^{d\times d}\}$$

is endowed with the Euclidean metric $g_{\mathbf{Q}}(\xi_{\mathbf{Q}}, \zeta_{\mathbf{Q}}) = \mathrm{Tr}(\xi_{\mathbf{Q}}^T \zeta_{\mathbf{Q}})$. The associated normal space is

$$N_{\mathbf{Q}}\mathcal{O}(d) = \{\mathbf{Q}\mathbf{S} : \mathbf{S}^T = \mathbf{S} \in \mathbb{R}^{d\times d}\},$$

and the projections onto $T_{\mathbf{Q}}\mathcal{O}(d)$ and $N_{\mathbf{Q}}\mathcal{O}(d)$ are respectively given by

$$P_{\mathbf{Q}}(\xi) = \mathbf{Q}\mathrm{Skew}(\mathbf{Q}^T\xi) \quad \text{and} \quad P_{\mathbf{Q}}^{\perp}(\xi) = \mathbf{Q}\mathrm{Sym}(\mathbf{Q}^T\xi).$$

For a cost function $\bar{f} : \mathbb{R}^{d\times r} \to \mathbb{R}$ that induces a cost function $f : \mathcal{O}(d) \to \mathbb{R}$ on the set of rotation matrices, the Riemannian gradient is given by

$$\mathrm{grad}\, f(\mathbf{Q}) = P_{\mathbf{Q}}(\mathrm{grad}\, \bar{f}(\mathbf{Q})) = P_{\mathbf{Q}}(\nabla_{\mathbf{Q}}\bar{f}(\mathbf{Q})).$$

We now apply this formula to the cost function

$$f(\mathbf{Q}) = \frac{1}{2}\|\mathbf{Q}\mathbf{x} - \mathbf{y}\|_2^2,$$

that is involved in the learning of an orthogonal matrix (Section 2.5.1). The gradient $\mathrm{grad}\bar{f}(\mathbf{Q})$ in the embedding space is obtained from the identity

$$D\bar{f}(\mathbf{Q})[\xi_\mathbf{Q}] = -\mathbf{y}^T\xi_\mathbf{Q}\mathbf{x} - \mathbf{x}^T\xi_\mathbf{Q}^T\mathbf{y} + \mathbf{x}^T\xi_\mathbf{Q}^T\mathbf{Q}\mathbf{x} + \mathbf{x}^T\mathbf{Q}^T\xi_\mathbf{Q}\mathbf{x} = g_\mathbf{Q}((\mathbf{Q}\mathbf{x} - \mathbf{y})\mathbf{x}^T, \xi_\mathbf{Q}),$$

from which we deduce $\mathrm{grad}\bar{f}(\mathbf{Q}) = (\hat{\mathbf{y}} - \mathbf{y})\mathbf{x}^T$. The projection of $\mathrm{grad}\bar{f}(\mathbf{Q})$ onto $T_\mathbf{Q}\mathcal{O}(d)$ yields

$$\mathrm{grad}f(\mathbf{Q}) = P_\mathbf{Q}((\hat{\mathbf{y}} - \mathbf{y})\mathbf{x}^T) = \mathbf{Q}\mathrm{Skew}(\mathbf{Q}^T(\hat{\mathbf{y}} - \mathbf{y})\mathbf{x}^T). \tag{3.13}$$

The retraction mapping

$$R_\mathbf{Q}(\xi_\mathbf{Q}) = R_\mathbf{Q}(\mathbf{Q}\mathbf{\Omega}) = \mathbf{Q}\exp(\mathbf{\Omega}), \tag{3.14}$$

where $\exp(\cdot)$ is the matrix exponential, induces a line-search along the geodesics of $\mathcal{O}(d)$ (see e.g. Absil et al., 2008; Arora, 2009). Combining (3.13) with (3.14) yields

$$\mathbf{Q}_{t+1} = \mathbf{Q}_t \exp(-s_t\,\mathrm{Skew}(\mathbf{Q}_t^T(\hat{\mathbf{y}}_t - \mathbf{y}_t)\mathbf{x}_t^T)), \tag{3.15}$$

which is update (2.20) proposed by Arora (2009).

3.4.2 Line-search algorithms on quotient manifolds

We consider a quotient manifold $\mathcal{W} = \overline{\mathcal{W}}/\sim$ with total space $\overline{\mathcal{W}}$ and equivalence relation $\sim$. The total space $\overline{\mathcal{W}}$ is endowed with a Riemannian metric $\bar{g}_\mathbf{W}$, and the horizontal space $\mathcal{H}_\mathbf{W}\mathcal{W}$ is chosen as the orthogonal complement of the vertical space $\mathcal{V}_\mathbf{W}\mathcal{W}$ in $T_\mathbf{W}\overline{\mathcal{W}}$ according to $\bar{g}_\mathbf{W}$,

$$\mathcal{H}_\mathbf{W}\mathcal{W} = \{\xi_\mathbf{W} \in T_\mathbf{W}\overline{\mathcal{W}} : \bar{g}_\mathbf{W}(\xi_\mathbf{W}, \chi_\mathbf{W}) = 0,\ \forall \chi_\mathbf{W} \in \mathcal{V}_\mathbf{W}\mathcal{W}\}. \tag{3.16}$$

Recall from the developments of Section 3.3 that a tangent vector $\xi_{[\mathbf{W}]} \in T_{[\mathbf{W}]}\mathcal{W}$ to the quotient manifold is represented by its horizontal lift. The horizontal lift of $\xi_{[\mathbf{W}]} \in T_{[\mathbf{W}]}\mathcal{W}$ is the unique tangent vector $\bar{\xi}_\mathbf{W} \in \mathcal{H}_\mathbf{W}\mathcal{W}$ that satisfies $D\pi(\mathbf{W})[\bar{\xi}_\mathbf{W}] = \xi_{[\mathbf{W}]}$.

If, for every $[\mathbf{W}] \in \mathcal{W}$ and every $\xi_{[\mathbf{W}]}, \zeta_{[\mathbf{W}]} \in T_{[\mathbf{W}]}\mathcal{W}$, the expression $\bar{g}_\mathbf{W}(\bar{\xi}_\mathbf{W}, \bar{\zeta}_\mathbf{W})$ does not depend on the choice of the representative $\mathbf{W} \in \overline{\mathcal{W}}$, then the metric in the total space $\bar{g}_\mathbf{W}$ induces a metric $g_{[\mathbf{W}]}$ on the quotient space,

$$g_{[\mathbf{W}]}(\xi_{[\mathbf{W}]}, \zeta_{[\mathbf{W}]}) \triangleq \bar{g}_\mathbf{W}(\bar{\xi}_\mathbf{W}, \bar{\zeta}_\mathbf{W}).$$

Endowed with such a metric, the quotient manifold $\mathcal{W}$ is called a *Riemannian quotient manifold* of $\overline{\mathcal{W}}$, and the quotient map $\pi : \overline{\mathcal{W}} \to \mathcal{W}$ is called a *Riemannian submersion*.

A cost function $\bar{f} : \overline{\mathcal{W}} \to \mathbb{R}$ in the total space is invariant along the fibers if $\bar{f}(\mathbf{W}_1) = \bar{f}(\mathbf{W}_2)$ whenever $\mathbf{W}_1 \sim \mathbf{W}_2$. Such a function induces a function $f : \mathcal{W} \to \mathbb{R}$ on the quotient space and one has the following convenient formula for computing its horizontal gradient

$$\overline{\mathrm{grad}\, f}(\mathbf{W}) = \mathrm{grad}\, \bar{f}(\mathbf{W}). \tag{3.17}$$

Indeed, since $\bar{f}$ is constant on each equivalence class, we have

$$\bar{g}_\mathbf{W}(\mathrm{grad}\bar{f}(\mathbf{W}), \chi_\mathbf{W}) = D\bar{f}(\mathbf{W})[\chi_\mathbf{W}] = 0, \quad \forall \chi_\mathbf{W} \in \mathcal{V}_\mathbf{W}\mathcal{W}.$$

Using the definition (3.16), the previous identity implies that $\mathrm{grad}\bar{f}(\mathbf{W}) \in \mathcal{H}_\mathbf{W}\mathcal{W}$.

Riemannian quotient manifolds are particularly convenient to work with because the computation of several differential geometric objects in the quotient space can be directly performed by means of their analogous in the total space. For instance, for a quotient manifold that is not a Riemannian quotient manifold, formula (3.17) must be modified into

$$\overline{\mathrm{grad}\, f}(\mathbf{W}) = P^\mathcal{H}_\mathbf{W}(\mathrm{grad}\, \bar{f}(\mathbf{W})).$$

Line-search on the Grassmann manifold

In this section, we exploit the material presented by Edelman et al. (1998) to derive a line-search algorithm on the Grassmann Manifold $\mathrm{Gr}(r,d) \simeq \mathrm{St}(r,d)/\mathcal{O}(r)$. The quotient manifold is defined by the set of equivalence classes

$$[\mathbf{U}] = \{\mathbf{UQ} : \mathbf{Q} \in \mathcal{O}(r)\}.$$

The set of vectors that are tangent to this set of equivalence classes is given by

$$\mathcal{V}_{\mathbf{U}}\mathrm{Gr}(r,d) = \{\mathbf{U\Omega} : \mathbf{\Omega}^T = -\mathbf{\Omega} \in \mathbb{R}^{r\times r}\}.$$

Indeed, let $\gamma : \mathbb{R} \to \mathrm{St}(r,d) : t \mapsto \mathbf{UQ}(t)$ be a curve along an equivalence class $[\mathbf{U}]$ and such that $\gamma(0) = \mathbf{U}$ (hence, $\mathbf{Q}(0) = \mathbf{I}$). The derivative of γ at $\mathbf{U}$ is $\dot\gamma(0) = \mathbf{U}\dot{\mathbf{Q}}(0) = \mathbf{UQ}(0)\mathbf{\Omega} = \mathbf{U\Omega}$.

The total space $\mathrm{St}(r,d)$ is endowed with the Euclidean metric

$$\bar g_{\mathbf{U}}(\xi_{\mathbf{U}}, \zeta_{\mathbf{U}}) = \mathrm{Tr}(\xi_{\mathbf{U}}^T \zeta_{\mathbf{U}}). \tag{3.18}$$

Horizontal vectors $\bar\xi_{\mathbf{U}} \in \mathcal{H}_{\mathbf{U}}\mathrm{Gr}(r,d)$ are chosen orthogonal to vertical vectors $\mathbf{U\Omega} \in \mathcal{V}_{\mathbf{U}}\mathrm{Gr}(r,d)$ according to (3.18). Therefore, any horizontal vector $\bar\xi_{\mathbf{U}} \in \mathcal{H}_{\mathbf{U}}\mathrm{Gr}(r,d)$ at $\mathbf{U}$ must satisfy

$$\mathrm{Tr}(\bar\xi_{\mathbf{U}}^T \mathbf{U\Omega}) = 0, \quad \forall \mathbf{\Omega}.$$

The latter condition holds for all skew-symmetric matrices $\mathbf{\Omega}$ if and only if

$$\bar\xi_{\mathbf{U}}^T \mathbf{U} = \mathbf{U}^T \bar\xi_{\mathbf{U}}. \tag{3.19}$$

Moreover, the expression of a generic tangent vector in the total space $\mathrm{St}(r,d)$ is

$$\bar\xi_{\mathbf{U}} = \mathbf{UA} + \mathbf{U}_\perp \mathbf{K} \in T_{\mathbf{U}}\mathrm{St}(r,d), \tag{3.20}$$

where $\mathbf{A}^T = -\mathbf{A} \in \mathbb{R}^{r\times r}$ and $\mathbf{K} \in \mathbb{R}^{(d-r)\times r}$. Substituting (3.20) into (3.19) yields $\mathbf{A}^T = \mathbf{A}$. Therefore, we deduce that $\mathbf{A} = 0$ and $\mathbf{K} \in \mathbb{R}^{(d-r)\times r}$, which gives us the horizontal space

$$\mathcal{H}_{\mathbf{U}}\mathrm{Gr}(r,d) = \{\mathbf{U}_\perp \mathbf{K} : \mathbf{K} \in \mathbb{R}^{(d-r)\times r}\}.$$

Given an element $\xi_{\mathbf{U}} \in T_{\mathbf{U}}\mathrm{St}(r,d)$, it admits the decomposition $\xi_{\mathbf{U}} = P_{\mathbf{U}}^{\mathcal{H}}(\xi_{\mathbf{U}}) + P_{\mathbf{U}}^{\mathcal{V}}(\xi_{\mathbf{U}})$, where

$$P_{\mathbf{U}}^{\mathcal{H}}(\xi_{\mathbf{U}}) = (\mathbf{I} - \mathbf{UU}^T)\xi, \quad P_{\mathbf{U}}^{\mathcal{V}}(\xi_{\mathbf{U}}) = \mathbf{U}\mathrm{Skew}(\mathbf{U}^T\xi_{\mathbf{U}}).$$

We now derive a gradient descent algorithm for learning a subspace (Section 2.5.3). We consider the cost function

$$\bar f(\mathbf{U}) = \frac{1}{2}\|\mathbf{UU}^T\mathbf{x} - \mathbf{x}\|_2^2. \tag{3.21}$$

For all $\mathbf{Q} \in \mathcal{O}(r)$, we have $\bar f(\mathbf{UQ}) = \bar f(\mathbf{U})$ and therefore $\bar f$ induces a function

$$f : \mathrm{Gr}(r,d) \to \mathbb{R} : \mathrm{range}(\mathbf{U}) \mapsto \frac{1}{2}\|\mathbf{UU}^T\mathbf{x} - \mathbf{x}\|_2^2, \tag{3.22}$$

which is defined on the quotient space. The horizontal gradient of f is given by

$$\overline{\mathrm{grad} f}(\mathbf{U}) = \mathrm{grad}\bar f(\mathbf{U}) = -(\mathbf{I} - \mathbf{UU}^T)\mathbf{xx}^T\mathbf{U}. \tag{3.23}$$

A standard retraction in $\mathrm{Gr}(r,d)$ is

$$\mathrm{Exp}_{\mathbf{U}}(\bar\xi_{\mathbf{U}}) = \mathbf{UR}\cos(\mathbf{\Sigma})\mathbf{R}^T + \mathbf{L}\sin(\mathbf{\Sigma})\mathbf{R}^T, \tag{3.24}$$

which is obtained from a singular value decomposition of the horizontal vector $\bar{\xi}_{\mathbf{U}} = \mathbf{L}\mathbf{\Sigma}\mathbf{R}^T$. This retraction induces a line-search along geodesics in $\mathrm{Gr}(r, d)$. Following Absil et al. (2004), an alternative convenient retraction in $\mathrm{Gr}(r, d)$ is given by

$$R_{\mathbf{U}}(s\bar{\xi}_{\mathbf{U}}) = \mathrm{qf}(\mathbf{U} + s\bar{\xi}_{\mathbf{U}}), \tag{3.25}$$

where $\mathrm{qf}(\cdot)$ is a function that extracts the orthogonal factor of the QR-decomposition of its argument. A possible advantage of the retraction (3.25) over the retraction (3.24) is that, in contrast to the SVD computation, the QR decomposition is computed in a fixed number $O(dr^2)$ of arithmetic operations. Combining (3.23) with (3.25) yields the gradient descent update

$$\mathbf{U}_{t+1} = \mathrm{qf}(\mathbf{U}_t + s_t(\mathbf{I} - \mathbf{U}_t\mathbf{U}_t^T)\mathbf{x}_t\mathbf{x}_t^T\mathbf{U}_t),$$

which is Oja's update for subspace tracking (Oja, 1992).

3.5 Trust-region algorithms on matrix manifolds

In contrast with line-search algorithms, trust-region algorithms exploit second-order derivative information on the cost function to identify the search direction. Therefore, they usually converge faster than their line-search counterpart. Indeed, the convergence rate of trust-region algorithms is superlinear, whereas line-search methods converge only linearly (Absil et al., 2007, 2008).

The second-order derivative information is exploited by means of the notion of Riemannian Hessian that generalizes to manifolds the classical notion of Hessian in $\mathbb{R}^d$.

Given a cost function $f : \mathcal{W} \to \mathbb{R}$ on a Riemannian manifold $\mathcal{W}$, the Riemannian Hessian of f at a point $\mathbf{W} \in \mathcal{W}$ is the linear mapping $\mathrm{Hess}\, f(\mathbf{W})[\xi_{\mathbf{W}}] : T_{\mathbf{W}}\mathcal{W} \to T_{\mathbf{W}}\mathcal{W}$ defined by

$$\mathrm{Hess}\, f(\mathbf{W})[\xi_{\mathbf{W}}] \triangleq \nabla_{\xi_{\mathbf{W}}} \mathrm{grad}\, f(\mathbf{W}),$$

for all $\xi_{\mathbf{W}} \in T_{\mathbf{W}}\mathcal{W}$, where $\nabla_{\xi_{\mathbf{W}}}$ is the Riemannian connection on $\mathcal{W}$.

Riemannian connections generalize the notion of directional derivative of a vector field to Riemannian manifolds. A vector field ζ on a manifold $\mathcal{W}$ is a map that assigns to each point $\mathbf{W} \in \mathcal{W}$ a tangent vector $\zeta_{\mathbf{W}} \in T_{\mathbf{W}}\mathcal{W}$. The Riemannian gradient of a function is a typical example of vector field. The general definition of the Riemannian connection follows from the fundamental theorem of differential geometry which states that, given a Riemannian metric $g_{\mathbf{W}}$, there exists a unique "torsion-free" connection associated with this metric.

Given two vector fields ξ and ζ on $\mathcal{W}$, the Riemannian connection associated to $g_{\mathbf{W}}$ is characterized by Koszul formula (Absil et al., 2008, Theorem 5.3.1),

$$\begin{aligned} 2g_{\mathbf{W}}(\nabla_\xi\zeta, \nu) = Dg_{\mathbf{W}}(\zeta, \nu)[\xi] + Dg_{\mathbf{W}}(\xi, \nu)[\zeta] - Dg_{\mathbf{W}}(\zeta, \xi)[\nu] \\ + g_{\mathbf{W}}(\nu, [\xi, \zeta]) + \bar{g}_{\mathbf{W}}(\zeta, [\nu, \xi]) - g_{\mathbf{W}}(\xi, [\zeta, \nu]). \end{aligned} \tag{3.26}$$

The vector field $[\xi, \zeta]$ is the *Lie bracket* between vector fields ξ and ζ. For a manifold $\mathcal{W}$ that is an open subset of a vector space, the Lie bracket vector field is given by

$$[\xi, \zeta] = D\zeta[\xi] - D\xi[\zeta], \tag{3.27}$$

where $D\zeta[\xi]$ is the classical directional derivative of a vector field in vector spaces,

$$D\zeta[\xi] = \lim_{t \to 0} \frac{\zeta_{\mathbf{W}+t\xi_{\mathbf{W}}} - \zeta_{\mathbf{W}}}{t}. \tag{3.28}$$

The Riemannian Hessian is exploited by trust-region algorithms on Riemannian manifolds that sequentially solve the problem

$$\min_{\xi_{\mathbf{W}} \in T_{\mathbf{W}}\mathcal{W}} m_{\mathbf{W}}(\xi_{\mathbf{W}}) \quad \text{subject to} \quad g_{\mathbf{W}}(\xi_{\mathbf{W}}, \xi_{\mathbf{W}}) \leq \delta^2, \tag{3.29}$$

which amounts to finding a search direction in the tangent space that minimizes the quadratic model of the cost function

$$m_{\mathbf{W}}(\xi_{\mathbf{W}}) = f(\mathbf{W}) + g_{\mathbf{W}}(\xi_{\mathbf{W}}, \operatorname{grad} f(\mathbf{W})) + \frac{1}{2} g_{\mathbf{W}}(\xi_{\mathbf{W}}, \operatorname{Hess} f(\mathbf{W})[\xi_{\mathbf{W}}]),$$

on a trust-region of radius δ around the current iterate $\mathbf{W}$. Once a search direction $\xi_{\mathbf{W}}$ is computed, the search variable is updated using the retraction. At time t, this iteration is

$$\mathbf{W}_{t+1} = R_{\mathbf{W}_t}(\xi_{\mathbf{W}_t}). \tag{3.30}$$

The trust-region radius δ varies according to the quality of the new iterate. When a good solution is found within the trust-region, then the trust-region is expanded. Conversely, if the iterate is of poor quality, then the region is contracted. The quality of an iterate is usually evaluated by computing the ratio between the decrease in the actual cost function and the decrease in the quadratic model of the cost function,

$$\rho_t = \frac{f(\mathbf{W}_t) - f(R_{\mathbf{W}_t}(\xi_{\mathbf{W}_t}))}{m_{\mathbf{W}_t}(0) - m_{\mathbf{W}_t}(\xi_{\mathbf{W}_t})}.$$

The larger ρ_t, the better the quality of the current iterate.

From the previous definitions, we see that a requirement for the derivation of a Riemannian trust-region algorithm is a convenient formula for computing the Riemannian connection.

In general, computing the Riemannian connection for a given manifold is not straightforward. However, convenient formulas are available for the case of embedded submanifolds (Section 3.5.1) and the case of quotient manifolds (Section 3.5.2). These formula are exploited in Chapter 6 to compute the Riemannian Hessian of the quadratic cost function for linear regression.

3.5.1 Riemannian connection on embedded submanifolds

Let $\mathcal{W}$ be a Riemannian submanifold embedded in the Euclidean space $\mathbb{R}^{d\times r}$. Let ζ be a vector field on $\mathcal{W}$ that assigns to each point $\mathbf{W} \in \mathcal{W}$ a tangent vector $\zeta_{\mathbf{W}} \in T_{\mathbf{W}}\mathcal{W}$. The Riemannian connection of the vector field ζ at $\mathbf{W} \in \mathcal{W}$ in a direction $\xi_{\mathbf{W}} \in T_{\mathbf{W}}\mathcal{W}$ is obtained as

$$\nabla_{\xi_{\mathbf{W}}} \zeta_{\mathbf{W}} = P_{\mathbf{W}}(D\zeta_{\mathbf{W}}[\xi_{\mathbf{W}}]), \tag{3.31}$$

where $P_{\mathbf{W}}(\cdot)$ is the orthogonal projection onto $T_{\mathbf{W}}\mathcal{W}$ and $D\zeta_{\mathbf{W}}[\xi_{\mathbf{W}}]$ is the directional derivative

$$D\zeta_{\mathbf{W}}[\xi_{\mathbf{W}}] = \lim_{t\to 0} \frac{\zeta_{\mathbf{W}+t\xi_{\mathbf{W}}} - \zeta_{\mathbf{W}}}{t}. \tag{3.32}$$

Riemannian connection on the Stiefel manifold

Consider $\tilde{\zeta}$ a vector field on $\mathbb{R}^{d\times r}$ and ζ the associated vector field on the Stiefel manifold $\mathrm{St}(r,d)$. The vector field ζ assigns to each point $\mathbf{U} \in \mathrm{St}(r,d)$ a tangent vector

$$\zeta_{\mathbf{U}} = P_{\mathbf{U}}(\tilde{\zeta}_{\mathbf{U}}) = (\mathbf{I} - \mathbf{U}\mathbf{U}^T)\tilde{\zeta}_{\mathbf{U}} + \mathbf{U}\mathrm{Skew}(\mathbf{U}^T\tilde{\zeta}_{\mathbf{U}}). \tag{3.33}$$

The Euclidean directional derivative of $\zeta_{\mathbf{U}}$ at $\mathbf{U}$ in a direction $\xi_{\mathbf{U}} \in T_{\mathbf{U}}\mathrm{St}(r,d)$ is

$$D\zeta_{\mathbf{U}}[\xi_{\mathbf{U}}] = \lim_{t\to 0} \frac{\zeta_{\mathbf{U}+t\xi_{\mathbf{U}}} - \zeta_{\mathbf{U}}}{t} = P_{\mathbf{U}}(D\tilde{\zeta}_{\mathbf{U}}[\xi_{\mathbf{U}}]) - \xi_{\mathbf{U}}\mathrm{Sym}(\mathbf{U}^T\tilde{\zeta}_{\mathbf{U}}) - \mathbf{U}\mathrm{Sym}(\xi_{\mathbf{U}}^T\tilde{\zeta}_{\mathbf{U}}),$$

which generally not belongs to $T_{\mathbf{U}}\mathrm{St}(r,d)$. Applying formula (3.31) yields

$$\nabla_{\xi_{\mathbf{U}}}\zeta_{\mathbf{U}} = P_{\mathbf{U}}(D\tilde{\zeta}_{\mathbf{U}}[\xi_{\mathbf{U}}] - \xi_{\mathbf{U}}\mathrm{Sym}(\mathbf{U}^T\tilde{\zeta}_{\mathbf{U}})).$$

3.5.2 Riemannian connection on quotient manifolds

For a Riemannian quotient manifold $\mathcal{W} = \overline{\mathcal{W}}/\sim$, the Riemannian connection in the quotient space $\mathcal{W}$ is computed from the Riemannian connection in the total space $\overline{\mathcal{W}}$. This is achieved by projecting the Riemannian connection in the total space $\overline{\nabla}$ onto the horizontal space,

$$\overline{\nabla_{\xi_{[\mathbf{W}]}}\, \zeta_{[\mathbf{W}]}} = P^{\mathcal{H}}_{\mathbf{W}}(\overline{\nabla}_{\bar{\xi}_{\mathbf{W}}} \bar{\zeta}_{\mathbf{W}}), \tag{3.34}$$

where $\bar{\xi}_{\mathbf{W}}$ is the horizontal lift of $\xi_{[\mathbf{W}]}$ and $\bar{\zeta}$ is the horizontal vector field associated with ζ. Given a vector field $\zeta_{[\mathbf{W}]}$ on the quotient space, its horizontal vector field $\bar{\zeta}$ assigns to each point of the total space $\mathbf{W} \in \overline{\mathcal{W}}$ a horizontal lift $\bar{\zeta}_{\mathbf{W}} \in \mathcal{H}_{\mathbf{W}}\mathcal{W}$. Formula (3.34) further simplifies if the total space $\overline{\mathcal{W}}$ is an open subset of a Euclidean space,

$$\overline{\nabla_{\xi_{[\mathbf{W}]}}\, \zeta_{[\mathbf{W}]}} = P^{\mathcal{H}}_{\mathbf{W}}(D\bar{\zeta}_{\mathbf{W}}[\bar{\xi}_{\mathbf{W}}]).$$

Riemannian connection on the Grassmann manifold

Consider $\tilde{\zeta}$ a vector field on $\mathbb{R}^{d\times r}$ and $\bar{\zeta}$ the associated horizontal vector field on the Grassmann manifold $\mathrm{Gr}(r,d) \simeq \mathrm{St}(r,d)/\mathcal{O}(r)$. The horizontal vector field $\bar{\zeta}$ assigns to each point in the total space $\mathbf{U} \in \mathrm{St}(r,d)$ a horizontal vector

$$\bar{\zeta}_{\mathbf{U}} = P^{\mathcal{H}}_{\mathbf{U}}(\tilde{\zeta}_{\mathbf{U}}) = (\mathbf{I} - \mathbf{U}\mathbf{U}^T)\tilde{\zeta}_{\mathbf{U}}.$$

By virtue of (3.31), the Riemannian connection of this horizontal vector field in $\mathrm{St}(r,d)$ with respect to the horizontal vector $\bar{\xi}_{\mathbf{U}} \in \mathcal{H}_{\mathbf{U}}\mathrm{Gr}(r,d)$ is

$$\overline{\nabla}_{\bar{\xi}_{\mathbf{U}}} \bar{\zeta}_{\mathbf{U}} = P_{\mathbf{U}}(P^{\mathcal{H}}_{\mathbf{U}}(D\tilde{\zeta}_{\mathbf{U}}[\bar{\xi}_{\mathbf{U}}]) - \bar{\xi}_{\mathbf{U}}\mathbf{U}^T\tilde{\zeta}_{\mathbf{U}} - \mathbf{U}\bar{\xi}^T_{\mathbf{U}}\tilde{\zeta}_{\mathbf{U}}).$$

Applying formula (3.34) leads to the desired Riemannian connection on the quotient space

$$\overline{\nabla_{\xi_{[\mathbf{U}]}}\, \zeta_{[\mathbf{U}]}} = P^{\mathcal{H}}_{\mathbf{U}}(\overline{\nabla}_{\bar{\xi}_{\mathbf{U}}} \bar{\zeta}_{\mathbf{U}}) = P^{\mathcal{H}}_{\mathbf{U}}(D\tilde{\zeta}_{\mathbf{U}}[\bar{\xi}_{\mathbf{U}}]) - \bar{\xi}_{\mathbf{U}}\mathbf{U}^T\tilde{\zeta}_{\mathbf{U}}.$$

Chapter 4

Regression on fixed-rank symmetric positive semidefinite matrices

Chapter abstract: In this chapter, we address the problem of learning a linear regression model parameterized by a fixed-rank symmetric positive semidefinite matrix. The focus is on the nonlinear nature of the search space and on scalability to high-dimensional problems. The mathematical developments rely on the theory of gradient descent algorithms adapted to the Riemannian geometry that underlies the set of fixed-rank positive semidefinite matrices. In contrast with previous contributions in the literature, no restrictions are imposed on the range space of the learned matrix. The resulting algorithms maintain a linear complexity in the problem size and enjoy important invariance properties. We apply the proposed algorithms to the problem of learning a distance function parameterized by a positive semidefinite matrix. Good performance is observed on classical distance and kernel learning benchmarks.

The material of this chapter is based on the following publications:

> G. Meyer, S. Bonnabel, R. Sepulchre
> Regression on fixed-rank positive semidefinite matrices: a Riemannian approach
> *Journal of Machine Learning Research.* 12(Feb):593-625, 2011.
>
> G. Meyer, M. Journée, S. Bonnabel and R. Sepulchre
> From subspace learning to distance learning: a geometrical optimization approach
> In *Proc. of the 15th Workshop on Statistical Signal Processing*, Cardiff (Wales), 2009.

Continuous-time gradient flow versions of the proposed algorithms and a connection with symmetry-preserving observers are presented in the following publication:

> S. Bonnabel, G. Meyer and R. Sepulchre
> Adaptive filtering for estimation of a low-rank positive semidefinite matrix
> In *Proc. of the 19th International Symposium on Mathematical Theory of Networks and Systems*, Budapest (Hungary), 2010.

4.1 Introduction

A fundamental problem of machine learning is the learning of a distance between data samples. When the distance can be written as a quadratic form (either in the data space (Mahalanobis distance) or in a kernel feature space (kernel distance)), the learning problem is a regression problem on the set of positive definite matrices.

Following the developments presented in Chapter 2, this regression problem is classically turned into the minimization of the prediction error, leading to an optimization framework and gradient-based learning algorithms.

In the present chapter, we exploit the rich geometry of the set of fixed-rank symmetric

positive semidefinite matrices to design efficient linear regression algorithms.

The classical framework of gradient-based learning can be generalized provided that the nonlinear search space of interest is equipped with a proper Riemannian geometry. Adopting this general framework, we design novel learning algorithms on the space of fixed-rank positive semidefinite matrices, denoted by $S_+(r,d)$, where d is the dimension of the matrix, and r is its rank. Learning a parametric model in $S_+(r,d)$ amounts to jointly learning a r-dimensional subspace and a quadratic distance in this subspace.

The framework is motivated by *low-rank learning* in large-scale applications. If the data space is of dimension d, the goal is to maintain a linear computational complexity $O(d)$. In contrast to the classical approach of first reducing the dimension of the data and then learning a distance in the reduced space, there is an obvious conceptual advantage in performing the two tasks simultaneously. If this objective can be achieved without increasing the numerical cost of the algorithm, the advantage becomes also practical.

Our approach makes use of two quotient geometries of the set $S_+(r,d)$ that have been recently studied by Journée et al. (2010) and Bonnabel and Sepulchre (2009). Making use of a general theory of line-search algorithms in quotient matrix spaces (Absil et al., 2008), we obtain concrete gradient updates that maintain the rank and the positivity of the learned model at each iteration. This is because the update is intrinsically constrained to belong to the nonlinear search space, in contrast to early learning algorithms that neglect the nonlinear nature of the search space in the update and impose the constraints a posteriori (Xing et al., 2002; Globerson and Roweis, 2005).

Not surprisingly, our approach has close connections with a number of recent contributions on learning algorithms. Learning problems over nonlinear matrix spaces include the learning of subspaces (Crammer, 2006; Warmuth, 2007), rotation matrices (Arora, 2009), and positive definite matrices (Tsuda et al., 2005). The space of (full-rank) positive definite matrices $S_{++}(d)$ is of particular interest since it coincides with our set of interest in the particular case $r = d$.

The use of Bregman divergences and alternating projection has been recently investigated for learning in $S_{++}(d)$. Tsuda et al. (2005) propose to use the *von Neumann* divergence, resulting in a generalization of the well-known AdaBoost algorithm (Schapire and Singer, 1999) to positive definite matrices. The use of the so-called *LogDet* divergence has also been investigated by Davis et al. (2007) in the context of Mahalanobis distance learning.

More recently, algorithmic work has focused on scalability in terms of dimensionality and data set size. A natural extension of the previous work on positive definite matrices is thus to consider low-rank positive semidefinite matrices. Indeed, whereas algorithms based on full-rank matrices scale as $O(d^3)$ and require $O(d^2)$ storage units, algorithms based on low-rank matrices scale as $O(dr^2)$ and require $O(dr)$ storage units (Fine et al., 2001; Bach and Jordan, 2005). This is a significant complexity reduction as the approximation rank r is typically very small compared to the dimension of the problem d.

Extending the work of Tsuda et al. (2005), Kulis et al. (2009) recently considered the learning of positive semidefinite matrices. The authors consider Bregman divergence measures that enjoy convexity properties and lead to updates that preserve the rank as well as the positive semidefinite property. However, these divergence-based algorithms intrinsically constrain the learning algorithm to a fixed range space. A practical limitation of this approach is that the subspace of the learned matrix is fixed beforehand by the initial condition of the algorithm.

The proposed approach is in a sense more classical (we just perform a line-search in a Riemannian manifold) but we show how to interpret Bregman divergence based algorithms in our framework. This is potentially a contribution of independent interest since a general convergence theory exists for line-search algorithms on Riemannian manifolds. The generality of the proposed framework is of course motivated by the non-convex nature of the rank constraint.

The current chapter is organized as follows. Section 4.2 reviews regression on positive definite matrices in the considered geometric framework. Section 4.3 presents the proposed generalization

to fixed-rank positive semidefinite matrices. Section 4.4 focuses on the development of novel algorithms for learning fixed-rank PSD matrices, discusses the relationship to existing work and presents potential extensions. Section 4.5 discusses the relationship to existing work as well as extensions of the proposed approach. Section 4.6 shows experimental results.

4.2 Regression on the cone of positive definite matrices

We first consider the learning of a full-rank positive definite matrix, which is recast as follows. Let $\mathcal{X} = \mathbb{R}^{d\times d}$ and $\mathcal{Y} = \mathbb{R}$, and consider the model

$$\hat{y} = \mathrm{Tr}(\mathbf{W}\mathbf{X}),$$

with $\mathbf{W} \in S_{++}(d) = \{\mathbf{W} \in \mathbb{R}^{d\times d} : \ \mathbf{W} = \mathbf{W}^T \succ 0\}$. Since $\mathbf{W}$ is symmetric, only the symmetric part of $\mathbf{X}$ will contribute to the trace. The previous model is thus equivalent to

$$\hat{y} = \mathrm{Tr}(\mathbf{W}\mathrm{Sym}(\mathbf{X})),$$

The quadratic loss is

$$f(\mathbf{W}) = \ell(\hat{y}, y) = \frac{1}{2}(\mathrm{Tr}(\mathbf{W}\mathrm{Sym}(\mathbf{X})) - y)^2. \tag{4.1}$$

The quotient geometries of $S_{++}(d)$ are rooted in the matrix factorization

$$\mathbf{W} = \mathbf{G}\mathbf{G}^T, \quad \mathbf{G} \in \mathrm{GL}(d),$$

where $\mathrm{GL}(d)$ is the set of all invertible $d \times d$ matrices,

$$\mathrm{GL}(d) = \{\mathbf{M} \in \mathbb{R}^{d\times d} : \det(\mathbf{M}) \neq 0\}.$$

Because the factorization is invariant by rotation $\mathbf{G} \mapsto \mathbf{G}\mathbf{O}$ whenever $\mathbf{O} \in \mathcal{O}(d)$, the search space can be identified to a quotient manifold

$$S_{++}(d) \simeq \mathrm{GL}(d)/\mathcal{O}(d),$$

which represents the set of equivalence classes

$$[\mathbf{G}] = \{\mathbf{G}\mathbf{O} : \ \mathbf{O} \in \mathcal{O}(d)\}.$$

We will equip this quotient with two meaningful Riemannian metrics.

4.2.1 A flat metric on $S_{++}(d)$

The first metric that we propose is induced by the standard metric in $\mathbb{R}^{d\times d}$,

$$\bar{g}_{\mathbf{G}}(\bar{\xi}_{\mathbf{G}}, \bar{\zeta}_{\mathbf{G}}) = \mathrm{Tr}(\bar{\xi}_{\mathbf{G}}^T \bar{\zeta}_{\mathbf{G}}).$$

As the chosen metric is invariant by rotation along the set of equivalence classes, it induces a metric on the quotient manifold $S_{++}(d) \simeq \mathrm{GL}(d)/\mathcal{O}(d)$,

$$g_{[\mathbf{G}]}(\xi_{[\mathbf{G}]}, \zeta_{[\mathbf{G}]}) \triangleq \bar{g}_{\mathbf{G}}(\bar{\xi}_{\mathbf{G}}, \bar{\zeta}_{\mathbf{G}}).$$

With this geometry, a tangent vector $\xi_{[\mathbf{G}]}$ at $[\mathbf{G}]$ is represented by a horizontal tangent vector $\bar{\xi}_{\mathbf{G}}$ at $\mathbf{G}$ by

$$\bar{\xi}_{\mathbf{G}} = \mathrm{Sym}(\mathbf{\Delta})\mathbf{G}, \quad \mathbf{\Delta} \in \mathbb{R}^{d\times d}.$$

The horizontal gradient of

$$f(\mathbf{G}) = \ell(\hat{y}, y) = \frac{1}{2}(\mathrm{Tr}(\mathbf{G}\mathbf{G}^T\mathrm{Sym}(\mathbf{X})) - y)^2 \tag{4.2}$$

is the unique horizontal vector $\overline{\text{grad}f}(\mathbf{G})$ that satisfies

$$Df(\mathbf{G})[\bar{\xi}_{\mathbf{G}}] = \bar{g}_{\mathbf{G}}(\bar{\xi}_{\mathbf{G}}, \overline{\text{grad}f}(\mathbf{G})).$$

Elementary computations yield

$$\overline{\text{grad}f}(\mathbf{G}) = 2(\hat{y} - y)\text{Sym}(\mathbf{X})\mathbf{G}.$$

Since the metric is flat, geodesics are straight lines and the exponential mapping is

$$\text{Exp}_{\mathbf{G}}(\bar{\xi}_{\mathbf{G}}) = [\mathbf{G} + \bar{\xi}_{\mathbf{G}}] = \mathbf{G} + \bar{\xi}_{\mathbf{G}}.$$

Combining the retraction with the horizontal gradient of the cost (4.2) yields the simple formula

$$\mathbf{G}_{t+1} = \mathbf{G}_t - 2s_t(\hat{y}_t - y_t)\text{Sym}(\mathbf{X}_t)\mathbf{G}_t, \tag{4.3}$$

for an online gradient algorithm and

$$\mathbf{G}_{t+1} = \mathbf{G}_t - 2s_t\frac{1}{n}\sum_{i=1}^{n}(\hat{y}_i - y_i)\text{Sym}(\mathbf{X}_i)\mathbf{G}_t, \tag{4.4}$$

for a batch gradient algorithm.

4.2.2 The affine-invariant metric on $S_{++}(d)$

Because $S_{++}(d) \simeq \text{GL}(d)/\mathcal{O}(d)$ is the quotient of two Lie groups, its (reductive) geometric structure can be further exploited (Faraut and Koranyi, 1994). Indeed the group $\text{GL}(d)$ has a natural action on $S_{++}(d)$ via the transformation $\mathbf{W} \mapsto \mathbf{AWA}^T$ for any $\mathbf{A} \in \text{GL}(d)$. The affine-invariant metric admits interesting invariance properties to these transformations. To build such an affine-invariant metric, the metric at identity

$$g_{\mathbf{I}}(\xi_{\mathbf{I}}, \zeta_{\mathbf{I}}) = \text{Tr}(\xi_{\mathbf{I}}\zeta_{\mathbf{I}}),$$

is extended to the entire space to satisfy the invariance property

$$g_{\mathbf{I}}(\xi_{\mathbf{I}}, \zeta_{\mathbf{I}}) = g_{\mathbf{W}}(\mathbf{W}^{\frac{1}{2}}\xi_{\mathbf{I}}\mathbf{W}^{\frac{1}{2}}, \mathbf{W}^{\frac{1}{2}}\zeta_{\mathbf{I}}\mathbf{W}^{\frac{1}{2}}) = g_{\mathbf{W}}(\xi_{\mathbf{W}}, \zeta_{\mathbf{W}}).$$

The resulting metric on $S_{++}(d)$ is defined by

$$g_{\mathbf{W}}(\xi_{\mathbf{W}}, \zeta_{\mathbf{W}}) = \text{Tr}(\xi_{\mathbf{W}}\mathbf{W}^{-1}\zeta_{\mathbf{W}}\mathbf{W}^{-1}). \tag{4.5}$$

The affine-invariant geometry of $S_{++}(d)$ has been well studied, in particular in the context of information geometry (Smith, 2005). Indeed, any positive definite matrix $\mathbf{W} \in S_{++}(d)$ can be identified to the multivariate normal distribution of zero mean $\mathcal{N}(0, \mathbf{W})$, whose probability density is $p(\mathbf{z}; \mathbf{W}) = \frac{1}{Z}\exp(-\frac{1}{2}\mathbf{z}^T\mathbf{W}^{-1}\mathbf{z})$, where Z is a normalizing constant. Using such a metric allows one to endow the space of parameters $S_{++}(d)$ with a distance that reflects the proximity of the probability distributions. The Riemannian metric thus distorts the Euclidean distances between positive definite matrices in order to reflect the amount of information between the two associated probability distributions. If $\xi_{\mathbf{W}}$ is a tangent vector to $\mathbf{W} \in S_{++}(d)$, we have the following approximation for the Kullback-Leibler divergence (up to third order terms)

$$D_{KL}(p(\mathbf{z}; \mathbf{W})||p(\mathbf{z}; \mathbf{W} + \xi_{\mathbf{W}})) \approx \frac{1}{2}\, g^{FIM}_{\mathbf{W}}(\xi_{\mathbf{W}}, \xi_{\mathbf{W}}) = \frac{1}{2}\, g_{\mathbf{W}}(\xi_{\mathbf{W}}, \xi_{\mathbf{W}}),$$

where $g^{FIM}_{\mathbf{W}}$ is the well-known Fisher information metric at $\mathbf{W}$, which coincides with the affine-invariant metric (4.5) (Smith, 2005). With this geometry, tangent vectors $\xi_{\mathbf{W}}$ are given by

$$\xi_{\mathbf{W}} = \text{Sym}(\mathbf{\Delta}), \quad \mathbf{\Delta} \in \mathbb{R}^{d\times d}.$$

The gradient $\mathrm{grad} f(\mathbf{W})$ is given by

$$Df(\mathbf{W})[\xi_{\mathbf{W}}] = g_{\mathbf{W}}(\xi_{\mathbf{W}}, \mathrm{grad} f(\mathbf{W})).$$

Applying this formula to (4.1) yields

$$\mathrm{grad} f(\mathbf{W}) = (\hat{y} - y)\mathbf{W}\mathrm{Sym}(\mathbf{X})\mathbf{W}. \tag{4.6}$$

The exponential mapping has the closed-form expression

$$\mathrm{Exp}_{\mathbf{W}}(\xi_{\mathbf{W}}) = \mathbf{W}^{\frac{1}{2}} \exp(\mathbf{W}^{-\frac{1}{2}} \xi_{\mathbf{W}} \mathbf{W}^{-\frac{1}{2}}) \mathbf{W}^{\frac{1}{2}}. \tag{4.7}$$

Its first-order approximation provides the convenient retraction

$$R_{\mathbf{W}}(s\xi_{\mathbf{W}}) = \mathbf{W} + s\xi_{\mathbf{W}}. \tag{4.8}$$

The formulas (4.6) and (4.7) applied to the cost (4.1) gives the following update

$$\mathbf{W}_{t+1} = \mathbf{W}_t^{\frac{1}{2}} \exp(-s_t(\hat{y}_t - y_t)\mathbf{W}_t^{\frac{1}{2}} \mathrm{Sym}(\mathbf{X}_t) \mathbf{W}_t^{\frac{1}{2}}) \mathbf{W}_t^{\frac{1}{2}}.$$

With the alternative retraction (4.8), the update becomes

$$\mathbf{W}_{t+1} = \mathbf{W}_t - s_t(\hat{y}_t - y_t)\mathbf{W}_t \mathrm{Sym}(\mathbf{X}_t) \mathbf{W}_t,$$

which is the update of Davis et al. (2007) based on the LogDet divergence (Section 4.5.1 of the present document).

4.2.3 The log-euclidean metric on $S_{++}(d)$

For the sake of completeness, we briefly review a third Riemannian geometry of $S_{++}(d)$, that exploits the property

$$\mathbf{W} = \exp(\mathbf{S}), \quad \mathbf{S} = \mathbf{S}^T \in \mathbb{R}^{d \times d}.$$

The matrix exponential thus provides a global diffeomorphism between $S_{++}(d)$ and the linear space of $d \times d$ symmetric matrices. This geometry is studied in detail in the paper of Arsigny et al. (2007). The cost function

$$f(\mathbf{S}) = \ell(\hat{y}, y) = \frac{1}{2}(\mathrm{Tr}(\exp(\mathbf{S})\mathrm{Sym}(\mathbf{X})) - y)^2$$

thus defines a cost function in the linear space of symmetric matrices. The gradient of this cost function is given by

$$\mathrm{grad} f(\mathbf{S}) = (\hat{y}_t - y_t)\mathrm{Sym}(\mathbf{X}_t),$$

and the retraction that is proposed by Arsigny et al. (2007) is

$$R_{\mathbf{S}}(s\xi_{\mathbf{S}}) = \exp(\mathbf{S} + s\xi_{\mathbf{S}}) = \exp(\log \mathbf{W} + s\xi_{\mathbf{S}}).$$

The corresponding gradient descent update is

$$\mathbf{W}_{t+1} = \exp(\log \mathbf{W}_t - s_t(\hat{y}_t - y_t)\mathrm{Sym}(\mathbf{X}_t)),$$

which is the update of Tsuda et al. (2005) based on the von Neumann divergence (see Section 4.5.1 of the present document).

4.3 From positive definite to fixed-rank positive semidefinite matrices

We now present the proposed generalizations to fixed-rank positive semidefinite matrices.

4.3.1 Linear regression with a flat geometry

The generalization of the results of Section 4.2.1 to the set $S_+(r,d)$ is a straightforward consequence of the factorization

$$\mathbf{W} = \mathbf{G}\mathbf{G}^T, \quad \mathbf{G} \in \mathbb{R}_*^{d\times r},$$

where

$$\mathbb{R}_*^{d\times r} = \{\mathbf{G} \in \mathbb{R}^{d\times r} : \det(\mathbf{G}^T\mathbf{G}) \neq 0\}.$$

Indeed, the flat quotient geometry of the manifold $S_{++}(d) \simeq \mathrm{GL}(d)/\mathcal{O}(d)$ is generalized to the quotient geometry of $S_+(r,d) \simeq \mathbb{R}_*^{d\times r}/\mathcal{O}(r)$ by a mere adaptation of matrix dimension, leading to the updates (4.3) and (4.4) for matrices $\mathbf{G}_t \in \mathbb{R}_*^{d\times r}$.

The mathematical derivation of these updates is a straight application of the material presented in the paper of Journée et al. (2010), where the quotient geometry of $S_+(r,d) \simeq \mathbb{R}_*^{d\times r}/\mathcal{O}(r)$ is studied in details. In the next section, we propose an alternative geometry that jointly learns a r-dimensional subspace and a full-rank quadratic model in this subspace.

4.3.2 Linear regression with a polar geometry

In contrast to the flat geometry, the affine-invariant geometry of $S_{++}(d) \simeq \mathrm{GL}(d)/\mathcal{O}(d)$ does not generalize directly to $S_+(r,d) \simeq \mathbb{R}_*^{d\times r}/\mathcal{O}(r)$ because $\mathbb{R}_*^{d\times r}$ is not a group. However, a generalization is possible by considering the polar matrix factorization

$$\mathbf{G} = \mathbf{U}\mathbf{R}, \quad \mathbf{U} \in \mathrm{St}(r,d),\ \mathbf{R} \in S_{++}(r).$$

The polar factorization is obtained from the SVD of $\mathbf{G} = \mathbf{Z}\mathbf{\Sigma}\mathbf{V}^T$ as $\mathbf{U} = \mathbf{Z}\mathbf{V}^T$ and $\mathbf{R} = \mathbf{V}\mathbf{\Sigma}\mathbf{V}^T$ (Golub and Van Loan, 1996). This gives a polar parameterization of $S_+(r,d)$

$$\mathbf{W} = \mathbf{U}\mathbf{R}^2\mathbf{U}^T.$$

This development leads to the quotient representation

$$S_+(r,d) \simeq (\mathrm{St}(r,d) \times S_{++}(r))/\mathcal{O}(r), \tag{4.9}$$

based on the invariance of $\mathbf{W}$ to the transformation $(\mathbf{U},\mathbf{R}^2) \mapsto (\mathbf{U}\mathbf{O}, \mathbf{O}^T\mathbf{R}^2\mathbf{O})$, $\mathbf{O} \in \mathcal{O}(r)$. It thus describes the set of equivalence classes

$$[(\mathbf{U},\mathbf{R}^2)] = \{(\mathbf{U}\mathbf{O}, \mathbf{O}^T\mathbf{R}^2\mathbf{O}) :\ \mathbf{O} \in \mathcal{O}(r)\}.$$

The cost function is now given by

$$f(\mathbf{U},\mathbf{R}^2) = \ell(\hat{y}, y) = \frac{1}{2}(\mathrm{Tr}(\mathbf{U}\mathbf{R}^2\mathbf{U}^T\mathrm{Sym}(\mathbf{X})) - y)^2. \tag{4.10}$$

The Riemannian geometry of (4.9) has been recently studied by Bonnabel and Sepulchre (2009). The metric

$$\begin{aligned} g_{[\mathbf{W}]}(\xi_{[\mathbf{W}]}, \zeta_{[\mathbf{W}]}) &\triangleq \bar{g}_{\mathbf{W}}(\bar{\xi}_{\mathbf{W}}, \bar{\zeta}_{\mathbf{W}}) \\ &= \frac{1}{\lambda}\,\bar{g}_{\mathbf{U}}(\bar{\xi}_{\mathbf{U}}, \bar{\zeta}_{\mathbf{U}}) + \frac{1}{1-\lambda}\,\bar{g}_{\mathbf{R}^2}(\bar{\xi}_{\mathbf{R}^2}, \bar{\zeta}_{\mathbf{R}^2}), \end{aligned} \tag{4.11}$$

where $\lambda \in (0,1)$, is induced by the metric of $\mathrm{St}(r,d)$ and the affine-invariant metric of $S_{++}(r)$,

$$\bar{g}_{\mathbf{U}}(\bar{\xi}_{\mathbf{U}}, \bar{\zeta}_{\mathbf{U}}) = \mathrm{Tr}(\bar{\xi}_{\mathbf{U}}^T\bar{\zeta}_{\mathbf{U}}), \qquad \bar{g}_{\mathbf{R}^2}(\bar{\xi}_{\mathbf{R}^2}, \bar{\zeta}_{\mathbf{R}^2}) = \mathrm{Tr}(\bar{\xi}_{\mathbf{R}^2}\mathbf{R}^{-2}\bar{\zeta}_{\mathbf{R}^2}\mathbf{R}^{-2}).$$

The proposed metric is invariant along the set of equivalence classes and thus induces a quotient structure on $S_+(r,d)$. Alternative metrics on $S_{++}(r)$ can be considered as long as the metric

remains invariant along the set of equivalence classes. For instance, the log-Euclidean metric discussed in Section 4.2.3 would qualify as a valid alternative.

With this geometry, a tangent vector $\xi_{[\mathbf{W}]} = (\xi_{\mathbf{U}}, \xi_{\mathbf{R}^2})_{[(\mathbf{U},\mathbf{R}^2)]}$ at $[(\mathbf{U}, \mathbf{R}^2)]$ is described by a horizontal tangent vector $\bar{\xi}_{\mathbf{W}} = (\bar{\xi}_{\mathbf{U}}, \bar{\xi}_{\mathbf{R}^2})_{(\mathbf{U},\mathbf{R}^2)}$ at $(\mathbf{U}, \mathbf{R}^2)$ by

$$\bar{\xi}_{\mathbf{U}} = (\mathbf{I} - \mathbf{U}\mathbf{U}^T)\mathbf{\Delta},\ \mathbf{\Delta} \in \mathbb{R}^{d\times r}, \qquad \bar{\xi}_{\mathbf{R}^2} = \mathbf{R}\mathrm{Sym}(\mathbf{\Psi})\mathbf{R},\ \mathbf{\Psi} \in \mathbb{R}^{r\times r}.$$

A retraction is provided by distinct retractions on $\mathbf{U}$ and $\mathbf{R}^2$,

$$R_{\mathbf{U}}(s\bar{\xi}_{\mathbf{U}}) = \mathrm{qf}(\mathbf{U} + s\bar{\xi}_{\mathbf{U}}) \tag{4.12}$$
$$R_{\mathbf{R}^2}(s\bar{\xi}_{\mathbf{R}^2}) = \mathbf{R}\exp(s\mathbf{R}^{-1}\bar{\xi}_{\mathbf{R}^2}\mathbf{R}^{-1})\mathbf{R}. \tag{4.13}$$

One should observe that this retraction is not the exponential mapping of $S_+(r, d)$. This illustrates the interest of considering more general retractions than the exponential mapping. Indeed, as discussed in the paper of Bonnabel and Sepulchre (2009), the geodesics (and therefore the exponential mapping) do not appear to have a closed form in the considered geometry. Combining the gradient of (4.10) with the retractions (4.12) and (4.13) gives us

$$\begin{aligned}\mathbf{U}_{t+1} &= \mathrm{qf}\left(\mathbf{U}_t - 2\lambda s_t(\hat{y}_t - y_t)(\mathbf{I} - \mathbf{U}_t\mathbf{U}_t^T)\mathrm{Sym}(\mathbf{X}_t)\mathbf{U}_t\mathbf{R}_t^2\right),\\ \mathbf{R}_{t+1}^2 &= \mathbf{R}_t\exp\left(-(1-\lambda)s_t(\hat{y}_t - y_t)\mathbf{R}_t\mathbf{U}_t^T\mathrm{Sym}(\mathbf{X}_t)\mathbf{U}_t\mathbf{R}_t\right)\mathbf{R}_t.\end{aligned}$$

A factorization $\mathbf{R}_{t+1}\mathbf{R}_{t+1}^T$ of $\mathbf{R}_{t+1}^2$ is obtained thanks to the property of matrix exponential, $\exp(\mathbf{A})^{\frac{1}{2}} = \exp(\frac{1}{2}\mathbf{A})$. Updating $\mathbf{R}_{t+1}$ instead of $\mathbf{R}_{t+1}^2$ is thus more efficient from a computational point of view, since it avoids the computation of a square root at each iteration. This yields the online gradient descent algorithm

$$\begin{aligned}\mathbf{U}_{t+1} &= \mathrm{qf}\left(\mathbf{U}_t - 2\lambda s_t(\hat{y}_t - y_t)(\mathbf{I} - \mathbf{U}_t\mathbf{U}_t^T)\mathrm{Sym}(\mathbf{X}_t)\mathbf{U}_t\mathbf{R}_t^2\right),\\ \mathbf{R}_{t+1} &= \mathbf{R}_t\exp\left(-\frac{1}{2}(1-\lambda)s_t(\hat{y}_t - y_t)\mathbf{R}_t^T\mathbf{U}_t^T\mathrm{Sym}(\mathbf{X}_t)\mathbf{U}_t\mathbf{R}_t\right),\end{aligned} \tag{4.14}$$

and the batch gradient descent algorithm

$$\begin{aligned}\mathbf{U}_{t+1} &= \mathrm{qf}\left(\mathbf{U}_t - 2\lambda s_t\frac{1}{n}\sum_{i=1}^n(\hat{y}_i - y_i)(\mathbf{I} - \mathbf{U}_t\mathbf{U}_t^T)\mathrm{Sym}(\mathbf{X}_i)\mathbf{U}_t\mathbf{R}_t^2\right),\\ \mathbf{R}_{t+1} &= \mathbf{R}_t\exp\left(-\frac{1}{2}(1-\lambda)s_t\frac{1}{n}\sum_{i=1}^n(\hat{y}_i - y_i)\mathbf{R}_t^T\mathbf{U}_t^T\mathrm{Sym}(\mathbf{X}_i)\mathbf{U}_t\mathbf{R}_t\right).\end{aligned} \tag{4.15}$$

4.4 Algorithms

This section documents implementation details of the proposed algorithms. Generic pseudo-codes are provided in Figure 4.1 and Table 4.1 summarizes computational complexities.

Data	Input space	Batch flat (4.4)	Batch polar (4.15)	Online flat (4.3)	Online polar (4.14)
$\mathbf{X}$	$\mathbb{R}^{d\times d}$	$O(d^2rn)$	$O(d^2r^2n)$	$O(d^2rb)$	$O(d^2r^2b)$
$\mathbf{x}\mathbf{x}^T$	$\mathbb{R}^d$	$O(drn)$	$O(dr^2n)$	$O(drb)$	$O(dr^2b)$

Table 4.1: Computational costs of the proposed algorithms.

Batch regression	Online regression
Input: $\{(\mathbf{X}_i, y_i)\}_{i=1}^n$	**Input:** $\{(\mathbf{X}_t, y_t)\}_{t\geq 1}$
Require: $\mathbf{G}_0$ or $(\mathbf{U}_0, \mathbf{R}_0)$, λ	**Require:** $\mathbf{G}_0$ or $(\mathbf{U}_0, \mathbf{R}_0)$, λ, b, s, t_0, T
$t = 0$	$t = 0$, `count` $= b$
repeat	**while** $t \leq T$ **do**
	if `count` > 0 **then**
	Accumulate gradient
	`count` $=$ `count` $- 1$
	else
Compute Armijo step s_A from (4.16)	Compute step size s_t according to (4.17)
Perform update (4.4) or (4.15) using s_A	Perform update (4.3) or (4.14) using s_t
	`count` $= b$
	end if
$t = t + 1$	$t = t + 1$
until stopping criterion (4.18) is satisfied	**end while**
return $\mathbf{G}_t$	**return** $\mathbf{G}_T$

Figure 4.1: Pseudo-codes for the proposed batch and online algorithms.

4.4.1 From subspace learning to distance learning

The update expressions (4.15) and (4.14) show that λ, the tuning parameter of the Riemannian metric (4.11), acts as a weighting factor on the search direction. A proper tuning of this parameter allows us to place more emphasis either on the learning of the subspace $\mathbf{U}$ or on the distance in that subspace $\mathbf{R}^2$. In the case $\lambda = 1$, the algorithm only performs subspace learning. Conversely, in the case $\lambda = 0$, the algorithm learns a distance for a fixed range space (see Section 4.5.1). Intermediate values of λ continuously interpolate between the subspace learning problem and the distance learning problem at fixed range space.

Possible reasons motivating a particular choice for λ are when a good estimate of the subspace is available (for instance a subspace given by a proper dimension reduction technique) or when too few observations are available to jointly estimate the subspace and the distance within that subspace. In the latter case, one has the choice to favor either subspace or distance learning.

Experimental results of Section 4.6 recommend the value $\lambda = 0.5$ as the default setting.

4.4.2 Invariance properties

A nice property of the proposed algorithms is that they are invariant with respect to a rotation transform $\mathbf{W} \mapsto \mathbf{O}^T\mathbf{W}\mathbf{O}$, $\forall \mathbf{O} \in \mathcal{O}(d)$. This invariance comes from the fact that the chosen metrics are invariant to rotations. A practical consequence is that a rotation of the input matrix $\mathbf{X} \mapsto \mathbf{O}\mathbf{X}\mathbf{O}^T$ (for instance a whitening transformation of the vectors $\mathbf{x} \mapsto \mathbf{O}\mathbf{x}$ if $\mathbf{X} = \mathbf{x}\mathbf{x}^T$) will not affect the behavior of the algorithms.

Besides being invariant to rotations, algorithms (4.14) and (4.15) are invariant with respect to scalings $\mathbf{W} \mapsto \mu\mathbf{W}$ with $\mu > 0$. Consequently, a scaling of the input data $(\mathbf{X}, y) \mapsto (\mu\mathbf{X}, \mu y)$, such as a change of units, will not affect the behavior of these algorithms.

4.4.3 Mini-batch extension of online algorithms

A classical speedup and stabilization heuristic for stochastic gradient algorithms is to perform each gradient step with respect to $b \geq 1$ examples at a time instead of a single one. In the particular case $b = 1$, one recovers plain stochastic gradient descent. Given a mini-batch of b samples $(\mathbf{X}_{t,1}, y_{t,1}), ..., (\mathbf{X}_{t,b}, y_{t,b})$, received at time t, the generic online update on manifolds (3.10) becomes

$$\mathbf{W}_{t+1} = R_{\mathbf{W}_t}\left(-s_t \, \frac{1}{b}\sum_{i=1}^{b} \operatorname{grad} \ell(\hat{y}_{t,i}, y_{t,i})\right).$$

4.4.4 Strategies for choosing the step size

We here present strategies for choosing the step size in both the batch and online cases.

Batch algorithms

For batch algorithms, classical backtracking methods exist (see Nocedal and Wright, 2006). For the experiments of this chapter, we use the Armijo rule, that is, at each iteration, we choose a step size s_A satisfying the condition

$$f(R_{\mathbf{W}_t}(-s_A \operatorname{grad} f(\mathbf{W}_t))) \leq f(\mathbf{W}_t) - c\, s_A \|\operatorname{grad} f(\mathbf{W}_t)\|^2_{\mathbf{W}_t}, \tag{4.16}$$

where $\mathbf{W}_t \in S_+(r,d)$ is the current iterate, $c \in (0,1)$, f is batch cost function of interest and $R_{\mathbf{W}}$ is the chosen retraction. For the experiments of this chapter, we choose the particular value $c = 0.5$ and repetitively divide by 2 a specified maximum step size s_{max} until condition (4.16) is satisfied for the considered iteration. In order to reduce the dependence on s_{max} in a particular problem, it is chosen inversely proportional to the norm of the gradient at each iteration,

$$s_{max} = \frac{s_0}{\|\operatorname{grad} f(\mathbf{W}_t)\|_{\mathbf{W}_t}}.$$

A typical value of $s_0 = 100$ showed satisfactory results for all the considered problems.

Online algorithms

For online algorithms, the choice of the step size is more involved. In this chapter, the step size schedule s_t is chosen as

$$s_t = \frac{s}{\hat{\mu}_{grad}} \cdot \frac{nt_0}{nt_0 + t}, \tag{4.17}$$

where $s > 0$, n is the number of considered learning samples, $\hat{\mu}_{grad}$ is an estimate of the average gradient norm $\|\operatorname{grad} f(\mathbf{W}_0)\|_{\mathbf{W}_0}$, and $t_0 > 0$ controls the annealing rate of s_t. During a pre-training phase of our online algorithms, we select a small subset of learning samples and try the values 2^k with $k = -3, ..., 3$ for both s and t_0. The values of s and t_0 that provide the best decay of the cost function are selected to process the complete set of learning samples.

4.4.5 Stopping criterion

Batch algorithms are stopped when the value or the relative change of the empirical cost f is small enough, or when the relative change in the parameter variation is small enough,

$$f(\mathbf{W}_{t+1}) \leq \epsilon_{tol}, \quad \text{or} \quad \frac{f(\mathbf{W}_{t+1}) - f(\mathbf{W}_t)}{f(\mathbf{W}_t)} \leq \epsilon_{tol}, \quad \text{or} \quad \frac{\|\mathbf{G}_{t+1} - \mathbf{G}_t\|_F}{\|\mathbf{G}_t\|_F} \leq \epsilon_{tol}. \tag{4.18}$$

We found $\epsilon_{tol} = 10^{-5}$ to be a good trade-off between accuracy and convergence time.

Online algorithms are run for a fixed number of epochs (number of passes through the set of learning samples). Typically, a few epochs are sufficient to attain satisfactory results.

4.4.6 Convergence

Gradient descent algorithms on matrix manifolds share the well-characterized convergence properties of their analog in $\mathbb{R}^d$. Batch algorithms converge linearly to a local minimum of the empirical cost that depends on the initial condition. Online algorithms converge asymptotically to a local minimum of the expected loss. They intrinsically have a much slower convergence rate than batch algorithms, but they generally decrease faster the expected loss in the large-scale regime (Bottou and Bousquet, 2007).

When learning a matrix $\mathbf{W} \in S_{++}(d)$, the problem is convex and the proposed algorithms converge toward a global minimum of the cost function, regardless of the initial condition. When learning a low-rank matrix $\mathbf{W} \in S_+(r, d)$, with $r < d$, the proposed algorithms converge to a local minimum of the cost function. This is not the case for heuristic methods proposed in the literature, which first reduce the dimensionality of the data before fitting a full-rank model on the reduced data (Davis and Dhillon, 2008; Weinberger and Saul, 2009).

For batch algorithms, the local convergence results follow from the convergence theory of line-search algorithms on Riemannian manifolds (see, for example, Absil et al., 2008).

For online algorithms, a local convergence proof for stochastic gradient descent algorithms on Riemannian manifolds has been recently proposed by Bonnabel (2011). This convergence result generalizes the classical convergence result presented in the paper of Bottou (1998) on almost sure convergence, that is asymptotic convergence with probability one, of stochastic gradient algorithms in $\mathbb{R}^d$. The result of Bonnabel (2011) provides the technical adaptations of the convergence proof in $\mathbb{R}^d$ to the Riemannian case and derives sufficient conditions for almost sure convergence of a stochastic gradient descent algorithm on a Riemannian manifold $\mathcal{W}$.

The provided sufficient conditions guarantee almost sure convergence of the algorithm toward a local minimum of the expected cost function. The convergence is obtained in terms of the Riemannian (geodesic) distance associated with the considered Riemannian manifold.

The proof is directly inspired by the analogous proof in $\mathbb{R}^d$, but it further takes into account the non linear nature of the search space as well as important curvature effects on the manifold. In particular, a general form of adaptive step size is provided in order to guarantee that the trajectories remain almost surely bounded. This adaptive step size ensures the convergence of algorithms, but it is conservative and may lead to slow convergence. Experimental results indicate however that more conventional choices for the step size give good results in practice.

An ad hoc convergence proof for algorithm (4.3) with a conservative choice for the step size is provided in (Meyer et al., 2011a). It is reproduced in Appendix A for the reader's convenience.

Due to the nonconvex nature of the considered rank-constrained problems, the convergence results are only local and little can be presently said about the global convergence of the algorithms. A global analysis of the critical points of the cost functions studied in the present chapter is nevertheless not hopeless and could be facilitated by the considered low-rank parameterizations. For instance, global convergence properties have been established for PCA algorithms from an explicit analysis of the critical points (Chen et al., 1998). Also, recent results suggest good global convergence properties for closely related rank minimization problems (Recht et al., 2010). Experimental results suggest the same conclusions for the algorithms considered in this chapter, which means that further research on global convergence results is certainly deserved.

4.5 Discussion

This section presents connections with existing works and extensions of the regression model.

4.5.1 Connection with closeness-based approaches

A standard derivation of learning algorithms is as follows (Kivinen and Warmuth, 1997). The (online) update at time t is viewed as an (approximate) solution of

$$\mathbf{W}_{t+1} = \operatorname*{arg\,min}_{\mathbf{W} \in \mathcal{W}} \; D(\mathbf{W}, \mathbf{W}_t) \; + \; s_t \; \ell(\hat{y}, y_t), \tag{4.19}$$

where D is a well-chosen measure of closeness between elements of $\mathcal{W}$ and s_t is a trade-off parameter that controls the balance between the conservative term $D(\mathbf{W}, \mathbf{W}_t)$ and the innovation (or data fitting) term $\ell(\hat{y}, y_t)$. One solves (4.19) by solving the algebraic equation

$$\operatorname{grad} D(\mathbf{W}, \mathbf{W}_t) = -s_t \operatorname{grad} \ell(\hat{y}_{t+1}, y_t), \tag{4.20}$$

which is a first-order (necessary) optimality condition. If the search space $\mathcal{W}$ is a Riemannian manifold and if the closeness measure $D(\mathbf{W}, \mathbf{W}_t)$ is the halved squared Riemannian distance, the solution of (4.20) is

$$\mathbf{W}_{t+1} = \mathrm{Exp}_{\mathbf{W}_t}(-s_t \operatorname{grad} \ell(\hat{y}_{t+1}, y_t)).$$

Because $\hat{y}_{t+1}$ must be evaluated in $\mathbf{W}_{t+1}$, this update equation is implicit. However, $\hat{y}_{t+1}$ is generally replaced by $\hat{y}_t$ (which is equal to $\hat{y}_{t+1}$ up to first order terms in s_t), which gives the update (3.10) where the exponential mapping is chosen as a retraction.

Bregman divergences have been popular closeness measures for $D(\mathbf{W}, \mathbf{W}_t)$ because they render the optimization of (4.19) convex. Bregman divergences on the cone of positive definite matrices include the von Neumann divergence

$$D_{vN}(\mathbf{W}, \mathbf{W}_t) = \mathrm{Tr}(\mathbf{W} \log \mathbf{W} - \mathbf{W} \log \mathbf{W}_t - \mathbf{W} + \mathbf{W}_t),$$

and the LogDet divergence

$$D_{ld}(\mathbf{W}, \mathbf{W}_t) = \mathrm{Tr}(\mathbf{W}\mathbf{W}_t^{-1}) - \log\det(\mathbf{W}\mathbf{W}_t^{-1}) - d.$$

We have shown in Section 4.2 that the resulting updates can be interpreted as line-search updates for the log-Euclidean metric and the affine-invariant metric of $S_{++}(d)$ and for specific choices of the retraction mapping.

Likewise, the algorithm (4.3) can be recast in the framework (4.19) by considering the closeness

$$D_{flat}(\mathbf{W}, \mathbf{W}_t) = \|\mathbf{G} - \mathbf{G}_t\|_F^2,$$

where $\mathbf{W} = \mathbf{G}\mathbf{G}^T$ and $\mathbf{W}_t = \mathbf{G}_t\mathbf{G}_t^T$. Algorithm (4.14) can be recast in the framework (4.19) by considering the closeness

$$D_{pol}(\mathbf{W}, \mathbf{W}_t) = \lambda \sum_{i=1}^{r} \theta_i^2 \; + \; (1-\lambda) \; \| \log \mathbf{R}_t^{-1}\mathbf{R}^2\mathbf{R}_t^{-1} \|_F^2.$$

where the θ_i's are the principal angles between the subspaces spanned by $\mathbf{W}$ and $\mathbf{W}_t$ (Golub and Van Loan, 1996), and the second term is the affine-invariant distance of $S_{++}(d)$ between matrices $\mathbf{R}^2$ and $\mathbf{R}_t^2$ involved in the polar representation of $\mathbf{W}$ and $\mathbf{W}_t$.

Obviously, these closeness measures are no longer convex due to the rank constraint. However they reduce to the popular divergences in the full-rank case, up to second order terms. In particular, when $\lambda = 1$, the subspace is fixed and one recovers the setup of learning low-rank matrices of a fixed range space (Kulis et al., 2009). Thus, the algorithms introduced in the present chapter can be viewed as generalizations of the ones presented in the paper of Kulis et al. (2009), where the issue of adapting the range space is presented as an open research question. Each of the proposed algorithms provides an efficient workaround for this problem at the expense of the (potential) introduction of local minima.

4.5.2 Kernelizing the regression model

In this chapter, we have not considered the kernelized model

$$\hat{y} = \mathrm{Tr}(\mathbf{W}\phi(\mathbf{x})\phi(\mathbf{x})^T),$$

whose predictions can be extended to new input data $\phi(\mathbf{x})$ in the feature space $\mathcal{F}$ induced by the nonlinear mapping $\phi : \mathbf{x} \in \mathcal{X} \mapsto \phi(\mathbf{x}) \in \mathcal{F}$. This is potentially a useful extension of the regression model that could be investigated in the light of recent theoretical results in this area (for example Chatpatanasiri et al., 2010; Jain et al., 2010).

4.5.3 Connection with multidimensional scaling algorithms

Given a set of m dissimilarity measures $\mathcal{D} = \{\delta_{ij}\}^m$ between n data objects, multidimensional scaling algorithms search for a r-dimensional embedding of the data objects into an Euclidean space representation $\mathbf{G} \in \mathbb{R}^{n \times r}$ (Cox and Cox, 2001; Borg and Groenen, 2005). Each row $\mathbf{g}$ of $\mathbf{G}$ is the coordinates of a data object in a Euclidean space of dimension r.

Multidimensional scaling algorithms based on gradient descent are equivalent to algorithms (4.3) and (4.4) when $\mathbf{X} = (\mathbf{e}_i - \mathbf{e}_j)(\mathbf{e}_i - \mathbf{e}_j)^T$, where $\mathbf{e}_i$ is the i-th unit vector (see Section 2.1.1), and when the multidimensional scaling reduction criterion is the SSTRESS

$$SSTRESS(\mathbf{G}) = \sum_{(i,j)\in\mathcal{D}} (\|\mathbf{g}_i - \mathbf{g}_j\|_2^2 - \delta_{ij})^2.$$

Vectors $\mathbf{g}_i$ and $\mathbf{g}_j$ are the i-th and j-th rows of matrix $\mathbf{G}$. Gradient descent is a popular technique in the context of multidimensional scaling algorithms. A stochastic gradient descent approach for minimizing the SSTRESS has also been proposed by Matsuda and Yamaguchi (2001). A potential area of future work is the application of the proposed online algorithm (4.3) for adapting a batch solution to slight modifications of the dissimilarities over time. This approach would have a much smaller computational cost than recomputing the offline solution at every time step. It further allows to keep the coordinate representation coherent over time since the solution does not brutally jump from one local minimum to another.

4.6 Experiments

Data Set	Samples	Features	Classes	Reference
GyrB	52	-	3	Tsuda et al. (2005)
Digits	300	16	3	Asuncion and Newman (2007)
Wine	178	13	13	Asuncion and Newman (2007)
Ionosphere	351	33	2	Asuncion and Newman (2007)
Balance Scale	625	4	3	Asuncion and Newman (2007)
Iris	150	4	3	Asuncion and Newman (2007)
Soybean	532	35	17	Asuncion and Newman (2007)
USPS	2,007	256	10	LeCun et al. (1989)
Isolet	7,797	617	26	Asuncion and Newman (2007)
Prostate	322	15,154	2	Petricoin et al. (2002)

Table 4.2: Considered data sets

In this section, we illustrate the potential of the proposed algorithms on several benchmark experiments. First, the proposed algorithms are evaluated on toy data. Then, they are compared to state-of-the-art kernel learning and Mahalanobis distance learning algorithms on real data sets. Overall, the experiments support that a joint estimation of a subspace and low-dimensional distance in that subspace is a major advantage of the proposed algorithms over methods that estimate the matrix for a subspace that is fixed beforehand.

Table 4.2 summarizes the different data sets that have been considered. As a normalization step, the data features are centered and rescaled to unit standard deviation.

The implementation of the proposed algorithms,[1] as well as the experiments of this chapter are performed with Matlab. The implementations of algorithms MVU,[2] KSR,[3] LMNN,[4] and ITML,[5] have been rendered publicly available by Weinberger et al. (2004), Cai et al. (2007),

[1]The source code is available from `http://www.montefiore.ulg.ac.be/~meyer`.
[2]MVU is available from `http://www.cse.wustl.edu/~kilian/Downloads/MVU.html`.
[3]KSR is available from `http://www.cs.uiuc.edu/homes/dengcai2/SR/`.
[4]LMNN is available from `http://www.cse.wustl.edu/~kilian/Downloads/LMNN.html`.
[5]ITML is available from `http://www.cs.utexas.edu/users/pjain/itml/`.

Weinberger and Saul (2009) and Davis et al. (2007) respectively. Algorithms POLA (Shalev-Shwartz et al., 2004), LogDet-KL (Kulis et al., 2009) and LEGO (Jain et al., 2008) have been implemented on our own.

4.6.1 Toy data

In this section, the proposed algorithms are evaluated on synthetic regression problems. The data vectors $\mathbf{x}_1, ..., \mathbf{x}_n \in \mathbb{R}^d$ and the target matrix $\mathbf{W}^* \in S_+(r, d)$ are generated with entries drawn from a standard Gaussian distribution $\mathcal{N}(0, 1)$. Observations follow

$$y_i = (\mathbf{x}_i^T \mathbf{W}^* \mathbf{x}_i)(1 + \nu_i), \quad i = 1, ..., n, \tag{4.21}$$

where ν_i is drawn from $\mathcal{N}(0, 0.01)$. A multiplicative noise model is preferred over an additive one to easily control that observations remain nonnegative after the superposition of noise.

Batch algorithms minimize the cost function

$$f(\mathbf{W}) = \frac{1}{n} \sum_{i=1}^{n} (\hat{y}_i - y_i)^2 = \frac{1}{n} \sum_{i=1}^{n} (\mathbf{x}_i^T \mathbf{W} \mathbf{x}_i - y_i)^2.$$

At time t, online algorithms minimize the cost function

$$f_t(\mathbf{W}) = \frac{1}{b} \sum_{\tau=t}^{t+b} (\mathbf{x}_\tau^T \mathbf{W} \mathbf{x}_\tau - y_\tau)^2,$$

where b is the mini batch size.

Learning the subspace vs. fixing the subspace up front

As an illustrative example, we show the difference between two approaches for fitting the data to observations when a target model $\mathbf{W}^* \in S_+(3, 3)$ is approximated with a parameter $\mathbf{W} \in S_+(2, 3)$.

A naive approach to tackle that problem is to first project the data $\mathbf{x}_i \in \mathbb{R}^3$ on a subspace of reduced dimension and then to compute a full-rank model based on the projected data. Recent methods compute that subspace of reduced dimension using principal component analysis (Davis and Dhillon, 2008; Weinberger and Saul, 2009), that is, a subspace that captures a maximal amount of variance in the data. However, in general, there is no reason why the subspace spanned by the top principal components should coincide with the subspace that is defined by the target model. Therefore, a more appropriate approach consists in learning jointly the subspace and a distance in that subspace that best fits the data to observations within that subspace.

To compare the two approaches, we generate a set of learning samples $\{(\mathbf{x}_i, y_i)\}_{i=1}^{200}$, with $\mathbf{x}_i \in \mathbb{R}^3$ and y_i that follows (4.21). The target model is

$$\mathbf{W}^* = \tilde{\mathbf{U}} \mathbf{\Lambda} \tilde{\mathbf{U}}^T$$

where $\tilde{\mathbf{U}}$ is a random 3×3 orthogonal matrix and $\mathbf{\Lambda}$ is a diagonal matrix with two dominant values $\Lambda_{11}, \Lambda_{22} \gg \Lambda_{33} > 0$ (for this specific example, $\Lambda_{11} = 4, \Lambda_{22} = 3$ and $\Lambda_{33} = 0.01$). Observations y_i are thus nearly generated by a rank-2 model, such that $\mathbf{W}^*$ should be well approximated with a matrix $\mathbf{W} \in S_+(2, 3)$ that minimizes the train error.

Results are presented in Figure 4.2. The top plot shows that the learned subspace (which identifies with the target subspace) is indeed very different from the subspace spanned by the top two principal components. Moreover, the bottom plots clearly demonstrate that the fit is much better when the subspace and the distance in that subspace are learned jointly. The difference is also significant in terms of the train error. This simple example shows that heuristic methods

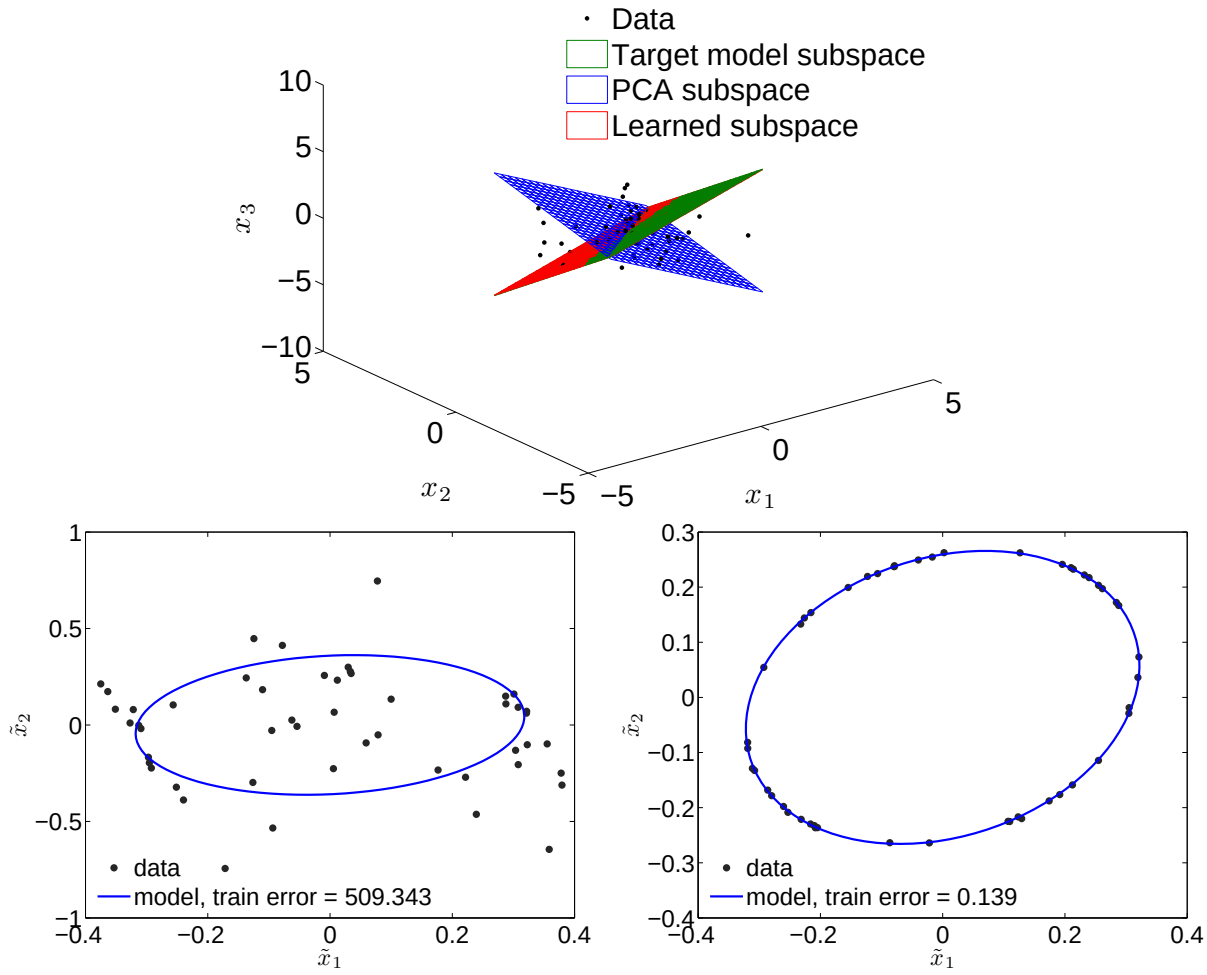


Figure 4.2: Learning vs fixing the subspace. **Top**: the learned subspace is very different from the subspace computed from a classical heuristic. **Bottom left**: fit after projection of the data onto a subspace fixed up front. **Bottom right**: fit obtained with a joint estimation of the subspace and a distance within that subspace. This figure is best viewed with colors.

that fix the range space in the first place may converge to a solution that is very different from a minimum of the desired cost function. For visualization purpose, the two dimensional model is represented by the ellipse

$$\mathcal{E} = \{\tilde{\mathbf{x}}_i \in \mathbb{R}^2 : \tilde{\mathbf{x}}_i^T \mathbf{R}^2 \tilde{\mathbf{x}}_i = 1\}, \qquad \text{where} \quad \tilde{\mathbf{x}}_i = \frac{\mathbf{U}^T \mathbf{x}_i}{\sqrt{y_i}},$$

and $(\mathbf{U}, \mathbf{R}^2)$ are computed with algorithm (4.15), either in the setting $\lambda = 0$ that fixes the subspace to the PCA subspace (left) or in the setting $\lambda = 0.5$ that simultaneously learned $\mathbf{U}$ and $\mathbf{B}$ (right). A perfect fit is obtained when all $\tilde{\mathbf{x}}_i$ are located on $\mathcal{E}$, which is the locus of points where $\hat{y}_i = y_i$.

Influence of λ on the algorithm based on the polar geometry

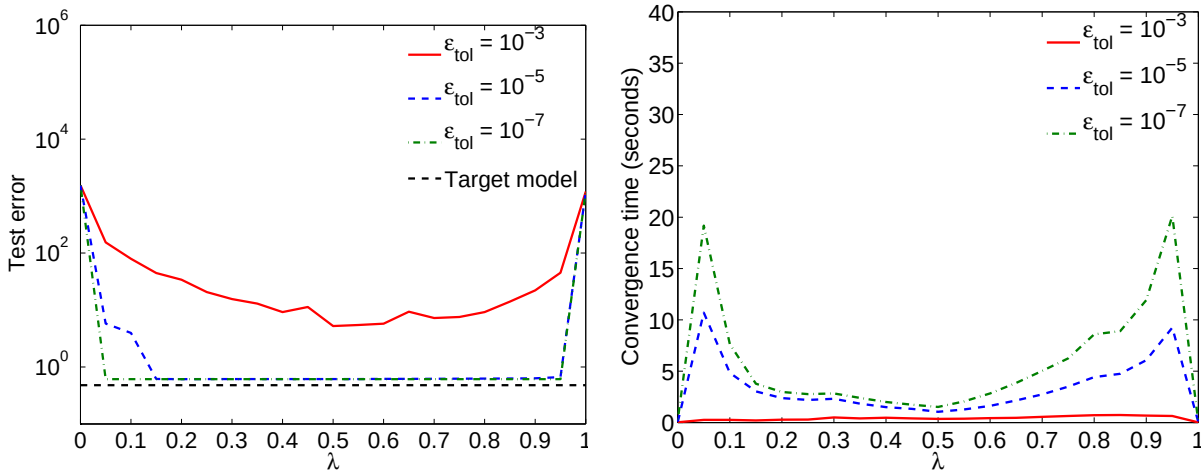


Figure 4.3: Influence of λ.

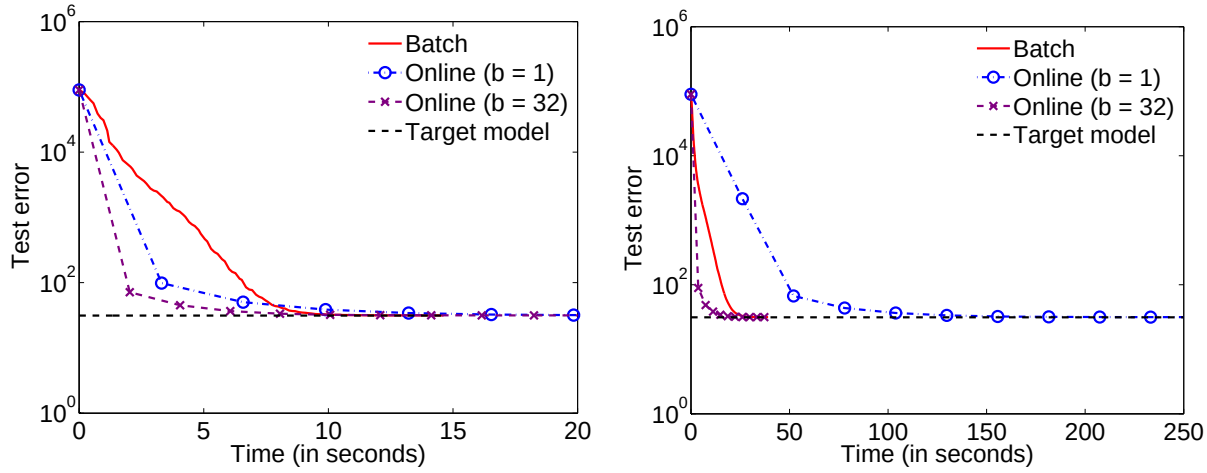
In theory, the parameter λ should not influence the algorithm since it has no effect on the first-order optimality conditions except for its two extreme values $\lambda = 0$ and $\lambda = 1$. In practice however, a sensitivity to this parameter is observed due to the finite tolerance of the stopping criterion: the looser the tolerance, the more sensitive the result to λ.

To investigate the sensitivity to λ, we try to recover a target parameter $\mathbf{W}^* \in S_+(5, 10)$ using pairs $(\mathbf{x}_i, y_i)$ generated according to (4.21). We generate 10 random regression problems with 1000 samples partitioned into 500 learning samples and 500 test samples. We compute the mean test error and the mean convergence time as a function of λ for different values of ϵ_{tol}. The results are presented in Figure 4.3. As ϵ_{tol} decreases, the test error becomes insensitive to λ, but an influence is observed on the convergence time of the algorithm.

In view of these results, we recommend the value 0.5 as the default setting for λ. Unless specified otherwise, we therefore use this particular value for all experiments in this chapter.

Online vs. batch

This experiment shows that when a large amount of samples is available ($80,000$ training samples and $20,000$ test samples for learning a parameter $\mathbf{W}^*$ in $S_+(10, 50)$), online algorithms minimize the test error more rapidly than batch ones. It further shows that the mini-batch extension allows to significantly improve the performance compared to the plain stochastic gradient descent setting ($b = 1$). We observe that the mini-batch size $b = 32$ generally gives good results. Figure 4.4 reports the test error as a function of the learning time. For the algorithm based on the polar geometry, the mini-batch extension is strongly recommended to amortize the larger cost of each update.



(a) Flat geometry (b) Polar geometry

Figure 4.4: Online vs Batch. For a large number of samples, online algorithms reduce the test error much more rapidly than batch ones. Using the mini-batch extension generally improves the performance significantly.

4.6.2 Kernel learning

In this section, the proposed algorithms are applied to the problem of learning a kernel matrix from pairwise distance constraints between data samples. As mentioned earlier, we only consider this problem in the transductive setting, that is, all samples $\mathbf{x}_1, ...\mathbf{x}_n$ are available up front and the learned kernel does not generalize to new samples.

Experimental setup

After transformation of the data with the kernel map $\mathbf{x} \mapsto \phi(\mathbf{x})$, the purpose is to compute a fixed-rank kernel matrix based on a limited amount of pairwise distances in the kernel feature space and on some information about class labels.

Following the setup presented in Section 2.1.1, distance constraints are generated as $\hat{y}_{ij} \leq y_u$ for identically labeled samples and $\hat{y}_{ij} \geq y_l$ for differentially labeled samples. The right hand sides of these inequalities are chosen as $y_u = y_{ij}(1-\alpha)$ and $y_l = y_{ij}(1+\alpha)$, where $0 \leq \alpha < 1$ is a scaling factor, $y_{ij} = \|\phi(\mathbf{x}_i) - \phi(\mathbf{x}_j)\|^2$ and $\hat{y}_{ij} = \mathrm{Tr}(\mathbf{W}(\mathbf{e}_i - \mathbf{e}_j)(\mathbf{e}_i - \mathbf{e}_j)^T) = (\mathbf{e}_i - \mathbf{e}_j)^T\mathbf{W}(\mathbf{e}_i - \mathbf{e}_j)$. To handle inequalities, we use the cost function (2.3) either in its batch or online version.

We investigate both the influence of the amount of side-information provided, the influence of the approximation rank and the computational time required by the algorithms.

To quantify the performance of the learned kernel matrix, we perform either a classification or a clustering of the samples based on the learned kernel. For classification, we compute the test set accuracy of a k-nearest neighbor classifier ($k = 5$) using a two-fold cross-validation protocol (results are averaged over 10 random splits). For clustering, we use the K-means algorithm with the number of clusters equal to the number of classes in the problem. To overcome K-means local minima, 10 runs are performed in order to select the result that has lead to the smallest value of the K-means objective. The quality of the clustering is measured by the normalized mutual information (NMI) shared between the random variables of cluster indicators C and target labels T (Strehl et al., 2000),

$$NMI = \frac{2\ I(C;T)}{(H(C) + H(T))},$$

where $I(X_1; X_2) = H(X_1) - H(X_1|X_2)$ is the mutual information between the random variables

X_1 and X_2, $H(X_1)$ is the Shannon entropy of X_1, and $H(X_1|X_2)$ is the conditional entropy of X_1 given X_2. This score ranges from 0 to 1, the larger the score, the better the clustering quality.

Compared methods

We compare the following methods:

1. Batch algorithms (4.4) and (4.15), adapted to handle inequalities (cost function (2.3)),
2. The kernel learning algorithm LogDet-KL (Kulis et al., 2009) which learn kernel matrices of fixed range space for a given set of distance constraints.
3. The kernel spectral regression (KSR) algorithm of Cai et al. (2007) using a similarity matrix $\mathbf{N}$ constructed as follows. Let $\mathbf{N}$ be the adjacency matrix of a 5-NN graph based on the initial kernel. We modify $\mathbf{N}$ according to the set of available constraints: $\mathbf{N}_{ij} = 1$ if samples $\mathbf{x}_i$ and $\mathbf{x}_j$ belong to the same class (must-link constraint), $\mathbf{N}_{ij} = 0$ if samples $\mathbf{x}_i$ and $\mathbf{x}_j$ do not belong to the same class (cannot-link constraint).
4. The Maximum Variance Unfolding (MVU) algorithm (Weinberger et al., 2004),
5. The Kernel PCA algorithm (Schölkopf et al., 1998).

The last two algorithms are unsupervised techniques that are provided as baselines.

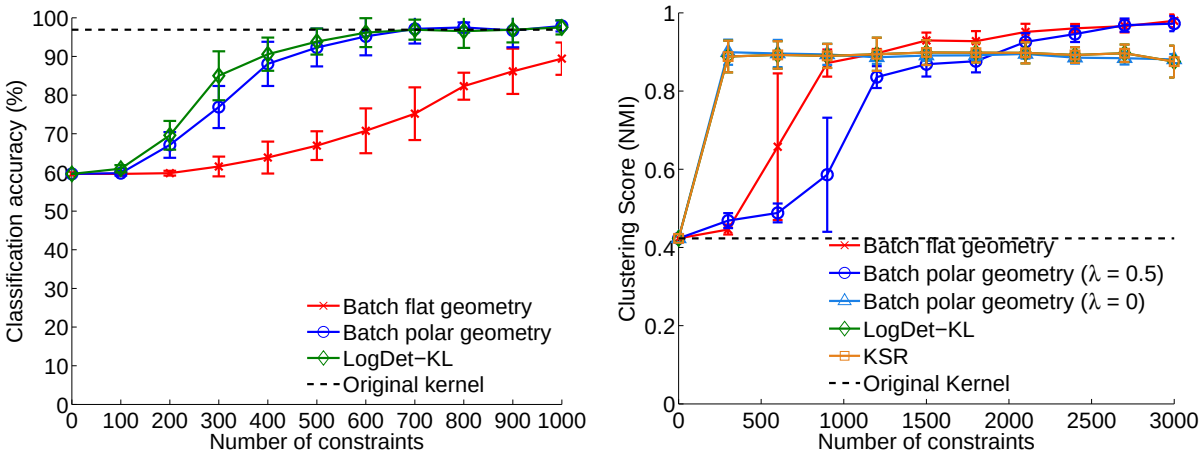


Figure 4.5: **Left**: full-rank kernel learning on the Gyrb data set. The algorithm based on the polar geometry competes with LogDet-KL. **Right**: low-rank kernel learning on the Digits data set. The proposed algorithms outperform the compared methods when a sufficiently large number of constraints is provided.

Results

The first experiment is reproduced from Tsuda et al. (2005) and Kulis et al. (2009). The goal is to reconstruct the GyrB kernel matrix based on distance constraints only. This matrix contains information about the proteins of three bacteria species. The distance constraints are randomly generated from the original kernel matrix with $\alpha = 0$. We compare the proposed batch methods with the LogDet-KL algorithm, the only competing algorithm that also learns directly from distance constraints. This algorithm is the best performer reported by Kulis et al. (2009) for this experiment. All algorithms start from the identity matrix that does not encode any domain information. Figure 4.5 (left) reports the k-NN classification accuracy as a function of the

number of distance constraints provided. In this full-rank learning setting, the algorithm based on the polar geometry competes with the LogDet-KL algorithm. The convergence time of the algorithm based on the polar geometry is however much faster (0.15 seconds versus 58 seconds for LogDet-KL when learning 1000 constraints). The algorithm based on the flat geometry has inferior performance when too few constraints are provided. This is because in the kernel learning setting, updates of this algorithm only involve the rows and columns that correspond to the set of points for which constraints are provided. It may thus result in a partial update of the kernel matrix entries. This issue disappears as the number of provided constraints increases.

The second experiment is reproduced from Kulis et al. (2009). It aims at improving an existing low-rank kernel using limited information about class labels. A rank-16 kernel matrix is computed for clustering a database of 300 handwritten digits randomly sampled from the 3, 8 and 9 digits of the Digits data set (since we could not find out the specific samples that have been selected by Kulis et al. (2009), we made our own samples selection). The distance constraints are randomly sampled from a linear kernel on the input data $\mathbf{K} = \mathbf{X}\mathbf{X}^T$ and $\alpha = 0.25$. The results are presented in Figure 4.5 (right). The figure shows that KSR, LogDet-KL and the algorithm based on the polar geometry with $\lambda = 0$ perform similarly. These methods are however outperformed by the proposed algorithms (flat geometry and polar geometry with $\lambda = 0.5$) when the number of constraints is large enough. This experiment also enlightens the flexibility of the polar geometry, which allows us to fix the subspace in situations where too few constraints are available.

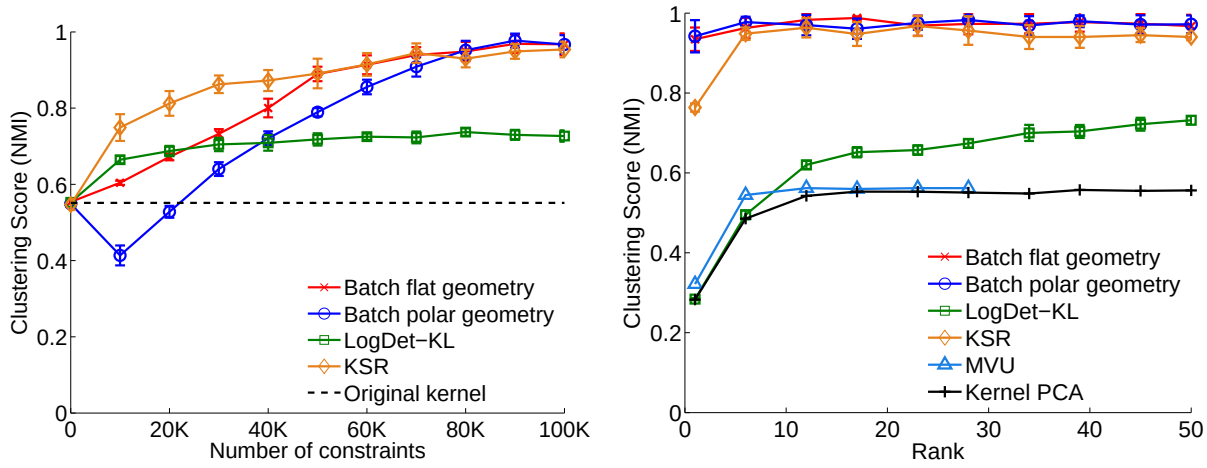


Figure 4.6: Clustering the USPS data set. **Left:** clustering score versus number of constraints. **Right:** clustering score versus approximation rank. When the number of provided constraints is large enough, the proposed algorithms perform as good as the KSR algorithm. It outperforms the LogDet-KL algorithm and baselines.

Finally, we tackle the kernel learning problem on a larger data set. We use the test set of the USPS data set,[6] which contains 2007 samples of handwritten zip code digits. The data are first transformed using the kernel map $\kappa(\mathbf{x}_i, \mathbf{x}_j) = \exp(-\gamma \|\mathbf{x}_i - \mathbf{x}_j\|_2^2)$ with $\gamma = 0.001$ and we further center the data in the kernel feature space. Pairwise distance constraints are randomly sampled from that kernel matrix with $\alpha = 0.5$. Except KSR that has its own initialization procedure, algorithms start from the kernel matrix provided by kernel PCA.

Figure 4.6 (left) shows the clustering performance as a function of the number of constraints provided when the approximation rank is fixed to $r = 25$. Figure 4.6 (right) reports the clustering performance as a function of the approximation rank when the number of constraints provided is fixed to $100K$. When the number of provided constraints is large enough, the proposed algorithms perform as well as KSR and outperform the LogDet-KL method that learns a kernel

[6] We use the ZIP code data from `http://www-stat-class.stanford.edu/~tibs/ElemStatLearn/data.html`.

of fixed-range space. Average computational times for learning a rank-6 kernel from $100K$ constraints are 0.57 seconds for KSR, 3.25 seconds for the algorithm based on the flat geometry, 46.78 seconds for LogDet-KL and 47.30 seconds for the algorithm based on the polar geometry. In comparison, the SDP-based MVU algorithm takes 676.60 seconds to converge.

4.6.3 Mahalanobis distance learning

In this section, we tackle the problem of learning from data a Mahalanobis distance for supervised classification and compare our methods to state-of-the-art Mahalanobis distance learning algorithms.

Experimental setup

For the considered problem, the purpose is to learn the parameter $\mathbf{W}$ of a Mahalanobis distance $d_{\mathbf{W}}(\mathbf{x}_i, \mathbf{x}_j) = (\mathbf{x}_i - \mathbf{x}_j)^T \mathbf{W}(\mathbf{x}_i - \mathbf{x}_j)$, such that the distance satisfies as much as possible a given set of constraints. As in the paper of Davis et al. (2007), we generate the constraints from the learning set of samples as $d_{\mathbf{W}}(\mathbf{x}_i, \mathbf{x}_j) \leq y_u$ for same-class pairs and $d_{\mathbf{W}}(\mathbf{x}_i, \mathbf{x}_j) \geq y_l$ for different-class pairs. The scalars y_l and y_u estimate the 5^{th} and 95^{th} percentiles of the observed distribution of Mahalanobis distances parameterized by a chosen baseline $\mathbf{W}_0$ within a given data set. Again, we use the cost function (2.3) to handle inequalities. The performance of the learned distance is then quantified by the test error rate of a k-nearest neighbor classifier based on the learned distance. All experiments use the setting $k = 5$. Except for the Isolet data set for which a specific train/test partition is provided, error rates are computed using two-fold cross validation. Results are averaged over 10 random partitions.

Compared methods

We compare the following distance learning algorithms:

1. Batch algorithms (4.4) and (4.15),
2. ITML (Davis et al., 2007),
3. LMNN (Weinberger and Saul, 2009),
4. Online algorithms (4.3) and (4.14),
5. LEGO (Jain et al., 2008),
6. POLA (Shalev-Shwartz et al., 2004).

When some methods require the tuning of a hyper-parameter, this is performed by a two-fold cross-validation procedure. The slack parameter of ITML as well as the step size of POLA, are selected in the range of values 10^k with $k = -3, ..., 3$. The step size of LEGO is selected in this same range of values for the UCI data sets, and in the range of values 10^k with $k = -10, ..., -5$ for the larger data sets Isolet and Prostate.

Results

Reproducing a classical benchmark experiment from Kulis et al. (2009), we demonstrate that the proposed batch algorithms compete with state-of-the-art full-rank Mahalanobis distance learning algorithms on several UCI data sets (Figure 4.7). We have not included the online versions of our algorithms in this comparison because we consider that the batch approaches are more relevant on such small data sets. Except POLA and LMNN which do not learn from provided pairwise constraints, all algorithms process $40c(c-1)$ constraints, where c is the number

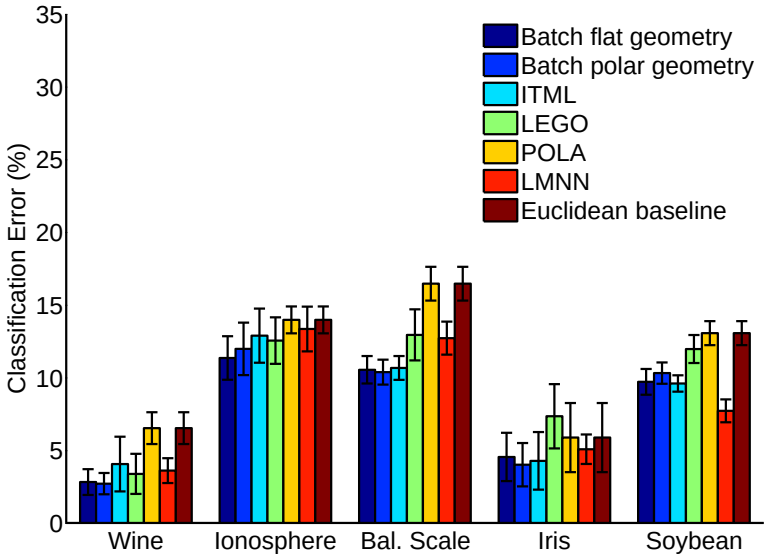
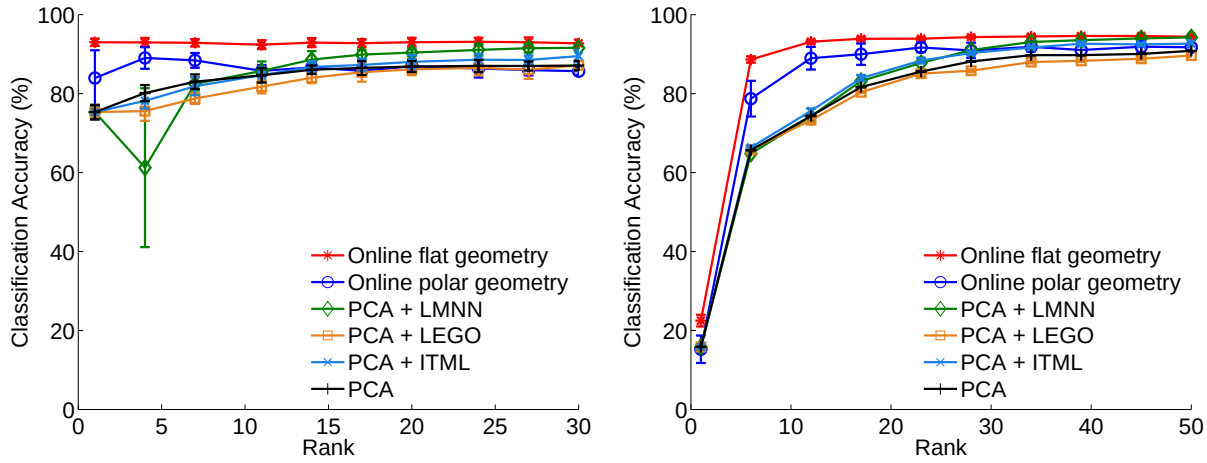


Figure 4.7: Full-rank distance learning on the UCI data sets. The proposed algorithms compete with state-of-the-art methods for learning a full-rank Mahalanobis distance.

of classes in the data. We choose the Euclidean distance ($\mathbf{W}_0 = \mathbf{I}$) as the baseline distance for initializing the algorithms. Figure 4.7 reports the results. The two proposed algorithms compete favorably with the other full-rank distance learning techniques, achieving the minimal average error for 4 of the 5 considered data sets.



(a) Prostate (b) Isolet

Figure 4.8: Low-rank Mahalanobis distance learning. For low values of the rank, the proposed algorithms perform much better than the methods that project the data on the top principal directions and learn a full-rank distance on the projected data.

We finally evaluate the proposed algorithms on higher-dimensional data sets in the low-rank regime (Figure 4.8). The distance constraints are generated as in the full-rank case, but the initial baseline matrix is now computed as $\mathbf{W}_0 = \mathbf{G}_0\mathbf{G}_0^T$, where $\mathbf{G}_0$'s columns are the top principal directions of the data. For the Isolet data set, $100K$ constraints are generated, and $10K$ constraints are generated for the Prostate data set. For scalability reasons, algorithms LEGO, LMNN and ITML must proceed in two steps: the data are first projected onto the top principal directions and then a full-rank distance is learned within the subspace spanned by these top principal directions. In contrast, our algorithms are initialized with the top principal direction, but they operate on the data in their original feature space. Overall, the proposed

algorithms achieve much better performance than the methods that first reduce the data. This is particularly striking when the rank is very small compared to problem size. The performance gap reduces as the rank increases. However, for high-dimensional problems, one is usually interested in efficient low-rank approximations that gives satisfactory results.

4.7 Conclusion

In this chapter, we propose gradient descent algorithms to learn a regression model parameterized by a fixed-rank positive semidefinite matrix. The Riemannian geometry of the set of fixed-rank PSD matrices is exploited through a geometric optimization approach.

The resulting algorithms overcome the main difficulties encountered by the previously proposed methods as they scale to high-dimensional problems, and they naturally enforce the rank constraint as well as the positive definite property while leaving the range space of the matrix free to evolve during optimization.

We apply the proposed algorithms to the problem of learning a distance function from data, when the distance is parameterized by a fixed-rank positive semidefinite matrix. The good performance of the proposed algorithms is illustrated on several kernel learning and distance learning benchmarks. In particular, the advantages of simultaneously learning the subspace and a distance within that subspace are illustrated.

Chapter 5

Regression on fixed-rank non-symmetric matrices

Chapter abstract: In this chapter, we address the problem of learning a linear regression model whose parameter is a fixed-rank non-symmetric matrix.

For this purpose, the algorithms of the previous chapter for fixed-rank symmetric positive semidefinite matrices are generalized to fixed-rank non-symmetric matrices. This is achieved by developing two novel quotient manifold geometries for the set of fixed-rank non-symmetric matrices.

The resulting algorithms apply to a broad range of applications, scale to high-dimensional problems, enjoy local convergence properties and confer a geometric basis to recent contributions on learning fixed-rank matrices. Numerical experiments on benchmarks show that the proposed algorithms compete with the state-of-the-art.

The material of this chapter is based on the following publication:

G. Meyer, S. Bonnabel and R. Sepulchre
Linear Regression under Fixed-Rank Constraints: A Riemannian Approach
In *Proc. of the 28th International Conference on Machine Learning*, Bellevue (USA), 2011.

5.1 Introduction

Learning a low-rank matrix from data is a fundamental problem arising in many modern machine learning applications: collaborative filtering (Rennie and Srebro, 2005), classification with multiple classes (Amit et al., 2007), learning on pairs (Abernethy et al., 2009), dimensionality reduction (Cai et al., 2007), learning of low-rank distances (Kulis et al., 2009; Meyer et al., 2011a) and low-rank similarity measures (Shalit et al., 2010), multi-task learning (Evgeniou et al., 2005), just to name a few. Parallel to the development of these new applications, the ever-growing size and number of large-scale datasets demands machine learning algorithms that can cope with very large matrices. Scalability to high dimensional problems is therefore a crucial issue in the design of machine learning algorithms that learn low-rank matrices.

Most of the recent algorithmic contributions on learning low-rank matrices have been proposed in the context of matrix completion. Convex relaxations based on the nuclear norm or trace norm heuristic (Fazel, 2002; Cai et al., 2008) have attracted a lot of attention as theoretical performance guarantees are available (Bach, 2008; Recht et al., 2010). However, an intrinsic limitation of the approach is that the rank of intermediate solutions cannot be bounded a priori. For large-scale problems, memory requirement may thus become prohibitively large.

A different yet complementary approach that resolves this issue, assumes a fixed-rank factorization of the solution and solves the corresponding non-convex optimization problem (Rennie and Srebro, 2005; Keshavan et al., 2010; Jain et al., 2010; Shalit et al., 2010). Despite the

potential introduction of local minima, fixed-rank factorizations achieve very good performance in practice. Moreover, Keshavan et al. (2010) and Jain et al. (2010) show that performance guarantees are also possible when a good heuristic is available for the initialization.

In this chapter, we pursue the research on fixed-rank factorizations and study the Riemannian geometry of two particular fixed-rank factorizations (Sections 5.2 and 5.3). We build on recent advances in optimization on Riemannian matrix manifolds (Absil et al., 2008) and exploit the manifold geometry of the search space. We design novel line-search algorithms for learning a linear regression model whose parameter is a fixed-rank matrix.

The current chapter makes the following contributions. We propose two novel quotient geometries for the set of fixed-rank non-symmetric matrices. The proposed geometries generalize previous works on symmetric fixed-rank positive semidefinite matrices (Bonnabel and Sepulchre, 2009; Journée et al., 2010). Using the developed quotient geometries, we propose novel efficient line-search algorithms for learning fixed-rank non-symmetric matrices. The proposed optimization framework confers a geometric basis to recent contributions on learning fixed-rank matrices and offers a versatile framework for the design of machine learning algorithms that learn fixed-rank matrices.

5.2 Quotient geometries of fixed-rank matrix factorizations

In this section, we review two matrix factorizations for fixed-rank non-symmetric matrices and study the associated quotient manifold geometries. The quotient nature of the underlying search space stems from the fact that an element $\mathbf{W} \in \mathcal{F}(r, d_1, d_2)$ is represented by an entire equivalence class of matrices due to intrinsic invariance properties of the factorization (Figure 5.1).

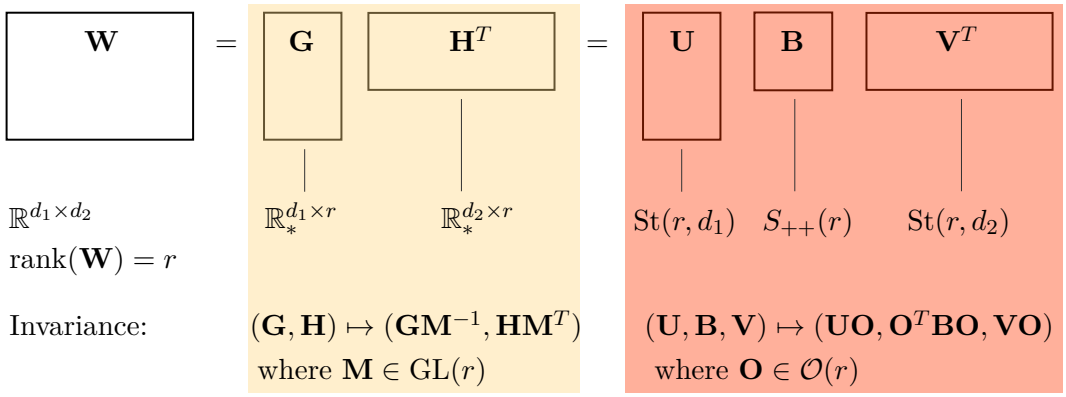


Figure 5.1: The considered fixed-rank factorizations for non-symmetric matrices admit a quotient manifold geometry due to intrinsic invariance properties of the factorization.

The factorizations of interest are rooted in the thin singular value decomposition (SVD)

$$\mathbf{W} = \mathbf{U}\mathbf{\Sigma}\mathbf{V}^T,$$

where $\mathbf{U} \in \mathrm{St}(r, d_1)$, $\mathbf{V} \in \mathrm{St}(r, d_2)$, and $\mathbf{\Sigma} \in \mathbb{R}^{r \times r}$ is diagonal with positive entries.

5.2.1 Balanced factorization

The SVD can be rearranged as

$$\mathbf{W} = (\mathbf{U}\mathbf{\Sigma}^{\frac{1}{2}})(\mathbf{\Sigma}^{\frac{1}{2}}\mathbf{V}^T) = \mathbf{G}\mathbf{H}^T, \tag{5.1}$$

where $\mathbf{G} = \mathbf{U}\mathbf{\Sigma}^{\frac{1}{2}} \in \mathbb{R}_*^{d_1 \times r}$ and $\mathbf{H} = \mathbf{V}\mathbf{\Sigma}^{\frac{1}{2}} \in \mathbb{R}_*^{d_2 \times r}$. The resulting fixed-rank factorization is not unique since the group action

$$(\mathbf{G}, \mathbf{H}) \mapsto (\mathbf{G}\mathbf{M}^{-1}, \mathbf{H}\mathbf{M}^T), \tag{5.2}$$

where $\mathbf{M} \in \mathrm{GL}(r) = \{\mathbf{M} \in \mathbb{R}^{r\times r} : \det(\mathbf{M}) \neq 0\}$, leaves the original matrix $\mathbf{W}$ unchanged.

The map (5.2) allows us to identify the search space of interest with the quotient space

$$\mathcal{F}(r, d_1, d_2) \simeq (\mathbb{R}_*^{d_1\times r} \times \mathbb{R}_*^{d_2\times r})/\mathrm{GL}(r), \tag{5.3}$$

which represents the set of equivalence classes

$$[\mathbf{W}] = [(\mathbf{G}, \mathbf{H})] = \{(\mathbf{G}\mathbf{M}^{-1}, \mathbf{H}\mathbf{M}^T) : \mathbf{M} \in \mathrm{GL}(r)\}. \tag{5.4}$$

Among the representatives $(\mathbf{G}, \mathbf{H})$ in (5.4), balanced factorizations are of particular interest. A factorization $\mathbf{W} = \mathbf{G}\mathbf{H}^T$ is balanced if $\mathbf{G}^T\mathbf{G} = \mathbf{H}^T\mathbf{H}$. Balanced factorizations are well-known in model reduction and system approximation (Helmke and Moore, 1996), they ensure good numerical conditioning and robustness to noise. Helmke and Moore (1996) show that balanced factorizations are the minimizers of the cost function

$$b(\mathbf{G}, \mathbf{H}) = \|\mathbf{G}\|_F^2 + \|\mathbf{H}\|_F^2, \tag{5.5}$$

within an equivalence class (5.4). Given a factorization $\mathbf{W} = \mathbf{G}\mathbf{H}^T$, a balanced factorization always exists but is not unique. The particular choice of representative (5.1) obtained from the SVD yields a balanced factorization with $\mathbf{G}^T\mathbf{G} = \mathbf{H}^T\mathbf{H} = \mathbf{\Sigma}$.

5.2.2 Polar factorization

A second factorization is obtained by considering the following group action on the SVD,

$$(\mathbf{U}, \mathbf{\Sigma}, \mathbf{V}) \mapsto (\mathbf{U}\mathbf{O}, \mathbf{O}^T\mathbf{\Sigma}\mathbf{O}, \mathbf{V}\mathbf{O}),$$

where $\mathbf{O} \in \mathcal{O}(r)$, the set of r-by-r rotation matrices. Since $\mathbf{O}^T\mathbf{\Sigma}\mathbf{O}$ now represents a positive definite matrix, this gives us the fixed-rank factorization

$$\mathbf{W} = \mathbf{U}\mathbf{B}\mathbf{V}^T,$$

where $\mathbf{U} \in \mathrm{St}(d_1, r)$, $\mathbf{B} \in S_{++}(r)$, $\mathbf{V} \in \mathrm{St}(d_2, r)$. The alternative choice $\mathbf{B}$ positive definite instead of $\mathbf{\Sigma}$ diagonal allows us to remove the discrete symmetries that are induced by the arbitrary order on the singular values. The search space of interest is again identified to a quotient manifold

$$\mathcal{F}(r, d_1, d_2) \simeq (\mathrm{St}(d_1, r) \times S_{++}(r) \times \mathrm{St}(d_2, r))/\mathcal{O}(r),$$

which represents the set of equivalence classes

$$[\mathbf{W}] = [(\mathbf{U}, \mathbf{B}, \mathbf{V})] = \{(\mathbf{U}\mathbf{O}, \mathbf{O}^T\mathbf{B}\mathbf{O}, \mathbf{V}\mathbf{O}) : \mathbf{O} \in \mathcal{O}(r)\}. \tag{5.6}$$

Since $\mathbf{U}$ and $\mathbf{V}$ are matrices with orthogonal columns, the polar factorization automatically encodes the property of a balanced factorization. Another nice property of the factorization is that $\|\mathbf{W}\|_F^2 = \|\mathbf{B}\|_F^2$. Consequently, a regularization on $\|\mathbf{W}\|_F^2$ is very cheap because it only involves a matrix of size r, with typically $r \ll d_1, d_2$.

5.3 Geometry of algorithms using a fixed-rank factorization

In this section, we develop Riemannian quotient manifold geometries for the balanced factorization $\mathbf{W} = \mathbf{G}\mathbf{H}^T$ (Section 5.3.1) and the polar factorization $\mathbf{W} = \mathbf{U}\mathbf{B}\mathbf{V}^T$ (Section 5.3.2). Each of these two quotient manifolds is equipped with a proper Riemannian metric and the associated expression of tangent vectors is computed. We also present efficient retractions for the two geometries. The corresponding line-search algorithms are presented in Section 5.4.

5.3.1 Geometry of algorithms using a balanced factorization

This section presents the first-order quotient geometry underlying the factorization $\mathbf{W} = \mathbf{G}\mathbf{H}^T$. As a first step, the total space $\mathbb{R}_*^{d_1\times r} \times \mathbb{R}_*^{d_2\times r}$ is endowed with the Riemannian metric,

$$\bar{g}_{(\mathbf{G},\mathbf{H})}((\bar{\xi}_\mathbf{G}, \bar{\xi}_\mathbf{H}), (\bar{\zeta}_\mathbf{G}, \bar{\zeta}_\mathbf{H})) = \mathrm{Tr}((\mathbf{G}^T\mathbf{G})^{-1}\bar{\xi}_\mathbf{G}^T\bar{\zeta}_\mathbf{G}) + \mathrm{Tr}((\mathbf{H}^T\mathbf{H})^{-1}\bar{\xi}_\mathbf{H}^T\bar{\zeta}_\mathbf{H}), \tag{5.7}$$

chosen to be invariant along the set of equivalence classes (5.4). We now compute the set of associated horizontal vectors and prove invariance of the metric (5.7) along the fibers (5.4).

Proposition 5.3.1. *The quotient manifold* $(\mathbb{R}_*^{d_1\times r} \times \mathbb{R}_*^{d_2\times r})/\mathrm{GL}(r)$ *endowed with the Riemannian metric* (5.7) *admits a set of horizontal vectors* $(\bar{\xi}_\mathbf{G}, \bar{\xi}_\mathbf{H}) \in \mathbb{R}^{d_1\times r} \times \mathbb{R}^{d_2\times r}$ *that satisfies*

$$\bar{\xi}_\mathbf{G}^T\mathbf{G}(\mathbf{H}^T\mathbf{H}) = (\mathbf{G}^T\mathbf{G})\mathbf{H}^T\bar{\xi}_\mathbf{H}. \tag{5.8}$$

An equivalent explicit expression for horizontal tangent vectors is given by

$$\bar{\xi}_\mathbf{G} = \mathbf{G}(\mathbf{G}^T\mathbf{G})^{-1}\mathbf{S}(\mathbf{G}^T\mathbf{G}) + \mathbf{G}_\perp, \quad \bar{\xi}_\mathbf{H} = \mathbf{H}(\mathbf{H}^T\mathbf{H})^{-1}\mathbf{S}^T(\mathbf{H}^T\mathbf{H}) + \mathbf{H}_\perp, \tag{5.9}$$

where $\mathbf{S} \in \mathbb{R}^{r\times r}$ *and matrices* $\mathbf{G}_\perp \in \mathbb{R}_*^{d_1\times r}$ *and* $\mathbf{H}_\perp \in \mathbb{R}_*^{d_2\times r}$ *satisfy* $\mathbf{G}_\perp^T\mathbf{G} = 0$ *and* $\mathbf{H}_\perp^T\mathbf{H} = 0$.

Proof. See Appendix A. □

The next proposition gives an explicit relation between horizontal vectors along a given fiber. This relation is exploited to show that the chosen metric (5.7) is invariant along the fibers (5.4).

Proposition 5.3.2. *Let* $(\mathbf{G}, \mathbf{H}) \in \mathbb{R}_*^{d_1\times r} \times \mathbb{R}_*^{d_2\times r}$*, and* $\xi_{[(\mathbf{G},\mathbf{H})]}$ *be a tangent vector to the quotient manifold at* $[(\mathbf{G}, \mathbf{H})]$*. The horizontal lifts of* $\xi_{[(\mathbf{G},\mathbf{H})]}$ *at point* $(\mathbf{G}, \mathbf{H})$ *and at point* $(\mathbf{G}\mathbf{M}^{-1}, \mathbf{H}\mathbf{M}^T)$ *are related as follows:*

$$(\bar{\xi}_{\mathbf{G}\mathbf{M}^{-1}}, \bar{\xi}_{\mathbf{H}\mathbf{M}^T}) = (\bar{\xi}_\mathbf{G}\mathbf{M}^{-1}, \bar{\xi}_\mathbf{H}\mathbf{M}^T), \quad \forall \mathbf{M} \in \mathrm{GL}(r).$$

Therefore, the chosen metric is invariant along the fibers,

$$\begin{aligned} g_{[(\mathbf{G},\mathbf{H})]}(\xi_{[(\mathbf{G},\mathbf{H})]}, \zeta_{[(\mathbf{G},\mathbf{H})]}) &\triangleq \bar{g}_{(\mathbf{G},\mathbf{H})}((\bar{\xi}_\mathbf{G}, \bar{\xi}_\mathbf{H}), (\bar{\zeta}_\mathbf{G}, \bar{\zeta}_\mathbf{H})) \\ &= \bar{g}_{(\mathbf{G}\mathbf{M}^{-1},\mathbf{H}\mathbf{M}^T)}((\bar{\xi}_{\mathbf{G}\mathbf{M}^{-1}}, \bar{\xi}_{\mathbf{H}\mathbf{M}^T}), (\bar{\zeta}_{\mathbf{G}\mathbf{M}^{-1}}, \bar{\zeta}_{\mathbf{H}\mathbf{M}^T})). \end{aligned}$$

Proof. See Appendix A. □

A simple and efficient retraction is provided by the formulas

$$\begin{aligned} R_\mathbf{G}(s\bar{\xi}_\mathbf{G}) &= \mathbf{G} + s\bar{\xi}_\mathbf{G}, \\ R_\mathbf{H}(s\bar{\xi}_\mathbf{H}) &= \mathbf{H} + s\bar{\xi}_\mathbf{H}, \end{aligned} \tag{5.10}$$

which is computed in only $O(d_1r + d_2r)$ operations.

5.3.2 Geometry of algorithms using a polar factorization

This section presents the first-order quotient geometry underlying the factorization $\mathbf{W} = \mathbf{U}\mathbf{B}\mathbf{V}^T$, which induces the quotient $\mathcal{F}(r, d_1, d_2) \simeq (\mathrm{St}(r, d_1) \times S_{++}(r) \times \mathrm{St}(r, d_2))/\mathcal{O}(r)$ (Section 5.2.2).

First, we endow the total space with the Riemannian metric

$$\bar{g}_{(\mathbf{U},\mathbf{B},\mathbf{V})}((\bar{\xi}_\mathbf{U}, \bar{\xi}_\mathbf{B}, \bar{\xi}_\mathbf{V}), (\bar{\zeta}_\mathbf{U}, \bar{\zeta}_\mathbf{B}, \bar{\zeta}_\mathbf{V})) = \mathrm{Tr}(\bar{\xi}_\mathbf{U}^T\bar{\zeta}_\mathbf{U}) + \mathrm{Tr}(\mathbf{B}^{-1}\bar{\xi}_\mathbf{B}\mathbf{B}^{-1}\bar{\zeta}_\mathbf{B}) + \mathrm{Tr}(\bar{\xi}_\mathbf{V}^T\bar{\zeta}_\mathbf{V}). \tag{5.11}$$

The set of horizontal vectors is given as follows.

Proposition 5.3.3. *The quotient manifold* $(\mathrm{St}(r,d_1)\times S_{++}(r)\times \mathrm{St}(r,d_2))/\mathcal{O}(r)$ *endowed with the Riemannian metric* (5.11) *admits a set of horizontal vectors* $(\bar{\xi}_{\mathbf{U}},\bar{\xi}_{\mathbf{B}},\bar{\xi}_{\mathbf{V}})$ *defined as*

$$\begin{aligned}
\bar{\xi}_{\mathbf{U}} &= \mathbf{U}\mathrm{Skew}(\mathbf{A})+\mathbf{U}_\perp, \quad && \mathbf{A}\in\mathbb{R}^{r\times r},\ \mathbf{U}_\perp\in\mathbb{R}_*^{d_1\times r} \text{ and } \mathbf{U}_\perp^T\mathbf{U}=0,\\
\bar{\xi}_{\mathbf{B}} &= \mathrm{Sym}(\mathbf{D}), && \mathbf{D}\in\mathbb{R}^{r\times r},\\
\bar{\xi}_{\mathbf{V}} &= \mathbf{V}\mathrm{Skew}(\mathbf{C})+\mathbf{V}_\perp, && \mathbf{C}\in\mathbb{R}^{r\times r},\ \mathbf{V}_\perp\in\mathbb{R}_*^{d_2\times r} \text{ and } \mathbf{V}_\perp^T\mathbf{V}=0,
\end{aligned}$$

with the additional requirement that

$$\mathbf{B}(\mathrm{Skew}(\mathbf{A})+\mathrm{Skew}(\mathbf{C}))\mathbf{B}=\bar{\xi}_{\mathbf{B}}\mathbf{B}-\mathbf{B}\bar{\xi}_{\mathbf{B}}.$$

Proof. See Appendix A. □

The next proposition shows that the chosen metric (5.11) is invariant along the fibers (5.6).

Proposition 5.3.4. *Let* $(\mathbf{U},\mathbf{B},\mathbf{V})\in\mathrm{St}(r,d_1)\times S_{++}(r)\times\mathrm{St}(r,d_2)$ *and* $\xi_{[(\mathbf{U},\mathbf{B},\mathbf{V})]}$ *be a tangent vector to the quotient manifold at* $[(\mathbf{U},\mathbf{B},\mathbf{V})]$*. The horizontal lifts of* $\xi_{[(\mathbf{U},\mathbf{B},\mathbf{V})]}$ *at point* $(\mathbf{U},\mathbf{B},\mathbf{V})$ *and at point* $(\mathbf{UO},\mathbf{O}^T\mathbf{BO},\mathbf{VO})$ *are related as follows,*

$$(\bar{\xi}_{\mathbf{UO}},\bar{\xi}_{\mathbf{O}^T\mathbf{BO}},\bar{\xi}_{\mathbf{VO}})=(\bar{\xi}_{\mathbf{U}}\mathbf{O},\mathbf{O}^T\bar{\xi}_{\mathbf{B}}\mathbf{O},\bar{\xi}_{\mathbf{V}}\mathbf{O}),\quad \forall\mathbf{O}\in\mathcal{O}(r).$$

Therefore, the chosen metric is invariant along the fibers, and, for all $\mathbf{O}\in\mathcal{O}(r)$*,*

$$\begin{aligned}
g_{[(\mathbf{U},\mathbf{B},\mathbf{V})]}(\xi_{[(\mathbf{U},\mathbf{B},\mathbf{V})]},\zeta_{[(\mathbf{U},\mathbf{B},\mathbf{V})]}) &\triangleq \bar{g}_{(\mathbf{U},\mathbf{B},\mathbf{V})}((\bar{\xi}_{\mathbf{U}},\bar{\xi}_{\mathbf{B}},\bar{\xi}_{\mathbf{V}}),(\bar{\zeta}_{\mathbf{U}},\bar{\zeta}_{\mathbf{B}},\bar{\zeta}_{\mathbf{V}}))\\
&= \bar{g}_{(\mathbf{UO},\mathbf{O}^T\mathbf{BO},\mathbf{VO})}((\bar{\xi}_{\mathbf{UO}},\bar{\xi}_{\mathbf{O}^T\mathbf{BO}},\bar{\xi}_{\mathbf{VO}}),(\bar{\zeta}_{\mathbf{UO}},\bar{\zeta}_{\mathbf{O}^T\mathbf{BO}},\bar{\zeta}_{\mathbf{VO}})).
\end{aligned}$$

Proof. See Appendix A □

We propose the following efficient retraction

$$\begin{aligned}
R_{\mathbf{U}}(s\bar{\xi}_{\mathbf{U}}) &= \mathrm{qf}(\mathbf{U}+s\bar{\xi}_{\mathbf{U}}),\\
R_{\mathbf{B}}(s\bar{\xi}_{\mathbf{B}}) &= \mathbf{B}^{\frac{1}{2}}\exp(s\mathbf{B}^{-\frac{1}{2}}\bar{\xi}_{\mathbf{B}}\mathbf{B}^{-\frac{1}{2}})\mathbf{B}^{\frac{1}{2}},\\
R_{\mathbf{V}}(s\bar{\xi}_{\mathbf{V}}) &= \mathrm{qf}(\mathbf{V}+s\bar{\xi}_{\mathbf{V}}),
\end{aligned}\tag{5.12}$$

where $\mathrm{qf}(\cdot)$ extracts the orthogonal factor of the QR factorization of its argument, $\exp(\cdot)$ stands for matrix exponential and $\mathbf{B}^{\frac{1}{2}}$ is the principal square root of $\mathbf{B}$.

5.4 Algorithms for regression on fixed-rank non-symmetric matrices

In this section, we exploit the geometry introduced in the previous section to develop line-search algorithms for the following linear regression problem. Given data matrix instances $\mathbf{X}\in\mathbb{R}^{d_2\times d_1}$, scalar observations $y\in\mathbb{R}$, and a linear regression model $\hat{y}=\mathrm{Tr}(\mathbf{WX})$, solve

$$\min_{\mathbf{W}\in\mathbb{R}^{d_1\times d_2}} \mathbb{E}_{\mathbf{X},y}\{\ell(\hat{y},y)\},\quad \text{subject to}\quad \mathrm{rank}(\mathbf{W})=r.\tag{5.13}$$

The loss function $\ell(\hat{y},y)$ penalizes the discrepancy between the observed value y and the value that is predicted by the model $\hat{y}$. Our focus will be on the quadratic loss $\ell(\hat{y},y)=\frac{1}{2}(\hat{y}-y)^2$.

We present the proposed regression algorithms for a general data $\mathbf{X}$ (Sections 5.4.1 and 5.4.2) as well as optimized versions of algorithms in the prevalent setting where the data $\mathbf{X}$ is rank-one.

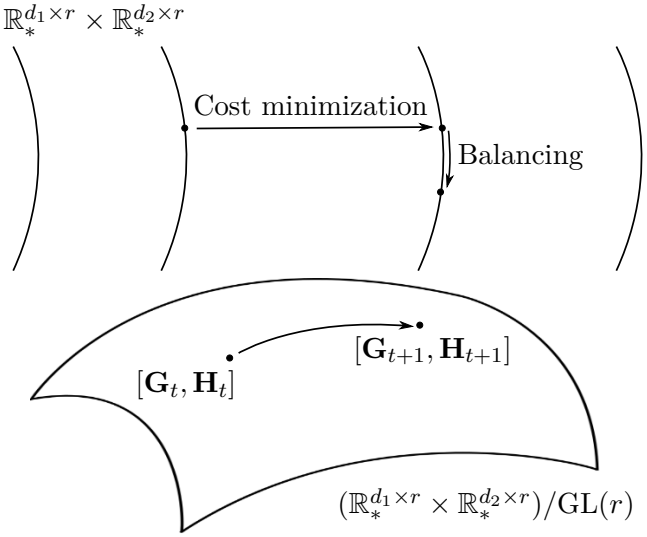


Figure 5.2: The proposed algorithm based on the balanced factorization proceeds in two cascaded steps: a cost minimization step, and a fiber balancing step that ensures good numerical conditioning. These two steps correspond to a single iteration on the quotient manifold.

5.4.1 Linear regression with a balanced factorization

In this section, we derive line-search algorithms to solve (5.13) using a balanced factorization. The resulting algorithms proceed in two cascaded updates (Figure 5.2). The first update moves the iterate from one equivalence class to another while minimizing the cost function of interest. The second "balancing" update is a change of representative along a given fiber minimizing (5.5). The balancing update ensures a good numerical conditioning of the algorithm. Eventually, the algorithm converges to a local minimum of the cost function that is a balanced factorization.

Algorithm

The factorization $\mathbf{W} = \mathbf{G}\mathbf{H}^T$ allows us to write the cost function of interest as

$$f(\mathbf{G}, \mathbf{H}) = \frac{1}{2}(\mathrm{Tr}(\mathbf{G}\mathbf{H}^T\mathbf{X}) - y)^2.$$

Applying formula (3.11) to this cost function yields the horizontal gradient

$$(\overline{\mathrm{grad}_{\mathbf{G}}\, f}, \overline{\mathrm{grad}_{\mathbf{H}}\, f}) = ((\hat{y} - y)\mathbf{X}^T\mathbf{H}(\mathbf{G}^T\mathbf{G}), (\hat{y} - y)\mathbf{X}\mathbf{G}(\mathbf{H}^T\mathbf{H})).$$

Combining this gradient with retraction (5.10) yields the online update

$$\begin{aligned} \widetilde{\mathbf{G}}_t &= \mathbf{G}_t - s_t(\hat{y}_t - y_t)\mathbf{X}_t^T\mathbf{H}_t(\mathbf{G}_t^T\mathbf{G}_t), \\ \widetilde{\mathbf{H}}_t &= \mathbf{H}_t - s_t(\hat{y}_t - y_t)\mathbf{X}_t\mathbf{G}_t(\mathbf{H}_t^T\mathbf{H}_t). \end{aligned} \tag{5.14}$$

The asymptotic computational complexity of this update is $O(d_1 d_2 r)$ for a general $\mathbf{X}_t$.

To balance a given factorization $\widetilde{\mathbf{W}}_t = \widetilde{\mathbf{G}}_t\widetilde{\mathbf{H}}_t^T$, Helmke and Moore (1996) propose the following gradient descent update,

$$\begin{aligned} \mathbf{G}_{t+1} &= \widetilde{\mathbf{G}}_t \exp(\alpha_t(\widetilde{\mathbf{H}}_t^T\widetilde{\mathbf{H}}_t - \widetilde{\mathbf{G}}_t^T\widetilde{\mathbf{G}}_t)), \\ \mathbf{H}_{t+1} &= \widetilde{\mathbf{H}}_t \exp(\alpha_t(\widetilde{\mathbf{G}}_t^T\widetilde{\mathbf{G}}_t - \widetilde{\mathbf{H}}_t^T\widetilde{\mathbf{H}}_t)), \end{aligned} \tag{5.15}$$

along with an adaptive step size

$$\alpha_t = 1/(2\lambda_{max}(\widetilde{\mathbf{G}}_t^T\widetilde{\mathbf{G}}_t + \widetilde{\mathbf{H}}_t^T\widetilde{\mathbf{H}}_t)), \tag{5.16}$$

where $\lambda_{max}(\cdot)$ denotes a function that extracts the largest eigenvalue of its argument.

To get insight into this fiber balancing update, observe that $\mathbf{G}_{t+1}\mathbf{H}_{t+1}^T = \widetilde{\mathbf{W}}_t$ for all α_t, and that the fixed points of the proposed iteration are balanced. A justification for the step size selection and a convergence proof for the balancing update is given by the following theorem.

Theorem 5.4.1. (Helmke and Moore, 1996, Theorem 6.1) *Algorithm* (5.15) *initialized with factorization* $\mathbf{W}_0 = \mathbf{G}_0\mathbf{H}_0^T$ *and endowed with adaptive step size* (5.16) *asymptotically converges to a balanced factorization* $\mathbf{W}_0 = \mathbf{G}_\infty\mathbf{H}_\infty^T$ *satisfying* $\mathbf{G}_\infty^T\mathbf{G}_\infty = \mathbf{H}_\infty^T\mathbf{H}_\infty$.

Proof. For the reader's convenience, the proof is reproduced in Appendix A. □

The computational complexity of the balancing update is $O((d_1 + d_2)r^2 + r^3)$.

The proposed cascaded algorithm asymptotically converges to a local minimum of the cost function with a balanced factorization. The insight comes from geometry: (5.14) is a gradient update on the quotient manifold $(\mathbb{R}_*^{d_1\times r} \times \mathbb{R}_*^{d_2\times r})/\mathrm{GL}(r)$, it is unaffected by the choice of the representative $(\mathbf{G}, \mathbf{H})$ in the fiber (5.4). In contrast, (5.15) is a gradient update in the fiber (5.4) for the cost function (5.5). In the quotient manifold, algorithm (5.14) is "blind" to the change of representative (5.15). The sequence of iterates thus converges to a fiber that minimizes the cost function. But algorithm (5.15) guarantees that the asymptotic factorization also minimizes the cost (5.5), implying the balancing condition $\mathbf{G}^T\mathbf{G} = \mathbf{H}^T\mathbf{H}$.

Optimizing the algorithm for rank-one data

Algorithm 1 presents optimized version of updates (5.14)-(5.15) in the prevalent case where the data $\mathbf{X}_t$ is rank-one, that is, $\mathbf{X}_t = \mathbf{x}_t\mathbf{z}_t^T$, with $\mathbf{x}_t \in \mathbb{R}^{d_2}$ and $\mathbf{z}_t \in \mathbb{R}^{d_1}$.

Algorithm 1 Balanced factorization

Input: $\{(\mathbf{x}_t, \mathbf{z}_t, y_t)\}_{t\geq 0}$ (dataset), $(\mathbf{G}_0, \mathbf{H}_0)$ (initial factorization), $T \geq 0$ (total number of iterations), $s_t > 0$ (sequence of step sizes), $\tau_B \geq 1$ (balancing period).
Output: the final factorization $(\mathbf{G}_T, \mathbf{H}_T)$.
Set $\tau = \tau_B$
Compute $\mathbf{S}_0 = \mathbf{G}_0^T\mathbf{G}_0$ and $\mathbf{R}_0 = \mathbf{H}_0^T\mathbf{H}_0$
for $t = 0$ **to** $T - 1$ **do**
  Pick a sample $(\mathbf{x}_t, \mathbf{z}_t, y_t)$
  Compute $\bar{\mathbf{x}}_t = \mathbf{H}_t^T\mathbf{x}_t$, $\bar{\mathbf{z}}_t = \mathbf{G}_t^T\mathbf{z}_t$, $\mathbf{s}_t = \mathbf{S}_t\bar{\mathbf{x}}_t$ and $\mathbf{r}_t = \mathbf{R}_t\bar{\mathbf{z}}_t$
  Predict $\hat{y}_t = \bar{\mathbf{x}}_t^T\bar{\mathbf{z}}_t$
  Set $\beta_t = s_t(\hat{y}_t - y_t)$
  Update $\widetilde{\mathbf{G}}_t = \mathbf{G}_t - \beta_t\mathbf{z}_t\mathbf{s}_t^T$
  Update $\widetilde{\mathbf{H}}_t = \mathbf{H}_t - \beta_t\mathbf{x}_t\mathbf{r}_t^T$
  Update $\widetilde{\mathbf{S}}_t = \mathbf{S}_t - \beta_t\bar{\mathbf{z}}_t\mathbf{s}_t^T - \beta_t\mathbf{s}_t\bar{\mathbf{z}}_t^T + \beta_t^2\|\bar{\mathbf{z}}_t\|_2^2\mathbf{s}_t\mathbf{s}_t^T$
  Update $\widetilde{\mathbf{R}}_t = \mathbf{R}_t - \beta_t\bar{\mathbf{x}}_t\mathbf{r}_t^T - \beta_t\mathbf{r}_t\bar{\mathbf{x}}_t^T + \beta_t^2\|\bar{\mathbf{x}}_t\|_2^2\mathbf{r}_t\mathbf{r}_t^T$
  Set $\tau = \tau - 1$
  if $\tau \leq 0$ **then**
    Perform a balancing step using (5.15)
    Set $\mathbf{S}_{t+1} = \mathbf{G}_{t+1}^T\mathbf{G}_{t+1}$ and $\mathbf{R}_{t+1} = \mathbf{H}_{t+1}^T\mathbf{H}_{t+1}$
    Set $\tau = \tau_B$
  else
    Set $\mathbf{G}_{t+1} = \widetilde{\mathbf{G}}_t$ and $\mathbf{H}_{t+1} = \widetilde{\mathbf{H}}_t$
    Set $\mathbf{S}_{t+1} = \widetilde{\mathbf{S}}_t$ and $\mathbf{R}_{t+1} = \widetilde{\mathbf{R}}_t$
  end if
end for

The complexity of update (5.14) reduces to $O((d_1 + d_2)r)$. As we observe in numerical experiments that it is not necessary to perform the balancing step (5.15) at each iteration, the

balancing step is only performed every τ_B iterations. This reduces the computational cost and is sufficient to ensure a good numerical conditioning of the algorithm.

Adding regularization

Although the rank constraint already enforces a spectral regularization effect on $\mathbf{W}$, it may be useful in practice to add a pointwise regularization term to the cost function,

$$f(\mathbf{G}, \mathbf{H}) = \frac{1}{2}(\mathrm{Tr}(\mathbf{G}\mathbf{H}^T\mathbf{X}) - y)^2 + \frac{\lambda}{2}\|\mathbf{G}\mathbf{H}^T\|_F^2.$$

A regularizer $\|\mathbf{W}\|_F^2 = \|\mathbf{G}\mathbf{H}^T\|_F^2$ is chosen because it is invariant along the set of equivalence classes (5.4), as opposed to the common choice $\|\mathbf{G}\|_F^2 + \|\mathbf{H}\|_F^2$ (Rennie and Srebro, 2005).

With such a regularization, update (5.14) becomes

$$\begin{aligned}\widetilde{\mathbf{G}}_t &= \mathbf{G}_t - s_t(\hat{y}_t - y_t)\mathbf{X}_t^T\mathbf{H}_t(\mathbf{G}_t^T\mathbf{G}_t) - \lambda\mathbf{G}_t(\mathbf{H}_t^T\mathbf{H}_t)(\mathbf{G}_t^T\mathbf{G}_t),\\ \widetilde{\mathbf{H}}_t &= \mathbf{H}_t - s_t(\hat{y}_t - y_t)\mathbf{X}_t\mathbf{G}_t(\mathbf{H}_t^T\mathbf{H}_t) - \lambda\mathbf{H}_t(\mathbf{G}_t^T\mathbf{G}_t)(\mathbf{H}_t^T\mathbf{H}_t).\end{aligned}$$

This modification does not significantly increase the computational cost, since the regularization terms have common subexpressions with the gradient of the loss. The computational complexity overhead is $O((d_1 + d_2)r^2)$. We will show in Section 5.4.2 that a more efficient approach to add regularization can be obtained using the factorization $\mathbf{W} = \mathbf{U}\mathbf{B}\mathbf{V}^T$.

Connection with existing algorithms

The proposed algorithm based on the balanced factorization is closely related to the gradient descent version of the Maximum Margin Matrix Factorization (MMMF) algorithm (Rennie and Srebro, 2005). The stochastic gradient descent version of MMMF writes as

$$\begin{aligned}\mathbf{G}_{t+1} &= \mathbf{G}_t - s_t(\hat{y}_t - y_t)\mathbf{X}_t^T\mathbf{H}_t,\\ \mathbf{H}_{t+1} &= \mathbf{H}_t - s_t(\hat{y}_t - y_t)\mathbf{X}_t\mathbf{G}_t.\end{aligned} \tag{5.17}$$

In contrast to (5.17), the proposed update (5.14) is invariant along the set of equivalence classes (5.4). This resolves the issue of choosing an appropriate step size when there is a discrepancy between $\|\mathbf{G}\|_F$ and $\|\mathbf{H}\|_F$. Indeed, this situation leads to a slow convergence of the MMMF algorithm, whereas it does not affect the proposed algorithm (Figure 5.3).

To illustrate this effect, the two algorithms are compared in batch mode with random data generated according to

$$y_i = \mathbf{z}_i^T\mathbf{W}^\star\mathbf{x}_i, \quad i = 1, ..., n,$$

where $n = 10^4$, $\mathbf{W}^\star \in \mathcal{F}(1000, 500, 10)$, $\mathbf{z}_i \in \mathbb{R}^{1000}$ and $\mathbf{x}_i \in \mathbb{R}^{500}$ have entries drawn from a standard Gaussian distribution $\mathcal{N}(0, 1)$. The step size is computed using the Armijo rule (Nocedal and Wright, 2006). The initial discrepancy between the factors is $\|\mathbf{G}_0\|_F \approx 5\|\mathbf{H}_0\|_F$.

The LORETA algorithm (Shalit et al., 2010) directly fits in the considered optimization framework. LORETA relies on an embedded manifold geometry of the set of fixed-rank matrices, where $\mathcal{F}(r, d_1, d_2)$ is regarded as a submanifold embedded in $\mathbb{R}^{d_1 \times d_2}$. Indeed, the tangent space at a given point $\mathbf{W} = \mathbf{G}\mathbf{H}^T$ is given by

$$T_{\mathbf{G}\mathbf{H}^T}\mathcal{F}(r, d_1, d_2) = \{\mathbf{G}\mathbf{\Psi}^T + \mathbf{\Delta}\mathbf{H}^T : \mathbf{\Delta} \in \mathbb{R}^{d_1 \times r} \text{ and } \mathbf{\Psi} \in \mathbb{R}^{d_2 \times r}\}. \tag{5.18}$$

For example, it is obtained by differentiation of a curve $\gamma(t) = \mathbf{G}(t)\mathbf{H}(t) \in \mathcal{F}(r, d_1, d_2)$ passing through the point $\gamma(0) = \mathbf{G}\mathbf{H}^T$. The chain rule yields $\dot{\gamma}(0) = \mathbf{G}(0)\dot{\mathbf{H}}(0)^T + \dot{\mathbf{G}}(0)\mathbf{H}(0)^T$, which

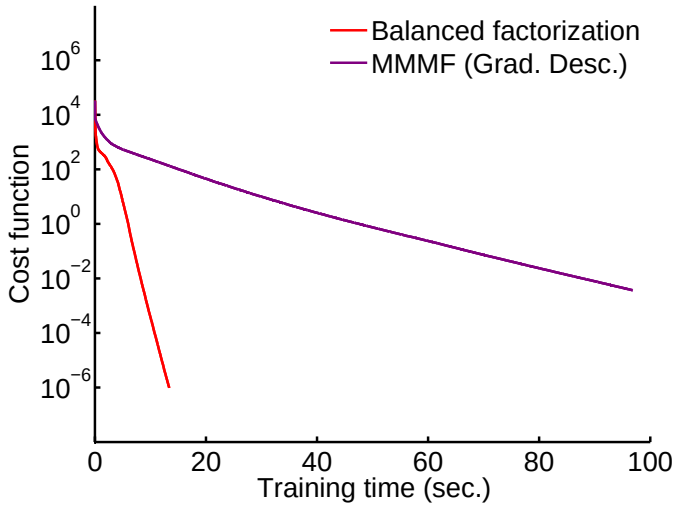


Figure 5.3: The proposed algorithm is not affected by a discrepancy between $\|\mathbf{G}\|_F$ and $\|\mathbf{H}\|_F$, a situation that leads to a slow convergence of the MMMF algorithm.

is directly identified to (5.18). Representation (5.18) is an over-parametrization of the tangent space $T_{\mathbf{G}\mathbf{H}^T}\mathcal{F}(r, d_1, d_2)$ at point $\mathbf{W} = \mathbf{G}\mathbf{H}^T$. A minimal representation is provided by

$$T_{\mathbf{G}\mathbf{H}^T}\mathcal{F}(r, d_1, d_2) = \{\underbrace{\mathbf{G}\mathbf{M}\mathbf{H}^T}_{\xi^S} + \underbrace{\mathbf{G}\mathbf{L}\mathbf{H}_\perp^T}_{\xi_l^P} + \underbrace{\mathbf{G}_\perp\mathbf{R}\mathbf{H}^T}_{\xi_r^P} : \mathbf{M} \in \mathbb{R}^{r\times r}, \mathbf{L} \in \mathbb{R}^{r\times(d_2-r)}, \mathbf{R} \in \mathbb{R}^{(d_1-r)\times r}\},$$

where $\mathbf{G}_\perp \in \mathbb{R}_*^{d_1\times(d_1-r)}$ and $\mathbf{H}_\perp \in \mathbb{R}_*^{d_2\times(d_2-r)}$ satisfy $\mathbf{G}_\perp^T\mathbf{G} = 0$ and $\mathbf{H}_\perp^T\mathbf{H} = 0$.

Given a search direction $\xi \in \mathbb{R}^{d_1\times d_2}$, computing its components ξ^S, ξ_r^P and ξ_l^P is achieved by considering projectors $\mathbf{G}\mathbf{G}^\dagger$ and $\mathbf{H}\mathbf{H}^\dagger$, where $\mathbf{A}^\dagger = (\mathbf{A}^T\mathbf{A})^{-1}\mathbf{A}^T$ is the pseudo-inverse of $\mathbf{A}$. One has $\xi^S = (\mathbf{G}\mathbf{G}^\dagger)\xi(\mathbf{H}\mathbf{H}^\dagger)$, $\xi_r^P = (\mathbf{I} - \mathbf{G}\mathbf{G}^\dagger)\xi(\mathbf{H}\mathbf{H}^\dagger)$ and $\xi_l^P = (\mathbf{G}\mathbf{G}^\dagger)\xi(\mathbf{I} - \mathbf{H}\mathbf{H}^\dagger)$, so that a given tangent vector writes as $\xi_\mathbf{W} = \xi^S + \xi_r^P + \xi_l^P$.

Shalit et al. (2010) propose the following retraction formula

$$R_\mathbf{W}(s\xi_\mathbf{W}) = \mathbf{S}_1(\mathbf{G}\mathbf{H}^T)^\dagger\mathbf{S}_2,$$

where

$$\begin{aligned}
\mathbf{S}_1 &= \mathbf{G}\mathbf{H}^T + \frac{1}{2}\xi^S + \xi_r^P - \frac{1}{8}\xi^S(\mathbf{G}\mathbf{H}^T)^\dagger\xi^S - \frac{1}{2}\xi_l^P(\mathbf{G}\mathbf{H}^T)^\dagger\xi^S,\\
\mathbf{S}_2 &= \mathbf{G}\mathbf{H}^T + \frac{1}{2}\xi^S + \xi_l^P - \frac{1}{8}\xi^S(\mathbf{G}\mathbf{H}^T)^\dagger\xi^S - \frac{1}{2}\xi^S(\mathbf{G}\mathbf{H}^T)^\dagger\xi_r^P.
\end{aligned}$$

Applying the formulas above to the Euclidean gradient descent search direction

$$\xi = -\nabla_\mathbf{W} f(\mathbf{W}) = -(\hat{y}_t - y_t)\mathbf{X}_t^T,$$

yields a version of LORETA for the considered linear regression problem.

In the case of rank-one data $\mathbf{X}_t = \mathbf{x}_t\mathbf{z}_t^T$, Shalit et al. (2010) show that LORETA can be optimized to achieve a computational complexity $O((d_1 + d_2)r)$. This is the same complexity as update (5.14). However, LORETA has a significantly larger constant factor as the involved optimizations for rank-one data rely on rank-one updates of pseudo-inverse matrices.

A regularization term proportional to $\|\mathbf{W}\|_F^2$ can be taken into account with an additional number of operations proportional to $O((d_1 + d_2)r^2)$. The regularization term however prevents the algorithm to rely on rank-one updates of pseudo-inverses. In that case pseudo-inverses must be computed at each iteration and significantly increase the computational cost of the algorithm.

5.4.2 Linear regression with cheap regularization

We now present the regression algorithm based on the factorization $\mathbf{W} = \mathbf{U}\mathbf{B}\mathbf{V}^T$.

Algorithms

With the factorization $\mathbf{W} = \mathbf{U}\mathbf{B}\mathbf{V}^T$, the cost function is given by

$$f(\mathbf{U}, \mathbf{B}, \mathbf{V}) = \frac{1}{2}(\mathrm{Tr}(\mathbf{U}\mathbf{B}\mathbf{V}^T\mathbf{X}) - y)^2. \tag{5.19}$$

Applying formula (3.11) and removing the component that lies in the normal space yields

$$\begin{aligned}
\overline{\mathrm{grad}_{\mathbf{U}} f} &= (\hat{y} - y)(\mathbf{X}^T\mathbf{V}\mathbf{B} - \mathbf{U}\mathrm{Sym}(\mathbf{U}^T\mathbf{X}^T\mathbf{V}\mathbf{B})), \\
\overline{\mathrm{grad}_{\mathbf{B}} f} &= (\hat{y} - y)\mathbf{B}\mathrm{Sym}(\mathbf{V}^T\mathbf{X}\mathbf{U})\mathbf{B}, \\
\overline{\mathrm{grad}_{\mathbf{V}} f} &= (\hat{y} - y)(\mathbf{X}\mathbf{U}\mathbf{B} - \mathbf{V}\mathrm{Sym}(\mathbf{V}^T\mathbf{X}\mathbf{U}\mathbf{B})).
\end{aligned}$$

Combining this horizontal gradient with the retraction (5.12) gives us the online algorithm

$$\begin{aligned}
\mathbf{U}_{t+1} &= \mathrm{qf}(\mathbf{U}_t - s_t(\hat{y}_t - y_t)(\mathbf{X}_t^T\mathbf{V}_t\mathbf{B}_t - \mathbf{U}_t\mathrm{Sym}(\mathbf{U}_t^T\mathbf{X}_t^T\mathbf{V}_t\mathbf{B}_t))), \\
\mathbf{B}_{t+1} &= \mathbf{B}_t^{\frac{1}{2}} \exp(-s_t(\hat{y}_t - y_t)\mathbf{B}_t^{\frac{1}{2}}\mathrm{Sym}(\mathbf{V}_t^T\mathbf{X}_t\mathbf{U}_t)\mathbf{B}_t^{\frac{1}{2}})\mathbf{B}^{\frac{1}{2}}, \\
\mathbf{V}_{t+1} &= \mathrm{qf}(\mathbf{V}_t - s_t(\hat{y}_t - y_t)(\mathbf{X}_t\mathbf{U}_t\mathbf{B}_t - \mathbf{V}_t\mathrm{Sym}(\mathbf{V}_t^T\mathbf{X}_t^T\mathbf{U}_t\mathbf{B}_t))).
\end{aligned} \tag{5.20}$$

Optimizing the algorithm for rank-one data

In the case of rank-one data, the $\mathrm{qf}(\cdot)$ function can be implemented using rank-one updates of the QR factorization (Daniel et al., 1976). This allows us to reduces the cost of a QR factorization to $O(dr)$, compared to $O(dr^2)$ when it is computed from scratch.

Algorithm 2 Polar factorization

Input: $\{(\mathbf{x}_t, \mathbf{z}_t, y_t)\}_{t\geq 0}$ (dataset), $(\mathbf{U}_0, \mathbf{B}_0, \mathbf{V}_0)$ (initial factorization), $T \geq 0$ (total number of iterations), $s_t > 0$ (sequence of step sizes).
Output: the final factorization $(\mathbf{U}_T, \mathbf{B}_T, \mathbf{V}_T)$
for $t = 0$ **to** $T - 1$ **do**
  Pick a sample $(\mathbf{x}_t, \mathbf{z}_t, y_t)$
  Set $\bar{\mathbf{x}}_t = \mathbf{V}_t^T\mathbf{x}_t$ and $\bar{\mathbf{z}}_t = \mathbf{U}_t^T\mathbf{z}_t$
  Compute $\mathbf{B}_t^{\frac{1}{2}}$, the matrix square root of $\mathbf{B}_t$
  Set $\mathbf{s}_t = \mathbf{B}_t^{\frac{1}{2}}\bar{\mathbf{x}}_t$, $\mathbf{r}_t = \mathbf{B}_t^{\frac{1}{2}}\bar{\mathbf{z}}_t$, $\bar{\mathbf{s}}_t = \mathbf{B}_t^{\frac{1}{2}}\mathbf{s}_t$ and $\bar{\mathbf{r}}_t = \mathbf{B}_t^{\frac{1}{2}}\mathbf{r}_t$
  Predict $\hat{y}_t = \mathbf{r}_t^T\mathbf{s}_t$
  Set $\beta_t = s_t(\hat{y}_t - y_t)$
  Set $\mathbf{U}_{t+1} = \mathrm{qf}(\mathbf{U}_t - \beta_t(\mathbf{z}_t\bar{\mathbf{s}}_t^T - \mathbf{U}_t\mathrm{Sym}(\bar{\mathbf{z}}_t\bar{\mathbf{s}}_t^T)))$
  Set $\mathbf{B}_{t+1} = \mathbf{B}_t^{\frac{1}{2}} \exp(-\beta_t\mathrm{Sym}(\mathbf{s}_t\mathbf{r}_t^T))\mathbf{B}_t^{\frac{1}{2}}$
  Set $\mathbf{V}_{t+1} = \mathrm{qf}(\mathbf{V}_t - \beta_t(\mathbf{x}_t\bar{\mathbf{r}}_t^T - \mathbf{V}_t\mathrm{Sym}(\bar{\mathbf{x}}_t\bar{\mathbf{r}}_t^T)))$
end for

Overall, each update of Algorithm 2 now costs $O((d_1 + d_2)r + r^3)$. For large-scale problems, the value of the rank r is typically much smaller than the dimensions of the problem d_1 and d_2. In that setting, r^3 is therefore negligible compared to $(d_1 + d_2)r$.

Adding regularization

With an additional regularization term proportional to $\|\mathbf{W}\|_F^2$, the cost (5.19) becomes

$$f(\mathbf{U},\mathbf{B},\mathbf{V}) = \frac{1}{2}(\mathrm{Tr}(\mathbf{U}\mathbf{B}\mathbf{V}^T\mathbf{X}) - y)^2 + \frac{\lambda}{2}\|\mathbf{B}\|_F^2,$$

and only the update of $\mathbf{B}$ needs the cheap modification

$$\mathbf{B}_{t+1} = \mathbf{B}_t^{\frac{1}{2}} \exp(-s_t(((\hat{y}_t - y_t)\mathbf{B}_t^{\frac{1}{2}}\mathrm{Sym}(\mathbf{V}_t^T\mathbf{X}_t\mathbf{U}_t)\mathbf{B}_t^{\frac{1}{2}}) + \mathbf{B}_t^2))\mathbf{B}^{\frac{1}{2}}.$$

Connection with existing algorithms

The OptSpace algorithm (Keshavan et al., 2010) also relies on the factorization $\mathbf{W} = \mathbf{U}\mathbf{B}\mathbf{V}^T$, but with $\mathbf{B} \in \mathbb{R}^{r\times r}$ not necessarily symmetric positive definite. This algorithm can be interpreted in the considered geometric optimization as a gradient descent algorithm for the cost function

$$\tilde{f}(\mathbf{U},\mathbf{V}) = \min_{\mathbf{B}\in\mathbb{R}^{r\times r}} f(\mathbf{U},\mathbf{B},\mathbf{V}),$$

over the product of Grassmann manifolds $\mathrm{St}(r,d_1)/\mathcal{O}(r) \times \mathrm{St}(r,d_2)/\mathcal{O}(r)$.

The algorithm alternates between a gradient descent step on the subspaces $\mathbf{U}$ and $\mathbf{V}$ for fixed $\mathbf{B}$, and a least-square estimation of $\mathbf{B}$ for fixed $\mathbf{U}$ and $\mathbf{V}$. The proposed algorithm is different from OptSpace in the choice $\mathbf{B}$ positive definite versus $\mathbf{B} \in \mathbb{R}^{r\times r}$. As a consequence, each step of our algorithm retains the geometry of a SVD factorization. Our algorithm also differs from OptSpace in the simultaneous and progressive nature of the updates. Furthermore, the choice $\mathbf{B} \succ 0$ allows us to derive alternative updates based on different metrics on the set $S_{++}(r)$. This flexibility is exploited in Chapter 4 to show that metrics of $S_{++}(r)$ are connected to Bregman divergences and information geometry. One iteration of OptSpace writes as

$$\begin{aligned}
\mathbf{U}_{t+1} &= \mathrm{qf}(\mathbf{U}_t - s_t\sum_{i=1}^n(\hat{y}_i - y_i)(\mathbf{I} - \mathbf{U}_t\mathbf{U}_t^T)\mathbf{X}_i^T\mathbf{V}_t\mathbf{B}_t),\\
\mathbf{V}_{t+1} &= \mathrm{qf}(\mathbf{V}_t - s_t\sum_{i=1}^n(\hat{y}_i - y_i)(\mathbf{I} - \mathbf{V}_t\mathbf{V}_t^T)\mathbf{X}_i\mathbf{U}_t\mathbf{B}_t),\\
\mathbf{B}_{t+1} &= \min_{\mathbf{B}\in\mathbb{R}^{r\times r}} \frac{1}{2}\sum_{i=1}^n(\mathrm{Tr}(\mathbf{U}_{t+1}\mathbf{B}\mathbf{V}_{t+1}^T\mathbf{X}_i) - y_i)^2.
\end{aligned} \tag{5.21}$$

Recently, the authors of OptSpace have extended the algorithm with a regularization term proportional to $\|\mathbf{B}\|_F^2$. Details can be found in the paper of Keshavan and Montanari (2010).

The singular value projection (SVP) algorithm (Jain et al., 2010) is based on the SVD factorization $\mathbf{W} = \mathbf{U}\mathbf{B}\mathbf{V}^T$ with $\mathbf{B} \in \mathbb{R}^{r\times r}$ and diagonal. An iteration of the algorithm writes as

$$\mathbf{U}_{t+1}\mathbf{B}_{t+1}\mathbf{V}_{t+1}^T = \mathrm{SVD}_r(\mathbf{U}_t\mathbf{B}_t\mathbf{V}_t^T - s_t\sum_{i=1}^n(\hat{y}_i - y_i)\mathbf{X}_i^T),$$

where $\mathrm{SVD}_r(\cdot)$ extracts the top-r singular values and singular vectors of its argument.

This algorithm is a projected gradient algorithm in the embedding space $\mathbb{R}^{d_1\times d_2}$ and relies on an efficient SVD-based retraction exploiting the sparse structure of the gradient.

When the gradient has not a sparse structure, the approach cannot be applied to large-scale problems because each iteration of the algorithm then relies on the SVD of a dense matrix.

The embedded geometry of $\mathcal{F}(r,d_1,d_2)$ can be exploited to generalize the approach when the gradient is not sparse. Indeed, a minimal parametrization of the tangent space (5.18) in the SVD coordinates $\mathbf{W} = \mathbf{U}\mathbf{B}\mathbf{V}^T$, yields a generic tangent vector

$$\xi_{\mathbf{W}} = (\mathbf{U}\mathbf{U}^T)\xi + \xi(\mathbf{V}\mathbf{V}^T) - (\mathbf{U}\mathbf{U}^T)\xi(\mathbf{V}\mathbf{V}^T), \quad \xi \in \mathbb{R}^{d_1\times d_2}, \tag{5.22}$$

which is at most of rank $2r$. Therefore, if the Euclidean gradient is projected onto the set of tangent vectors (5.22), a more efficient retraction can be used (see Vandereycken, 2011).

Simonsson and Eldén (2010) considered the variant factorization $\mathbf{W} = \mathbf{U}\mathbf{Z}^T$, for which $\mathbf{U} \in \mathrm{St}(r, d_1)$ and $\mathbf{Z} \in \mathbb{R}_*^{d_2 \times r}$. Although they propose a Newton's algorithm, the corresponding gradient descent version directly fits into the considered optimization framework.

The same factorization is also exploited in GROUSE (Balzano et al., 2010). This online algorithm estimates the subspace $\mathbf{U}$ with a gradient descent on the Grassmann manifold $\mathrm{St}(r, d_1)/\mathcal{O}(r)$ and computes the remaining factor $\mathbf{Z}$ using least-squares.

5.4.3 Kernelized algorithms for rank-one data

Using a result from Abernethy et al. (2009, Theorem 3), the regression model based on rank-one data $\mathbf{X} = \mathbf{x}\mathbf{z}^T$ generalizes to arbitrary non-linear transformations $\phi(\mathbf{x})$ and $\varphi(\mathbf{z})$, with $\phi : \mathbb{R}^{d_1} \to \mathcal{X}$ and $\varphi : \mathbb{R}^{d_2} \to \mathcal{Z}$. The regression model is now a bilinear form

$$\hat{y} = \langle \phi(\mathbf{x}), W\varphi(\mathbf{z}) \rangle_{\mathcal{X}} ,$$

and $W \in \mathcal{B}_r(\mathcal{Z}, \mathcal{X})$ is a rank-r bounded compact operator from $\mathcal{Z}$ to $\mathcal{X}$.

A finite dimensional representation of the operator minimizing the empirical risk can be constructed from the n-by-n Gram matrices $\mathbf{K}_{ij} = \langle \phi(\mathbf{x}_i), \phi(\mathbf{x}_j) \rangle_{\mathcal{X}}$, and $\mathbf{L}_{ij} = \langle \varphi(\mathbf{z}_i), \varphi(\mathbf{z}_j) \rangle_{\mathcal{Z}}$. The finite dimensional representation is obtained by solving the regression problem

$$\min_{\mathbf{W} \in \mathbb{R}^{\tilde{d}_1 \times \tilde{d}_2}} \frac{1}{n} \sum_{i=1}^{n} (\tilde{\mathbf{z}}_i^T \mathbf{W} \tilde{\mathbf{x}}_i - y_i)^2 + \|\mathbf{W}\|_F^2,$$

where $\mathbf{K} = \widetilde{\mathbf{X}}\widetilde{\mathbf{X}}^T$ and $\mathbf{L} = \widetilde{\mathbf{Z}}\widetilde{\mathbf{Z}}^T$, and where $\widetilde{\mathbf{X}} \in \mathbb{R}^{n \times \tilde{d}_1}$ and $\widetilde{\mathbf{Z}} \in \mathbb{R}^{n \times \tilde{d}_2}$ are any square root decomposition of $\mathbf{K}$ and $\mathbf{L}$. For instance, a valid decomposition can be obtained using kernel PCA or incomplete Cholesky decomposition (Fine et al., 2001; Bach and Jordan, 2005).

5.5 Experiments

We illustrate the good behavior of the proposed algorithms on two benchmarks.

5.5.1 Learning on pairs

We consider the general setup of learning on data pairs (see Section 2.5.5).

Toy data Random data are generated according to

$$y_i = \mathbf{z}_i^T \mathbf{W}^\star \mathbf{x}_i + \epsilon_i, \quad i = 1, ..., n, \tag{5.23}$$

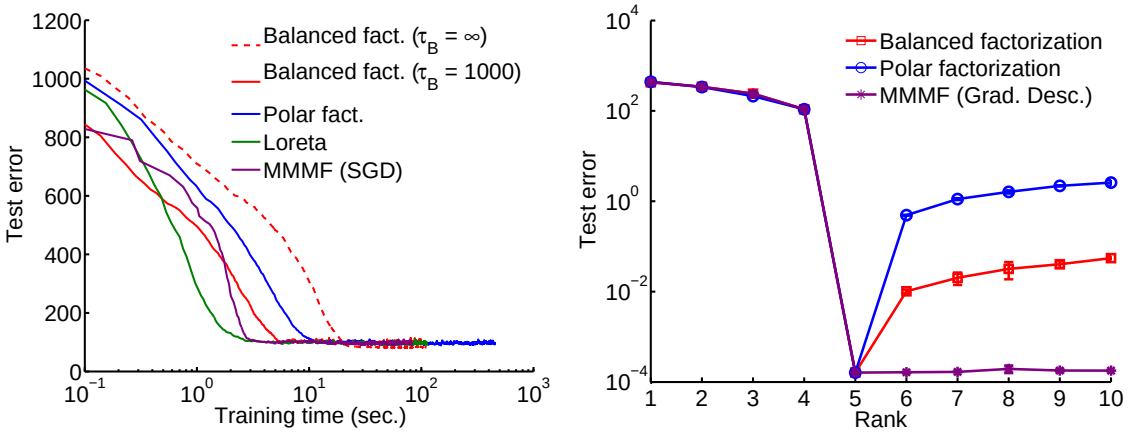
where $\mathbf{W}^\star \in \mathcal{F}(50, 25, 5)$, $\mathbf{z}_i \in \mathbb{R}^{50}$ and $\mathbf{x}_i \in \mathbb{R}^{25}$ have entries drawn from a standard Gaussian distribution $\mathcal{N}(0, 1)$. Gaussian noise $\epsilon_i \sim \mathcal{N}(0, 10^{-2})$ is added to the data. In batch mode, we minimize the cost function

$$f(\mathbf{W}) = \sum_{i=1}^{n} (\mathbf{z}_i^T \mathbf{W}^\star \mathbf{x}_i - y_i)^2.$$

In the online regime, we randomly pick data $(\mathbf{x}_t, \mathbf{z}_t, y_t)$ from the learning set of samples and minimize the instantaneous cost function

$$f_t(\mathbf{W}) = (\mathbf{z}_t^T \mathbf{W}^\star \mathbf{x}_t - y_t)^2.$$

We first show that the proposed algorithms perform well in the online setting (Figure 5.4(a)). The dataset is generated from (5.23), 40000 samples are used for learning and 10000 for testing.



(a) (b)

Figure 5.4: Learning on pairs (toy data): both online (a) and batch (b) algorithms perform well compared to the existing approaches.

At each iteration, the algorithms pick a sample at random and update the model. The algorithms all process the same set of samples. The step size is selected during a pre-training phase of 5000 iterations, the step size leading to the smallest train error is retained.

Figure 5.4(a) reports the test error as a function of the training time. The proposed algorithms compete with Loreta Shalit et al. (2010) and with an online version of MMMF. Balancing reduces the time to achieve convergence.

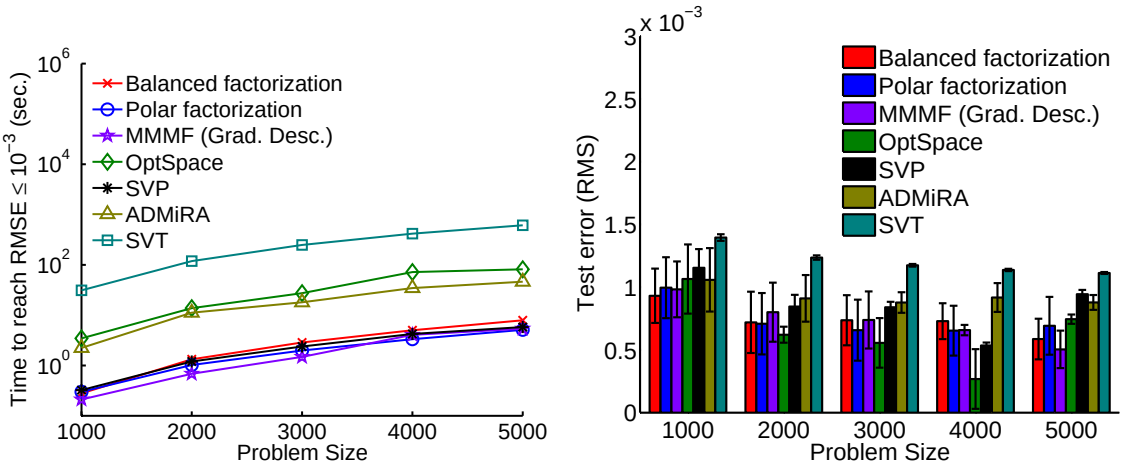
We now test the proposed algorithms in batch mode. Using (5.23), we generate a dataset of 3000 samples and compute the test error as a function of the approximation rank (Figure 5.4(b)). The validation protocol is 90/10 train/test split. The results are averaged over 10 random partitions. The regularization parameter λ is selected using cross-validation. Not surprisingly, the competing algorithms all achieve a minimal error when the rank equals the rank of the target model. When the rank further increases, the proposed algorithms start overfitting. These observations suggest to increase progressively the value of the rank until performance degrades.

5.5.2 Low-rank matrix completion

We consider the problem of low-rank matrix completion presented in Section 2.1.2. The considered cost function is (2.5). The proposed algorithms are run in batch mode and are compared to: a gradient descent version of MMMF (Rennie and Srebro, 2005), OptSpace (Keshavan et al., 2010), SVP (Jain et al., 2010), ADMiRA (Lee and Bresler, 2009), a matching pursuit based algorithm, and SVT (Cai et al., 2007), a nuclear norm minimization based algorithm. We use the Matlab code provided by the respective authors except for MMMF for which we use our own implementation.

Synthetic data with uniform sampling Following (Jain et al., 2010), we generate random rank-2 matrices $\mathbf{W}^\star \in \mathbb{R}^{d\times d}$ of various sizes d and sample a fraction $p = 0.1$ of entries for learning. Figure 5.5(a) reports the time taken by the algorithms to reach a root mean square error (RMSE) of 10^{-3} on the learning set. The corresponding RMSE on the test set is presented in Figure 5.5(b). Results are averaged over 10 runs. The proposed algorithms compete with the other methods both in terms of convergence speed and test error.

Movielens data Finally, we compare the fixed-rank factorization based algorithms on the 1M movielens collaborative filtering data, which contains one million ratings for 6,040 users and 3,952 movies. We average the test RMSE for different values of the rank over 10 random 90/10



(a) (b)

Figure 5.5: Matrix completion on synthetic data. The proposed algorithms compete with state-of-the-art low-rank matrix completion algorithms, both in terms of time to reach convergence (c) and test error (d).

train/test partitions. Results are presented in Table 5.1. The proposed algorithms compete with the other methods. In particular, the algorithm based on the polar factorization achieves the smallest RMSE for rank 10 and 12. Standard deviations of the errors are not reported since they are not significant.

Table 5.1: Test RMSE on Movielens data

r	Balanced	Polar	MMMF	SVP	OptSpace	ADMiRa
2	0.90	0.89	0.88	0.89	0.90	1.11
3	0.90	0.89	0.88	0.88	0.90	1.09
5	0.90	0.87	0.86	0.88	0.90	1.07
7	0.88	0.86	0.86	0.89	0.89	1.04
10	0.88	**0.85**	0.86	0.90	0.89	1.04
12	0.88	**0.85**	0.87	0.92	0.89	1.03

5.6 Conclusion

In this chapter, we present novel algorithms for learning fixed-rank matrices in high-dimensional regression problems. The developed geometrical framework unifies several recent contributions in the literature and generalizes the results of Chapter 4 on learning symmetric fixed-rank positive semidefinite matrices. The proposed algorithms apply to a large number of applications and compete with the state-of-the-art on preliminary experiments.

Chapter 6

From first-order to second-order optimization algorithms

Chapter abstract: This chapter presents recent and ongoing research on rank-constrained second-order optimization algorithms. Our purpose is to provide the necessary material for the derivation of second-order optimization algorithms for learning a fixed-rank matrix.

We propose convenient formulas for computing the Riemannian connections associated with the quotient manifolds presented in Chapters 4 and 5. We then exploit those formulas to design novel trust-region algorithms for two matrix completion problems.

We show that for these problems, the underlying sparse structure allows us to compute efficiently the Riemannian Hessian. The proposed trust-region algorithms enjoy superlinear convergence properties and maintain a linear complexity both in the leading matrix dimension and in the number of available observations.

Numerical experiments illustrate the potential of the proposed trust-region algorithms.

Matrix and distance matrix completion are chosen as illustrative applications, but the proposed algorithms can be easily extended to other fixed-rank linear regression problems.

The material of this chapter is part of the following ongoing works:

B. Mishra, G. Meyer and R. Sepulchre
Low-rank optimization for distance matrix completion
Submitted to the 50th IEEE Conference on Decision and Control, Orlando (USA), 2011.

B. Mishra, G. Meyer and R. Sepulchre
Low-rank optimization for large-scale non-symmetric problems
Technical report, University of Liège, Belgium, 2011.

6.1 Introduction

Most developed algorithms for learning a low-rank matrix are first-order optimization methods. Although first-order methods cannot yield as high numerical precision as second-order methods, they are often preferred because they can be applied to much larger problems. The purpose of this chapter is to demonstrate that efficient second-order algorithms for learning a low-rank matrix can be obtained. The proposed approach combines the computational advantages of fixed-rank matrix factorizations with the theory of Riemannian trust-region algorithms.

Trust-region methods on Riemannian manifolds have been well-studied and come with a well-characterized convergence theory (Absil et al., 2007, 2008). With a proper choice of algorithm parameters, they enjoy superlinear converge to a local minimum of the considered cost function.

A potential limitation in the use of fixed-rank matrix factorizations for the design of second-

order algorithms is that the invariance properties of the factorization induce a cost function with non-isolated critical points. This is indeed a source of difficulty for most existing second-order algorithms that might prevent the algorithm to converge (see Absil et al., 2009). Existing algorithms have addressed this issue with ad hoc heuristics that consist in normalizing the considered factorization (Kearsley et al., 1998) or adding a symmetry breaking penalization term to the considered cost function (Tarazaga and Trosset, 1993).

In contrast to the existing heuristics, the proposed approach lifts the invariance properties of fixed-rank factorizations in the search space and reformulates the initial problem on a quotient manifold. The proposed approach is cheaper from a computational point of view, preserves the initial problem geometry, and renders the critical points of the cost function isolated.

The material associated with Riemannian trust-region algorithms is presented in Section 3.5.

With the material presented in Chapters 4 and 5, the only missing ingredient for the derivation of Riemannian trust-region algorithms for learning a fixed-rank matrix is convenient formulas for computing the Riemannian connection on the considered quotient manifolds.

We will provide such formulas and exploit them to develop trust-region algorithms for matrix completion and distance matrix completion. The sparse structure of these problems allows us to compute efficiently the Riemannian Hessian resulting in trust-region algorithms that scale linearly with the problem dimension.

6.2 Formulas for the Riemannian connection

In this section, we derive closed-form formulas for the Riemannian connections associated with the quotient manifold geometries presented in Chapters 4 and 5.

The followed approach crucially relies on the structure of a Riemannian quotient manifold. We will make use of formula (3.34) that relates the Riemannian connection in the quotient space to the Riemannian connection in the total space. When needed, the Riemannian connections in the total space will be identified from Koszul formula (3.26).

6.2.1 Riemannian connections on $S_+(r, d)$

This section presents the Riemannian connections associated with the two quotient geometries

$$S_+(r,d) \simeq \mathbb{R}_*^{d\times r}/\mathcal{O}(r) \simeq (\mathrm{St}(r,d) \times S_{++}(r))/\mathcal{O}(r). \tag{6.1}$$

Riemannian connection on $\mathbb{R}_*^{d\times r}/\mathcal{O}(r)$

Let $\xi_{[\mathbf{G}]}, \zeta_{[\mathbf{G}]}$ be two vector fields on the quotient manifold and $\bar{\xi}_{\mathbf{G}}, \bar{\zeta}_{\mathbf{G}}$ the associated horizontal vector fields. A straight application of the material presented in Journée et al. (2010) gives us the following formula for the Riemannian connection on $\mathbb{R}_*^{d\times r}/\mathcal{O}(r)$,

$$\overline{\nabla_{\xi_{[\mathbf{G}]}}\, \zeta_{[\mathbf{G}]}} = P^{\mathcal{H}}_{\mathbf{G}}(D\bar{\zeta}_{\mathbf{G}}[\bar{\xi}_{\mathbf{G}}]). \tag{6.2}$$

The projection of a tangent vector $\xi_{\mathbf{G}} \in \mathbb{R}^{d\times r}$ in the total space onto the horizontal space is

$$P^{\mathcal{H}}_{\mathbf{G}}(\xi_{\mathbf{G}}) = \xi_{\mathbf{G}} - \mathbf{G}\boldsymbol{\Omega},$$

where the skew-symmetric matrix $\boldsymbol{\Omega} \in \mathbb{R}^{r\times r}$ is the solution of the Sylvester equation

$$\boldsymbol{\Omega}\mathbf{G}^T\mathbf{G} + \mathbf{G}^T\mathbf{G}\boldsymbol{\Omega} = \mathbf{G}^T\xi_{\mathbf{G}} - \xi_{\mathbf{G}}^T\mathbf{G}.$$

This matrix equation can be transformed to a diagonal system of linear equations using an eigendecomposition of $\mathbf{G}^T\mathbf{G}$. The computation of $\mathbf{G}^T\mathbf{G}$ and $\mathbf{G}^T\xi_{\mathbf{G}}$ requires $O(dr^2)$ operations, and the resolution of the previous equation can be performed in $O(r^3)$ operations.

Riemannian connection on $(\mathrm{St}(r,d) \times S_{++}(r))/\mathcal{O}(r)$

Formula (3.34) that relates the Riemannian connection in the quotient space to the Riemannian connection in the total space only applies to a Riemannian quotient manifold.

The quotient geometry introduced in Bonnabel and Sepulchre (2009) is not a Riemannian quotient manifold because the vertical and horizontal spaces are not chosen orthogonal with respect to the Riemannian metric. As a preliminary step to the derivation of the Riemannian connection on $(\mathrm{St}(r,d) \times S_{++}(r))/\mathcal{O}(r)$, we generalize the quotient geometry introduced by Bonnabel and Sepulchre (2009) and turn it into a Riemannian quotient manifold.

Proposition 6.2.1. *The quotient space* $(\mathrm{St}(r,d) \times S_{++}(r))/\mathcal{O}(r)$ *admits the structure of a Riemannian quotient manifold when* $\mathrm{St}(r,d) \times S_{++}(r)$ *is endowed with the Riemannian metric*

$$\bar{g}_{(\mathbf{U},\mathbf{R}^2)}((\xi_\mathbf{U},\xi_{\mathbf{R}^2}),(\zeta_\mathbf{U},\zeta_{\mathbf{R}^2})) = \mathrm{Tr}(\xi_\mathbf{U}^T\zeta_\mathbf{U}) + \mathrm{Tr}(\mathbf{R}^{-2}\xi_{\mathbf{R}^2}\mathbf{R}^{-2}\zeta_{\mathbf{R}^2}), \tag{6.3}$$

and when the set of associated horizontal vectors $(\bar{\xi}_\mathbf{U},\bar{\xi}_{\mathbf{R}^2})$ *is defined by*

$$\begin{aligned} \bar{\xi}_\mathbf{U} &= \mathbf{U}\mathrm{Skew}(\mathbf{A}) + \mathbf{U}_\perp, \quad && \mathbf{A} \in \mathbb{R}^{r\times r},\ \mathbf{U}_\perp \in \mathbb{R}_*^{d\times r} \text{ and } \mathbf{U}_\perp^T\mathbf{U} = 0, \\ \bar{\xi}_{\mathbf{R}^2} &= \mathrm{Sym}(\mathbf{D}), && \mathbf{D} \in \mathbb{R}^{r\times r}, \end{aligned}$$

with the additional requirement that

$$\mathrm{Skew}(\mathbf{A}) = \mathbf{R}^{-2}\bar{\xi}_{\mathbf{R}^2} - \bar{\xi}_{\mathbf{R}^2}\mathbf{R}^{-2}. \tag{6.4}$$

Proof. See Appendix A. □

In the special case $\mathbf{A} = 0$, we recover the geometry proposed by Bonnabel and Sepulchre (2009). The projection onto the set of horizontal vectors is defined as follows.

Proposition 6.2.2. *Let* $(\xi_\mathbf{U},\xi_{\mathbf{R}^2})$ *be a tangent vector in the total space* $\mathrm{St}(r,d) \times S_{++}(r)$*, its projection onto the set of horizontal vectors is given by*

$$P^\mathcal{H}_{(\mathbf{U},\mathbf{R}^2)}(\xi_\mathbf{U},\xi_{\mathbf{R}^2}) = (\xi_\mathbf{U} - \mathbf{U}\mathbf{\Omega},\ \xi_{\mathbf{R}^2} - \mathbf{R}^2\mathbf{\Omega} + \mathbf{\Omega}\mathbf{R}^2), \tag{6.5}$$

where $\mathbf{\Omega}^T = -\mathbf{\Omega} \in \mathbb{R}^{r\times r}$ *is the unique solution of the Sylvester equation*

$$\mathbf{\Omega}\mathbf{R}^4 + \mathbf{R}^4\mathbf{\Omega} = \mathbf{R}^2\mathrm{Skew}(\mathbf{U}^T\xi_\mathbf{U})\mathbf{R}^2 + \mathbf{R}^2\mathrm{Sym}(\xi_{\mathbf{R}^2}) - \mathrm{Sym}(\xi_{\mathbf{R}^2})\mathbf{R}^2. \tag{6.6}$$

Proof. See Appendix A □

Let $\xi_{[(\mathbf{U},\mathbf{R}^2)]},\zeta_{[(\mathbf{U},\mathbf{R}^2)]}$ be two vector fields on the quotient manifold and $(\bar{\xi}_\mathbf{U},\bar{\xi}_{\mathbf{R}^2}),(\bar{\zeta}_\mathbf{U},\bar{\zeta}_{\mathbf{R}^2})$ the associated horizontal vector fields. Exploiting formula 3.34, the Riemannian connection on the quotient manifold $(\mathrm{St}(r,d) \times S_{++}(r))/\mathcal{O}(r)$ is given by

$$\overline{\nabla_{\xi_{[(\mathbf{U},\mathbf{R}^2)]}}\zeta_{[(\mathbf{U},\mathbf{R}^2)]}} = P^\mathcal{H}_{(\mathbf{U},\mathbf{R}^2)}(\overline{\nabla}_{(\bar{\xi}_\mathbf{U},\bar{\xi}_{\mathbf{R}^2})}\bar{\zeta}_\mathbf{U},\ \overline{\nabla}_{(\bar{\xi}_\mathbf{U},\bar{\xi}_{\mathbf{R}^2})}\bar{\zeta}_{\mathbf{R}^2}), \tag{6.7}$$

where

$$\begin{aligned} \overline{\nabla}_{(\bar{\xi}_\mathbf{U},\bar{\xi}_{\mathbf{R}^2})}\bar{\zeta}_\mathbf{U} &= P_\mathbf{U}(D\bar{\zeta}_\mathbf{U}[\bar{\xi}_\mathbf{U},\bar{\xi}_{\mathbf{R}^2}]), \\ \overline{\nabla}_{(\bar{\xi}_\mathbf{U},\bar{\xi}_{\mathbf{R}^2})}\bar{\zeta}_{\mathbf{R}^2} &= D\bar{\zeta}_{\mathbf{R}^2}[\bar{\xi}_\mathbf{U},\bar{\xi}_{\mathbf{R}^2}] - \mathrm{Sym}(\bar{\xi}_{\mathbf{R}^2}\mathbf{R}^{-2}\bar{\zeta}_{\mathbf{R}^2}). \end{aligned}$$

The derivation of the Riemannian connection $\overline{\nabla}_{(\bar{\xi}_\mathbf{U},\bar{\xi}_{\mathbf{R}^2})}\bar{\zeta}_{\mathbf{R}^2}$ on $S_{++}(r)$ is provided in Appendix B.

6.2.2 Riemannian connections on $\mathcal{F}(r, d_1, d_2)$

This section presents the Riemannian connections associated with the two quotient geometries

$$\mathcal{F}(r, d_1, d_2) \simeq (\mathbb{R}_*^{d_1 \times r} \times \mathbb{R}_*^{d_2 \times r})/\mathrm{GL}(r) \simeq (\mathrm{St}(r, d_1) \times S_{++}(r) \times \mathrm{St}(r, d_2))/\mathcal{O}(r),$$

The two geometries have the structure of a Riemannian quotient manifold (see Chapter 5).

Riemannian connection on $(\mathbb{R}_*^{d_1 \times r} \times \mathbb{R}_*^{d_2 \times r})/\mathrm{GL}(r)$

We first define the projection onto the set of horizontal vectors.

Proposition 6.2.3. *Let* $(\xi_{\mathbf{G}}, \xi_{\mathbf{H}})$ *be a tangent vector in the total space* $\mathbb{R}_*^{d_1 \times r} \times \mathbb{R}_*^{d_2 \times r}$*, its projection onto the set of horizontal vectors is given by*

$$P^{\mathcal{H}}_{(\mathbf{G},\mathbf{H})}(\xi_{\mathbf{G}}, \xi_{\mathbf{H}}) = (\xi_{\mathbf{G}} + \mathbf{G}\mathbf{\Lambda},\ \xi_{\mathbf{H}} - \mathbf{H}\mathbf{\Lambda}^T)$$

where $\mathbf{\Lambda} \in \mathbb{R}^{r \times r}$ *is the unique solution of the Sylvester equation*

$$(\mathbf{H}^T\mathbf{H})(\mathbf{G}^T\mathbf{G})\mathbf{\Lambda} + \mathbf{\Lambda}(\mathbf{H}^T\mathbf{H})(\mathbf{G}^T\mathbf{G}) = \xi_{\mathbf{H}}^T\mathbf{H}(\mathbf{G}^T\mathbf{G}) - (\mathbf{H}^T\mathbf{H})\mathbf{G}^T\xi_{\mathbf{G}}. \tag{6.8}$$

Proof. See Appendix A □

The solution of the Sylvester equation (6.8) can be obtained in $O(r^3)$ operations using a standard solver such as SLICOT-SB04MD which is embedded in the Matlab function `lyap`.

Consider $\xi_{[(\mathbf{G},\mathbf{H})]}, \zeta_{[(\mathbf{G},\mathbf{H})]}$ two vector fields on the quotient manifold and $(\bar{\xi}_{\mathbf{G}}, \bar{\xi}_{\mathbf{H}}), (\bar{\zeta}_{\mathbf{G}}, \bar{\zeta}_{\mathbf{H}})$ the associated horizontal vector fields. The Riemannian connection for the quotient manifold $(\mathbb{R}_*^{d_1 \times r} \times \mathbb{R}_*^{d_2 \times r})/\mathrm{GL}(r)$ is defined by

$$\overline{\nabla_{\xi_{[(\mathbf{G},\mathbf{H})]}} \zeta_{[(\mathbf{G},\mathbf{H})]}} = P^{\mathcal{H}}_{(\mathbf{G},\mathbf{H})}(\overline{\nabla}_{(\bar{\xi}_{\mathbf{G}}, \bar{\xi}_{\mathbf{H}})}\bar{\zeta}_{\mathbf{G}}, \overline{\nabla}_{(\bar{\xi}_{\mathbf{G}}, \bar{\xi}_{\mathbf{H}})}\bar{\zeta}_{\mathbf{H}}), \tag{6.9}$$

where

$$\begin{aligned}
\overline{\nabla}_{(\bar{\xi}_{\mathbf{G}}, \bar{\xi}_{\mathbf{H}})}\bar{\zeta}_{\mathbf{G}} = D\bar{\zeta}_{\mathbf{G}}[\bar{\xi}_{\mathbf{G}}, \bar{\xi}_{\mathbf{H}}] &- \bar{\zeta}_{\mathbf{G}}(\mathbf{G}^T\mathbf{G})^{-1}\mathrm{Sym}(\bar{\xi}_{\mathbf{G}}^T\mathbf{G}) - \bar{\xi}_{\mathbf{G}}(\mathbf{G}^T\mathbf{G})^{-1}\mathrm{Sym}(\bar{\zeta}_{\mathbf{G}}^T\mathbf{G}) \\
&+ \mathbf{G}(\mathbf{G}^T\mathbf{G})^{-1}\mathrm{Sym}(\bar{\zeta}_{\mathbf{G}}^T\bar{\xi}_{\mathbf{G}}), \\
\overline{\nabla}_{(\bar{\xi}_{\mathbf{G}}, \bar{\xi}_{\mathbf{H}})}\bar{\zeta}_{\mathbf{H}} = D\bar{\zeta}_{\mathbf{H}}[\bar{\xi}_{\mathbf{G}}, \bar{\xi}_{\mathbf{H}}] &- \bar{\zeta}_{\mathbf{H}}(\mathbf{H}^T\mathbf{H})^{-1}\mathrm{Sym}(\bar{\xi}_{\mathbf{H}}^T\mathbf{H}) - \bar{\xi}_{\mathbf{H}}(\mathbf{H}^T\mathbf{H})^{-1}\mathrm{Sym}(\bar{\zeta}_{\mathbf{H}}^T\mathbf{H}) \\
&+ \mathbf{H}(\mathbf{H}^T\mathbf{H})^{-1}\mathrm{Sym}(\bar{\zeta}_{\mathbf{H}}^T\bar{\xi}_{\mathbf{H}}).
\end{aligned}$$

The derivation of these two formulas is provided in Appendix B.

Riemannian connection on $(\mathrm{St}(r, d_1) \times S_{++}(r) \times \mathrm{St}(r, d_2))/\mathcal{O}(r)$

The projection onto the set of horizontal vectors is defined as follows.

Proposition 6.2.4. *Let* $(\xi_{\mathbf{U}}, \xi_{\mathbf{B}}, \xi_{\mathbf{V}})$ *be a tangent vector in the total space* $\mathrm{St}(r, d_1) \times S_{++}(r) \times \mathrm{St}(r, d_2)$*, its projection onto the set of horizontal vectors is given by*

$$P^{\mathcal{H}}_{(\mathbf{U},\mathbf{B},\mathbf{V})}(\xi_{\mathbf{U}}, \xi_{\mathbf{B}}, \xi_{\mathbf{V}}) = (\xi_{\mathbf{U}} - \mathbf{U}\mathbf{\Omega},\ ,\xi_{\mathbf{B}} - \mathbf{B}\mathbf{\Omega} + \mathbf{\Omega}\mathbf{B},\ \xi_{\mathbf{V}} - \mathbf{V}\mathbf{\Omega}) \tag{6.10}$$

where $\mathbf{\Omega}^T = -\mathbf{\Omega} \in \mathbb{R}^{r \times r}$ *is the unique solution of the Sylvester equation*

$$\mathbf{\Omega}\mathbf{B}^2 + \mathbf{B}^2\mathbf{\Omega} = \mathbf{B}(\mathrm{Skew}(\mathbf{U}^T\xi_{\mathbf{U}}) + \mathrm{Skew}(\mathbf{V}^T\xi_{\mathbf{V}}))\mathbf{B} + \mathbf{B}\mathrm{Sym}(\xi_{\mathbf{B}}) - \mathrm{Sym}(\xi_{\mathbf{B}})\mathbf{B}. \tag{6.11}$$

Proof. See Appendix A □

Consider $\xi_{[(\mathbf{U},\mathbf{B},\mathbf{V})]}, \zeta_{[(\mathbf{U},\mathbf{B},\mathbf{V})]}$ two vector fields on the quotient manifold and their associated horizontal vector fields $(\bar{\xi}_{\mathbf{U}}, \bar{\xi}_{\mathbf{B}}, \bar{\xi}_{\mathbf{V}}), (\bar{\zeta}_{\mathbf{U}}, \bar{\zeta}_{\mathbf{B}}, \bar{\zeta}_{\mathbf{V}})$. The Riemannian connection for the quotient manifold $(\mathrm{St}(r, d_1) \times S_{++}(r) \times \mathrm{St}(r, d_2))/\mathcal{O}(r)$ is defined by

$$\overline{\nabla_{\xi_{[(\mathbf{U},\mathbf{B},\mathbf{V})]}} \zeta_{[(\mathbf{U},\mathbf{B},\mathbf{V})]}} = P^{\mathcal{H}}_{(\mathbf{U},\mathbf{B},\mathbf{V})}(\overline{\nabla}_{(\bar{\xi}_{\mathbf{U}},\bar{\xi}_{\mathbf{B}},\bar{\xi}_{\mathbf{V}})}\bar{\zeta}_{\mathbf{U}}, \overline{\nabla}_{(\bar{\xi}_{\mathbf{U}},\bar{\xi}_{\mathbf{B}},\bar{\xi}_{\mathbf{V}})}\bar{\zeta}_{\mathbf{B}}, \overline{\nabla}_{(\bar{\xi}_{\mathbf{U}},\bar{\xi}_{\mathbf{B}},\bar{\xi}_{\mathbf{V}})}\bar{\zeta}_{\mathbf{V}}), \tag{6.12}$$

where

$$\begin{aligned}
\overline{\nabla}_{(\bar{\xi}_{\mathbf{U}},\bar{\xi}_{\mathbf{B}},\bar{\xi}_{\mathbf{V}})}\bar{\zeta}_{\mathbf{U}} &= P_{\mathbf{U}}(D\bar{\zeta}_{\mathbf{U}}[\bar{\xi}_{\mathbf{U}}, \bar{\xi}_{\mathbf{B}}, \bar{\xi}_{\mathbf{V}}]), \\
\overline{\nabla}_{(\bar{\xi}_{\mathbf{U}},\bar{\xi}_{\mathbf{B}},\bar{\xi}_{\mathbf{V}})}\bar{\zeta}_{\mathbf{B}} &= D\bar{\zeta}_{\mathbf{B}}[\bar{\xi}_{\mathbf{U}}, \bar{\xi}_{\mathbf{B}}, \bar{\xi}_{\mathbf{V}}] - \mathrm{Sym}(\bar{\xi}_{\mathbf{B}}\mathbf{B}^{-1}\bar{\zeta}_{\mathbf{B}}), \\
\overline{\nabla}_{(\bar{\xi}_{\mathbf{U}},\bar{\xi}_{\mathbf{B}},\bar{\xi}_{\mathbf{V}})}\bar{\zeta}_{\mathbf{V}} &= P_{\mathbf{V}}(D\bar{\zeta}_{\mathbf{V}}[\bar{\xi}_{\mathbf{U}}, \bar{\xi}_{\mathbf{B}}, \bar{\xi}_{\mathbf{V}}]).
\end{aligned}$$

6.3 Computing the Riemannian Hessian

In this section, we provide formulas for computing efficiently the Riemannian Hessian associated with the cost function of two matrix completion problems. We show how to exploit the sparse structure of the underlying problem to maintain a linear complexity in the number of available observations and in the leading matrix dimension.

6.3.1 Riemannian Hessian for low-rank distance matrix completion

We consider the distance matrix completion formulation (2.5.4) with the cost function

$$f : S_+(r, n) \to : \mathbf{W} \mapsto \sum_{(i,j)\in\mathcal{D}} (\hat{y}_{ij} - y_{ij})^2, \tag{6.13}$$

where $\hat{y}_{ij} = \mathrm{Tr}((\mathbf{W}(\mathbf{e}_i - \mathbf{e}_j)(\mathbf{e}_i - \mathbf{e}_j)^T)$ and $\mathbf{e}_i, \mathbf{e}_j \in \mathbb{R}^n$ are canonical basis vectors. The total number of available distances y_{ij} is $|\mathcal{D}|$. We first construct a n-by-$|\mathcal{D}|$ sparse matrix $\mathbf{E}$ whose columns are defined as $(\mathbf{e}_i - \mathbf{e}_j)$, for all observations $(i, j) \in \mathcal{D}$. Each column of $\mathbf{E}$ thus contains two elements and the number of nonzero elements in $\mathbf{E}$ is $2|\mathcal{D}|$. We also define the $|\mathcal{D}|$-by-$|\mathcal{D}|$ diagonal matrix $\mathbf{\Delta}$ which contains all signed prediction errors $(\hat{y}_{ij} - y_{ij})$ on the diagonal.

Riemannian Hessian on $\mathbb{R}^{d\times r}_*/\mathcal{O}(r)$

With the previous definitions, we compute the Riemannian Hessian of the cost function

$$f(\mathbf{G}) = \sum_{(i,j)\in\mathcal{D}} ((\mathbf{e}_i - \mathbf{e}_j)^T\mathbf{G}\mathbf{G}^T(\mathbf{e}_i - \mathbf{e}_j) - y_{ij})^2 \tag{6.14}$$

on the quotient manifold $\mathbb{R}^{d\times r}_*/\mathcal{O}(r)$. The Riemannian Hessian of this cost function is obtained by applying the Riemannian connection (6.2) to the gradient vector field

$$\overline{\mathrm{grad} f}(\mathbf{G}) = 4 \sum_{(i,j)\in\mathcal{D}} (\hat{y}_{ij} - \hat{y}_{ij})(\mathbf{e}_i - \mathbf{e}_j)(\mathbf{e}_i - \mathbf{e}_j)^T\mathbf{G} = 4\mathbf{E}\mathbf{\Delta}\mathbf{E}^T\mathbf{G}. \tag{6.15}$$

This gradient can be evaluated in $O(|\mathcal{D}|r)$ operations. We obtain the Riemannian Hessian as

$$\mathrm{Hess} f(\mathbf{G})[\bar{\xi}_{\mathbf{G}}] = 4P^{\mathcal{H}}_{\mathbf{G}}(\mathbf{E}\mathbf{\Delta}\mathbf{E}^T\bar{\xi}_{\mathbf{G}} + \mathbf{E}\tilde{\mathbf{\Delta}}\mathbf{E}^T\mathbf{G}), \tag{6.16}$$

where $\tilde{\mathbf{\Delta}} = 2\mathrm{Diag}((\mathbf{E}^T\mathbf{G})(\bar{\xi}^T_{\mathbf{G}}\mathbf{E}))$. Computing (6.16) requires $O(|\mathcal{D}|r + nr^2 + r^3)$ operations.

Riemannian Hessian on $(\mathrm{St}(r,d) \times S_{++}(r))/\mathcal{O}(r)$

We now compute the Riemannian Hessian of the cost function

$$f(\mathbf{U}, \mathbf{R}^2) = \sum_{(i,j)\in\mathcal{D}} ((\mathbf{e}_i - \mathbf{e}_j)^T \mathbf{U}\mathbf{R}^2\mathbf{U}^T(\mathbf{e}_i - \mathbf{e}_j) - y_{ij})^2 \tag{6.17}$$

on the quotient manifold $(\mathrm{St}(r,d)\times S_{++}(r))/\mathcal{O}(r)$. The Riemannian Hessian of this cost function is obtained by applying the Riemannian connection (6.7) to the gradient vector field

$$\begin{aligned}
\overline{\mathrm{grad}_{\mathbf{U}} f} &= 4 \sum_{(i,j)\in\mathcal{D}} (\hat{y}_{ij} - \hat{y}_{ij}) P_{\mathbf{U}}((\mathbf{e}_i - \mathbf{e}_j)(\mathbf{e}_i - \mathbf{e}_j)^T \mathbf{U}\mathbf{R}^2) = 4P_{\mathbf{U}}(\mathbf{E}\boldsymbol{\Delta}\mathbf{E}^T\mathbf{U}\mathbf{R}^2), \\
\overline{\mathrm{grad}_{\mathbf{R}^2} f} &= 2 \sum_{(i,j)\in\mathcal{D}} (\hat{y}_{ij} - \hat{y}_{ij}) \mathbf{R}^2(\mathbf{U}^T(\mathbf{e}_i - \mathbf{e}_j)(\mathbf{e}_i - \mathbf{e}_j)^T \mathbf{U})\mathbf{R}^2 = 2\mathbf{R}^2(\mathbf{U}^T\mathbf{E}\boldsymbol{\Delta}\mathbf{E}^T\mathbf{U})\mathbf{R}^2.
\end{aligned}$$

The gradient can be evaluated in $O(|\mathcal{D}|r + dr^2 + r^2)$ operations. The Riemannian Hessian is

$$\mathrm{Hess} f(\mathbf{U}, \mathbf{R}^2)[\bar{\xi}_{\mathbf{U}}, \bar{\xi}_{\mathbf{R}^2}] = P^{\mathcal{H}}_{(\mathbf{U},\mathbf{R}^2)}(\mathrm{Hess}_{\mathbf{U}} f, \mathrm{Hess}_{\mathbf{R}^2}), \tag{6.18}$$

where

$$\begin{aligned}
\mathrm{Hess}_{\mathbf{U}} f &= 4P_{\mathbf{U}}(\mathbf{E}\boldsymbol{\Delta}\mathbf{E}^T(\mathbf{U}\bar{\xi}_{\mathbf{R}^2} + \bar{\xi}_{\mathbf{U}}\mathbf{R}^2) + \mathbf{E}\tilde{\boldsymbol{\Delta}}\mathbf{E}^T\mathbf{U}\mathbf{R}^2 - \bar{\xi}_{\mathbf{U}}\mathrm{Sym}(\mathbf{U}^T\mathbf{E}\boldsymbol{\Delta}\mathbf{E}^T\mathbf{U}\mathbf{R}^2)), \\
\mathrm{Hess}_{\mathbf{R}^2} f &= \bar{\xi}_{\mathbf{R}^2}(\mathbf{E}\boldsymbol{\Delta}\mathbf{E}^T)\mathbf{R}^2 + \mathbf{R}^2(\mathbf{E}\boldsymbol{\Delta}\mathbf{E}^T)\bar{\xi}_{\mathbf{R}^2} - \mathrm{Sym}(\bar{\xi}_{\mathbf{R}^2}\mathbf{R}^{-2}\overline{\mathrm{grad}_{\mathbf{R}^2} f}) \\
&\quad + \mathbf{R}^2(\bar{\xi}_{\mathbf{U}}^T\mathbf{E}\boldsymbol{\Delta}\mathbf{E}^T\mathbf{U} + \mathbf{U}^T\mathbf{E}\boldsymbol{\Delta}\mathbf{E}^T\bar{\xi}_{\mathbf{U}} + \mathbf{U}^T\mathbf{E}\tilde{\boldsymbol{\Delta}}\mathbf{E}^T\mathbf{U})\mathbf{R}^2.
\end{aligned}$$

and $\tilde{\boldsymbol{\Delta}} = 2\mathrm{Diag}((\mathbf{E}^T\mathbf{U}\bar{\xi}_{\mathbf{R}^2})(\mathbf{U}^T\mathbf{E}) + 2(\mathbf{E}^T\mathbf{U}\mathbf{R}^2)(\bar{\xi}_{\mathbf{U}}^T\mathbf{E}))$. With the projection step, the computation of the Riemannian Hessian (6.18) can be performed in $O(|\mathcal{D}|r + nr^2 + r^3)$ operations.

6.3.2 Riemannian Hessian for low-rank matrix completion

We consider the low-rank matrix completion problem (2.5) with the cost function

$$f : \mathcal{F}(r, d_1, d_2) \to : \mathbf{W} \mapsto \sum_{(i,j)\in\Omega} (\hat{y}_{ij} - y_{ij})^2, \tag{6.19}$$

where $\hat{y}_{ij} = \mathrm{Tr}(\mathbf{W}\mathbf{e}_j\mathbf{e}_i^T)$ and with the canonical basis vectors $\mathbf{e}_i \in \mathbb{R}^{d_1}$ and $\mathbf{e}_j \in \mathbb{R}^{d_2}$. The total number of available entries $y_{ij} = \mathbf{W}^\star_{ij}$ is $|\Omega|$. We define the d_1-by-d_2 sparse matrix

$$\mathbf{S} = \sum_{(i,j)\in\Omega} (\hat{y}_{ij} - y_{ij})\mathbf{e}_i\mathbf{e}_j^T,$$

which contains at most $|\Omega|$ nonzero elements.

Riemannian Hessian on $(\mathbb{R}_*^{d_1\times r} \times \mathbb{R}_*^{d_2\times r})/\mathrm{GL}(r)$

With the previous definitions, we compute the Riemannian Hessian of the cost function

$$f(\mathbf{G}, \mathbf{H}) = \sum_{(i,j)\in\Omega} (\mathbf{e}_i^T\mathbf{G}\mathbf{H}^T\mathbf{e}_j - y_{ij})^2 \tag{6.20}$$

on the quotient manifold $(\mathbb{R}_*^{d_1\times r} \times \mathbb{R}_*^{d_2\times r})/\mathrm{GL}(r)$. The Riemannian Hessian of this cost function is obtained by applying the Riemannian connection (6.9) to the gradient vector field

$$\begin{aligned}
\overline{\mathrm{grad}_{\mathbf{G}} f} &= 2 \sum_{(i,j)\in\Omega} (\hat{y}_{ij} - \hat{y}_{ij})\mathbf{e}_i\mathbf{e}_j^T\mathbf{H}(\mathbf{G}^T\mathbf{G}) = 2\mathbf{S}\mathbf{H}(\mathbf{G}^T\mathbf{G}), \\
\overline{\mathrm{grad}_{\mathbf{H}} f} &= 2 \sum_{(i,j)\in\Omega} (\hat{y}_{ij} - \hat{y}_{ij})\mathbf{e}_j\mathbf{e}_i^T\mathbf{G}(\mathbf{H}^T\mathbf{H}) = 2\mathbf{S}^T\mathbf{G}(\mathbf{H}^T\mathbf{H}).
\end{aligned}$$

The gradient can be evaluated in $O(|\Omega|r + (d_1 + d_2)r^2)$ operations. The Riemannian Hessian is

$$\mathrm{Hess} f(\mathbf{G}, \mathbf{H})[\bar{\xi}_{\mathbf{G}}, \bar{\xi}_{\mathbf{H}}] = P^{\mathcal{H}}_{(\mathbf{G},\mathbf{H})}(\mathrm{Hess}_{\mathbf{G}} f, \mathrm{Hess}_{\mathbf{H}} f), \tag{6.21}$$

where

$$\begin{aligned}
\mathrm{Hess}_{\mathbf{G}} f &= \mathbf{S}(\mathbf{H}\mathrm{Sym}(\mathbf{G}^T\bar{\xi}_{\mathbf{G}}) + \bar{\xi}_{\mathbf{H}}(\mathbf{G}^T\mathbf{G})) + \bar{\mathbf{S}}\mathbf{H}(\mathbf{G}^T\mathbf{G}) \\
&\quad - \bar{\xi}_{\mathbf{G}}(\mathbf{G}^T\mathbf{G})^{-1}\mathrm{Sym}(\mathbf{G}^T\overline{\mathrm{grad}_{\mathbf{G}} f}) + \mathbf{G}(\mathbf{G}^T\mathbf{G})^{-1}\mathrm{Sym}(\mathbf{G}^T\bar{\xi}_{\mathbf{G}}), \\
\mathrm{Hess}_{\mathbf{H}} f &= \mathbf{S}^T(\mathbf{G}\mathrm{Sym}(\mathbf{H}^T\bar{\xi}_{\mathbf{H}}) + \bar{\xi}_{\mathbf{G}}(\mathbf{H}^T\mathbf{H})) + \bar{\mathbf{S}}^T\mathbf{G}(\mathbf{H}^T\mathbf{H}) \\
&\quad - \bar{\xi}_{\mathbf{H}}(\mathbf{H}^T\mathbf{H})^{-1}\mathrm{Sym}(\mathbf{H}^T\overline{\mathrm{grad}_{\mathbf{H}} f}) + \mathbf{H}(\mathbf{H}^T\mathbf{H})^{-1}\mathrm{Sym}(\mathbf{H}^T\bar{\xi}_{\mathbf{H}}),
\end{aligned}$$

and

$$\bar{\mathbf{S}} = \sum_{(i,j)\in\Omega} (\mathbf{e}_i^T(\mathbf{G}\bar{\xi}_{\mathbf{H}}^T + \bar{\xi}_{\mathbf{G}}\mathbf{H}^T)\mathbf{e}_j)\mathbf{e}_i\mathbf{e}_j^T.$$

The Riemannian Hessian (6.22) can be computed in $O(|\Omega|r + (d_1 + d_2)r^2 + r^3)$ operations.

Riemannian Hessian on $(\mathrm{St}(r, d_1) \times S_{++}(r) \times \mathrm{St}(r, d_2))/\mathcal{O}(r)$

We now compute the Riemannian Hessian of the cost function

$$f(\mathbf{U}, \mathbf{B}, \mathbf{V}) = \sum_{(i,j)\in\Omega} (\mathbf{e}_i^T\mathbf{U}\mathbf{B}\mathbf{V}^T\mathbf{e}_j - y_{ij})^2$$

on the quotient manifold $(\mathrm{St}(r, d_1) \times S_{++}(r) \times \mathrm{St}(r, d_2))/\mathcal{O}(r)$. The Riemannian Hessian of this cost function is obtained by applying the Riemannian connection (6.12) to the gradient vector field

$$\begin{aligned}
\overline{\mathrm{grad}_{\mathbf{U}} f} &= 2\sum_{(i,j)\in\Omega} (\hat{y}_{ij} - \hat{y}_{ij})P_{\mathbf{U}}(\mathbf{e}_i\mathbf{e}_j^T\mathbf{V}\mathbf{B}) = 2P_{\mathbf{U}}(\mathbf{S}\mathbf{V}\mathbf{B}), \\
\overline{\mathrm{grad}_{\mathbf{B}} f} &= 2\sum_{(i,j)\in\Omega} (\hat{y}_{ij} - \hat{y}_{ij})\mathbf{B}\mathrm{Sym}(\mathbf{V}^T\mathbf{e}_j\mathbf{e}_i^T\mathbf{U})\mathbf{B} = 2\mathbf{B}\mathrm{Sym}(\mathbf{V}^T\mathbf{S}^T\mathbf{U})\mathbf{B}, \\
\overline{\mathrm{grad}_{\mathbf{V}} f} &= 2\sum_{(i,j)\in\Omega} (\hat{y}_{ij} - \hat{y}_{ij})P_{\mathbf{V}}(\mathbf{e}_j\mathbf{e}_i^T\mathbf{U}\mathbf{B}) = 2P_{\mathbf{V}}(\mathbf{S}^T\mathbf{U}\mathbf{B}),
\end{aligned}$$

The gradient can be evaluated in $O(|\Omega|r + (d_1 + d_2)r^2 + r^2)$ operations. We have

$$\mathrm{Hess} f(\mathbf{U}, \mathbf{B}, \mathbf{V})[\bar{\xi}_{\mathbf{U}}, \bar{\xi}_{\mathbf{B}}, \bar{\xi}_{\mathbf{V}}] = P^{\mathcal{H}}_{(\mathbf{U},\mathbf{B},\mathbf{V})}(\mathrm{Hess}_{\mathbf{U}} f, \mathrm{Hess}_{\mathbf{B}}, \mathrm{Hess}_{\mathbf{U}} f), \tag{6.22}$$

where

$$\begin{aligned}
\mathrm{Hess}_{\mathbf{U}} f &= P_{\mathbf{U}}(\mathbf{S}(\mathbf{V}\bar{\xi}_{\mathbf{B}} + \bar{\xi}_{\mathbf{V}}\mathbf{B}) + \bar{\mathbf{S}}\mathbf{V}\mathbf{B} - \bar{\xi}_{\mathbf{U}}\mathrm{Sym}(\mathbf{U}^T\mathbf{S}\mathbf{V}\mathbf{B})) \\
\mathrm{Hess}_{\mathbf{B}} f &= \bar{\xi}_{\mathbf{B}}\mathrm{Sym}(\mathbf{V}^T\mathbf{S}^T\mathbf{U})\mathbf{B} + \mathbf{B}\mathrm{Sym}(\mathbf{V}^T\mathbf{S}^T\mathbf{U})\bar{\xi}_{\mathbf{B}} \\
&\quad + \mathbf{B}\mathrm{Sym}(\bar{\xi}_{\mathbf{U}}\mathbf{S}\mathbf{V} + \mathbf{U}^T\bar{\mathbf{S}}\mathbf{V} + \mathbf{U}^T\mathbf{S}\bar{\xi}_{\mathbf{V}})\mathbf{B} - \mathrm{Sym}(\bar{\xi}_{\mathbf{B}}\mathbf{B}^{-1}\overline{\mathrm{grad}_{\mathbf{B}} f}) \\
\mathrm{Hess}_{\mathbf{U}} f &= P_{\mathbf{V}}(\mathbf{S}(\mathbf{U}\bar{\xi}_{\mathbf{B}} + \bar{\xi}_{\mathbf{U}}\mathbf{B}) + \bar{\mathbf{S}}\mathbf{U}\mathbf{B} - \bar{\xi}_{\mathbf{V}}\mathrm{Sym}(\mathbf{V}^T\mathbf{S}^T\mathbf{U}\mathbf{B}))
\end{aligned}$$

and

$$\bar{\mathbf{S}} = \sum_{(i,j)\in\Omega} (\mathbf{e}_i^T(\bar{\xi}_{\mathbf{U}}\mathbf{B}\mathbf{V}^T + \mathbf{U}\bar{\xi}_{\mathbf{B}}\mathbf{V}^T + \mathbf{U}\mathbf{B}\bar{\xi}_{\mathbf{V}}^T)\mathbf{e}_j)\mathbf{e}_i\mathbf{e}_j^T.$$

The Riemannian Hessian (6.22) can be computed in $O(|\Omega|r + (d_1 + d_2)r^2 + r^3)$ operations.

6.4 Experiments

Sections 6.4.1 and 6.4.2 present numerical experiments that illustrate the good behavior of the proposed trust-region algorithms on symmetric and non-symmetric matrix completion problems.

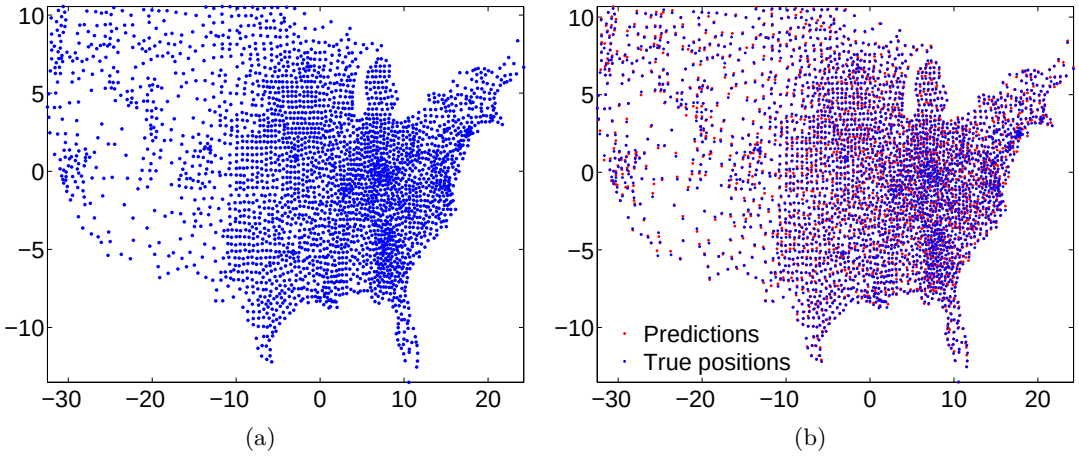
For these experiments, we choose the `GenRTR` Matlab toolbox (Absil et al., 2007) that only requires interface with manifold related functions (such as retraction, projection on the tangent space and metric) and cost related functions (such as evaluation of the cost function, gradient and Hessian).[1] The chosen implementation solves the trust-region subproblem (3.29) using a truncated conjugate-gradient algorithm (see Algorithm 11 in Absil et al., 2007, for more details).

Additional experiments are provided in the papers Mishra et al. (2011a,b).

6.4.1 Low-rank distance matrix completion

The experiments in this section aim at reconstructing the two-dimensional US Cities data embedding (Figure 6.1(a)) from a subset of pairwise distances selected uniformly and at random.

The US cities data set (Cucuringu et al., 2011) contains the positions of $n = 3075$ US cities. We sample 25% of the total number of pairwise distances $n(n-1)/2 = 4,726,275$ and learn a two dimensional Euclidean embedding of the data from these pairwise distances only. This results in learning from $|\mathcal{D}| = 1,181,569$ pairwise distances. Gaussian noise is superposed to the observed distances. The noise level is 10% of the standard deviation of the observed distances. All algorithms start from the same initial condition which is chosen at random and they all minimize the cost function (6.19).



(a) (b)

Figure 6.1: **Left:** exact data embedding for the US cities data set. **Right:** predictions (red dots) are very close to true positions (blue dots). The current figure is best viewed in color.

.

Excellent reconstruction of the two-dimensional data embedding is observed (Figure 6.1(b)). Since the algorithms learn from pairwise distances, they can only identify the embedding up to a rotation transform. As a post-processing step, we thus rotate the embedding such it appears as the original data. We only show the obtained result for the trust-region algorithm based on the flat geometry. The obtained results for the other algorithms are almost identical. Additional observations for this experiment are summarized in Table 6.1. All algorithms roughly achieve the same final value of the cost function. Regarding time required to achieve convergence, the algorithms based on the flat geometry are clearly the best performers for this problem.

[1] The toolbox is available from `http://www.math.fsu.edu/~cbaker/GenRTR/`.

Table 6.1: Statistics for the US Cities experiment

	Trust-Region		Gradient Descent	
	Flat Geometry	Polar Geometry	Flat Geometry	Polar Geometry
Time taken	2.89 minutes	6.09 minutes	2.22 minutes	41.05 minutes
Num. iter.	108	56	154	751
Cost function	1531.52	1531.52	1531.71	1533.05

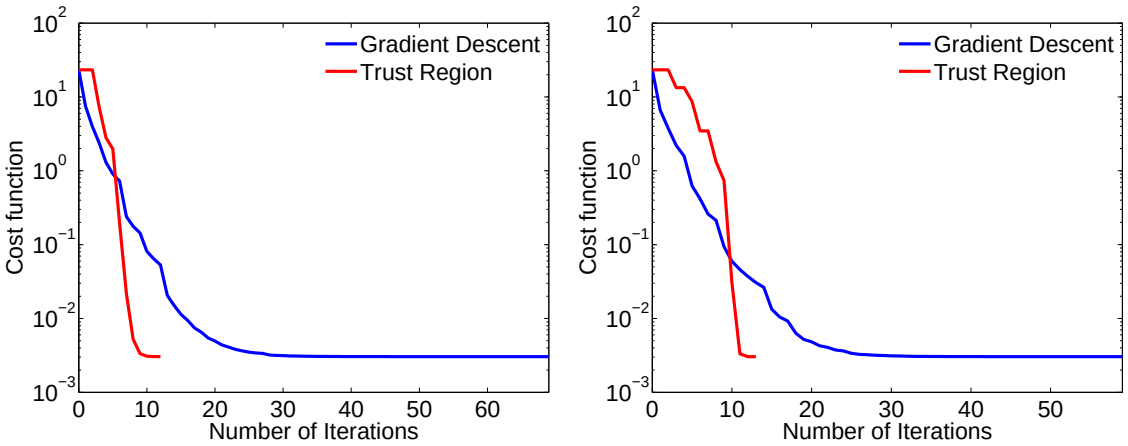
6.4.2 Low-rank matrix completion

We now perform experiments on non-symmetric matrix completion problems (Section 2.1.2). All algorithms presented in this section minimize the cost function (2.5).

Gradient descent versus trust-region

We first illustrate that trust-region algorithms require less iterations to converge than gradient descent algorithms. For this purpose, a data matrix $\mathbf{W}^\star \in \mathbb{R}^{1000\times 1000}$ of rank $r = 50$ is generated and 25% of its entries are selected uniformly and at random for learning. The resulting number of entries for learning is $|\Omega| = 250,000$. The matrix $\mathbf{W}^\star$ is generated with entries distributed according to a Gaussian distribution $\mathcal{N}(0, 1)$. Gaussian noise is added to the observed entries. The noise level is 10% of the standard deviation of the observed entries. All algorithms start from the rank-50 SVD of the matrix $P_\Omega(\mathbf{W}^\star)$. They all terminate when the cost function drops below 10^{-4} or when the relative improvement in the cost function drops below 10^{-5}.

The obtained results show that trust-region algorithms require less iterations to converge than gradient descent algorithms, both for the balanced (Figure 6.2(a)) and polar factorization (Figure 6.2(b)). Moreover, trust-region algorithms also require less time to converge. For the balanced factorization, the time required to achieve convergence is 21.06 seconds for the trust-region algorithm and 33.34 seconds for the gradient descent algorithm. For the polar factorization, the time required to achieve convergence is 28.79 seconds for the trust-region algorithm and 31.34 seconds for the gradient descent algorithm. The observed tendency is confirmed by the experiments in the next section on larger data sets.



(a) Balanced Factorization $\mathbf{W} = \mathbf{G}\mathbf{H}^T$. (b) Polar Factorization $\mathbf{W} = \mathbf{U}\mathbf{B}\mathbf{V}^T$.

Figure 6.2: Gradient descent versus trust-region on a toy random matrix completion problem. For both factorizations, trust-region algorithms require less iterations and less time than gradient descent algorithms to achieve the desired optimization precision.

Scaling test

We now evaluate the speed of the proposed trust-region algorithms on larger random data sets, and compare them to their batch gradient descent counterpart. Random matrices $\mathbf{W}^\star \in \mathbb{R}^{d\times d}$ of rank 10 and of various sizes d are generated with entries distributed according to a Gaussian distribution $\mathcal{N}(0,1)$. The time and number of iterations required to achieve a mean square error of 10^{-3} on the set of training samples is measured. For each problem, we sample 10% of the total amount of entries for learning and the algorithms are run with a fixed-rank $r = 10$. All algorithms start from the rank-10 SVD of the matrix $P_\Omega(\mathbf{W}^\star)$. Results are averaged over 5 runs. This test has been performed on a single core Intel L5420 2.5 GHz with 5GB of RAM.

The time taken by the algorithms to achieve convergence is reported in Figure 6.3(a). The number of iterations required to achieve convergence is reported at Figure 6.3(c) for the trust-region algorithms, and in Figure 6.3(d) for the gradient descent algorithms. Root mean square errors on the set of test samples are provided in Figure 6.3(b).

For this benchmark, trust-region algorithms run faster their gradient descent counterparts. This is illustrated in terms of the time required to achieve the desired precision (Figure 6.3(a)). When the problem size is $d = 10,000$, the number of known entries is about 10 millions. In that setting, the trust-region algorithm based on the balanced geometry converges in 3.05 minutes on average whereas the trust-region algorithm based on the polar geometry converges in 4.27 minutes on average. In contrast, the gradient descent algorithm based on the balanced geometry converges in 47.88 minutes on average whereas the gradient descent algorithm based on the polar geometry converges in 47.51 minutes on average.

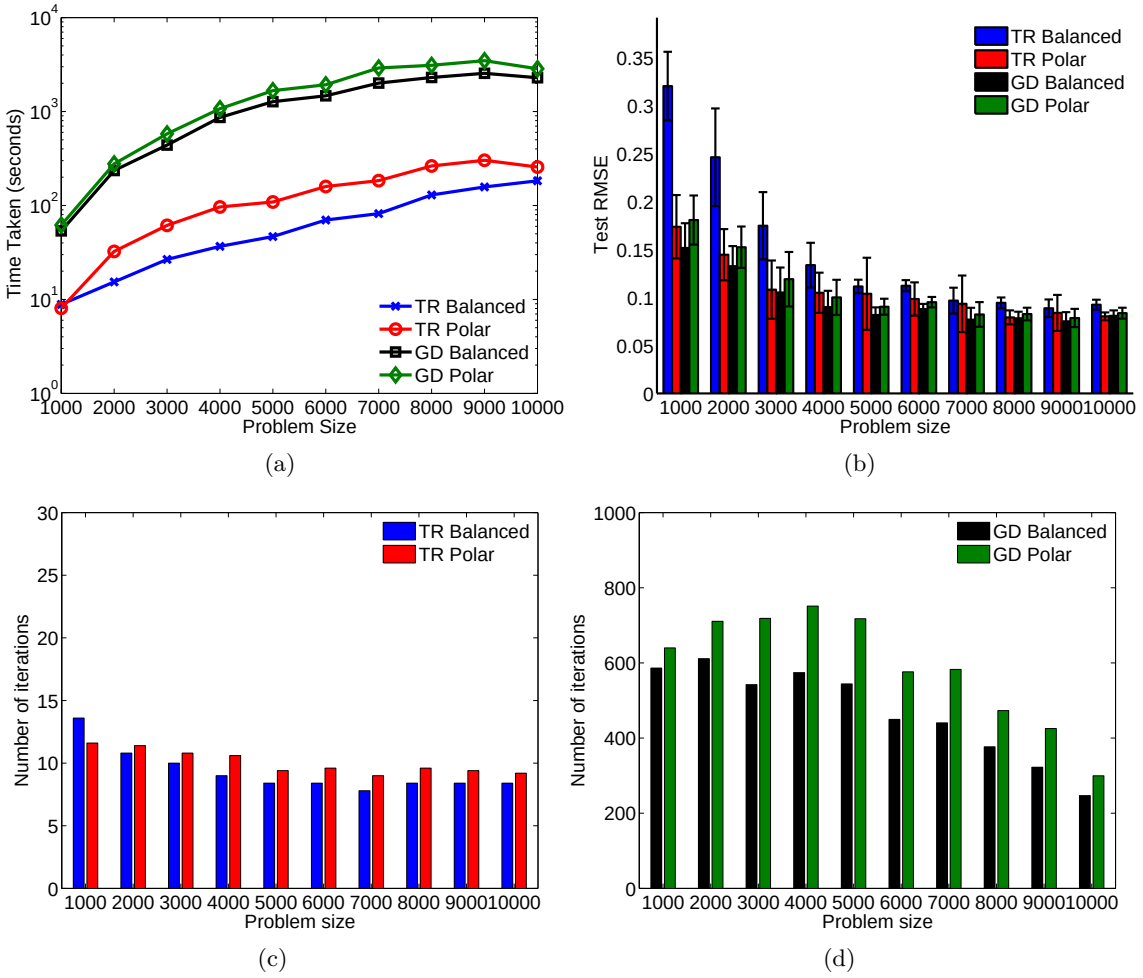


Figure 6.3: Scaling test for the trust-region algorithms on random matrix completion problems.

6.5 Conclusion

In this chapter, we equip the quotient manifold geometries developed in Chapters 4 and 5 with closed-form expressions for computing the Riemannian connections. The resulting formulas allow us to derive efficient trust-region algorithms for matrix and distance matrix completion.

We show that for these problems, the underlying sparse structure of the data leads to trust-region algorithms that maintain a linear complexity in the matrix dimension and in the number of available observations. The proposed algorithms come with a well-characterized convergence theory and enjoy a superlinear convergence rate.

Preliminary experiments on matrix completion problems demonstrate that the proposed trust-region algorithms scale well with the problem dimension and that they typically run faster than their batch gradient descent counterparts.

Chapter 7

Conclusion and perspectives for future research

In this thesis, we exploit the rich Riemannian geometry of the set of fixed-rank matrices as a quotient manifold to design efficient linear regression algorithms. We propose novel gradient descent and trust-region algorithms for learning a fixed-rank matrix. We further establish connections between the proposed algorithms and earlier algorithmic contributions in the literature on learning fixed-rank matrices. The established connections provide new geometric insights into previously proposed algorithms as well as perspectives for extension.

We thereby show that optimization on matrix manifolds is an effective and versatile optimization framework for the design of rank-constrained machine learning algorithms.

A first contribution of the thesis is to cast several modern machine learning applications as linear regression problems on the set of fixed-rank matrices. Problems such as Mahalanobis distance learning, kernel learning, low-rank matrix completion, learning on data pairs, multitask regression or ranking problems directly fit into the considered linear regression framework for particular choice of input data, observations and loss functions.

The main contribution of the thesis is to develop novel algorithms for linear regression on symmetric fixed-rank positive semidefinite matrices and to generalize these developments to fixed-rank non-symmetric matrices. The proposed algorithms rely on the rich quotient manifold geometry underlying fixed-rank matrix factorizations. They scale to high-dimensional problems, enjoy local convergence properties and preserve the geometric structure of the problem.

The potential of the proposed algorithms is illustrated on several benchmark experiments for which the proposed algorithms compete with the state-of-the-art.

As a specific contribution to the learning of fixed-rank symmetric positive semidefinite matrices, we generalize the results of Kulis et al. (2009) for learning matrices of a fixed range space. Indeed, our algorithms impose no restriction on the range space of the learned matrix and numerical experiments illustrate the benefits of simultaneously learning the subspace of the matrix and a positive definite linear operator within that subspace. The performance improvement over algorithms that fix the subspace of the learned matrix beforehand can be significant, especially in the case where the chosen rank is small compared to the original problem dimension.

The proposed algorithms for learning fixed-rank non-symmetric matrices rely on novel quotient geometries for the set of fixed-rank matrices. The proposed quotient geometries generalize previous works on analogous quotient geometries for fixed-rank symmetric positive semidefinite matrices (Bonnabel and Sepulchre, 2009; Journée et al., 2010). The proposed algorithms have appealing computational properties. In particular, using a balanced factorization allows us to ensure a good numerical conditioning of the algorithm. The algorithm based on the polar factorization allows us to perform regularization with minimal increase of the algorithm complexity.

Finally, we provide the material for the design of second-order rank-constrained optimization algorithms. Novel formulas for computing the Riemannian connection associated with the quotient manifold geometries considered in the thesis are proposed. We exploit those formula in the design of efficient trust-region algorithms for matrix completion problems. The proposed algorithms scale linearly in the problem size and enjoy a superlinear convergence rate.

Perspectives for future research

In this thesis, we have not investigated the probabilistic interpretation of the considered linear regression models and of the proposed algorithms. This is a topic left for future research that could be investigated in the light of recent work in this area (Ilin and Raiko, 2010).

A promising research direction pertaining to low-rank matrix factorizations is to investigate the interplay between geometric optimization methods based on low-rank matrix factorizations and convex relaxation approaches based on the nuclear norm. As the nuclear norm also enjoys a rich convex geometry, it could be exploited in combination with manifold based optimization techniques to develop custom solvers for efficient nuclear norm minimization, possibly yielding global convergence properties, faster convergence rates, while being memory-efficient.

An open question related to low-rank matrix factorizations concerns the choice of the best factorization for a given problem. It is tempting to believe that a given parametrization of the search space could be better suited to a given problem structure. Our intuition comes from the analogy with the polar coordinates system in the Euclidean space that provide dramatic simplifications for axisymmetric problems.

Due to the nonconvex nature of the fixed-rank constraint, the convergence results are only local and little can be presently said about the global convergence of the proposed algorithms. Global convergence properties are however not hopeless and could be facilitated by the considered low-rank factorizations. For instance, global convergence properties have been established for PCA algorithms from an explicit analysis of the critical points (Chen et al., 1998). Recent results also suggest good global convergence properties for low-rank matrix completion algorithms that rely on a good heuristic for the initialization (Keshavan et al., 2010; Jain et al., 2010). Experimental results obtained in this thesis suggest the same conclusions for the proposed algorithms, meaning that further research on global convergence results is certainly worthwhile.

Another promising research direction is the development of distributed and parallelized versions of the proposed algorithms. Indeed, with the increasingly growing size and number of large-scale problems, modern data sets no longer fit on a single computer and traditional machine learning algorithms often have prohibitively long running times. Existing approaches for distributed and parallel stochastic gradient optimization in $\mathbb{R}^d$ (Tsitsiklis et al., 1986; Zinkevich et al., 2010; Louppe and Geurts, 2010; Agarwal and Duchi, 2011) require technical adaptations to be generalized to manifolds. The recent developments in consensus and distributed algorithms on manifolds (Sarlette, 2009) as well as recent works on averaging over Riemannian manifolds (Arsigny et al., 2007; Bonnabel and Sepulchre, 2009) provide a good starting point.

Likewise, accelerated stochastic gradient techniques in $\mathbb{R}^d$ progressively gain popularity (Bordes et al., 2009; Yu et al., 2010), and their adaptation to manifolds is of topical interest.

Appendix A

Proofs

Convergence proof of algorithm (4.3)

Bottou (1998) reviews the mathematical tools required to prove almost sure convergence, that is asymptotic convergence with probability one, of stochastic gradient algorithms. Almost sure convergence follows from the following five assumptions:

(A1) $F(\mathbf{G}) = \mathbb{E}_{\mathbf{X},y}\{\ell(\hat{y}, y)\} \geq 0$ is three times differentiable with bounded derivatives,

(A2) the step sizes satisfy $\sum_{t=1}^{\infty} \eta_t^2 < \infty$ and $\sum_{t=1}^{\infty} \eta_t = \infty$,

(A3) $\mathbb{E}_{\mathbf{X},y}\{\|\text{grad} f(\mathbf{G})\|_F^2\} \leq k_1 + k_2\|\mathbf{G}\|_F^2$, where $f(\mathbf{G}) = \ell(\hat{y}, y)$,

(A4) $\exists h_1 > 0, \inf_{\|\mathbf{G}\|_F^2 > h_1} \text{Tr}(\mathbf{G}^T \mathbb{E}_{\mathbf{X},y}\{\text{grad} f(\mathbf{G})\}) > 0$,

(A5) $\exists h_2 > h_1, \forall (\mathbf{X}, y) \in \mathcal{X} \times \mathcal{Y}, \sup_{\|\mathbf{G}\|_F^2 < h_2} \|\text{grad} f(\mathbf{G})\|_F \leq k_3$.

Provided that algorithm (4.3) is equipped with an adaptive step size $s_t = \eta_t / \max(\|\mathbf{G}_t\|_F^2, 1)$, where the η_t satisfy (A2), we have the following convergence result.

Proposition *for bounded data* $(\mathbf{x}, y)$*, algorithm* (4.3) *equipped with the step size* s_t *defined above converges almost surely to the set of stationary points of the cost function*

$$F(\mathbf{G}) = \mathbb{E}_{\mathbf{X},y}\{f(\mathbf{G})\}, \quad f(\mathbf{G}) = \frac{1}{2}(\hat{y} - y)^2. \tag{A.1}$$

Proof. The proof is completed in two steps. First, it is shown that the stochastic sequence

$$u_t = \max(h_2, \|\mathbf{G}_t\|_F^2),$$

defines a Lyapunov process (always positive and decreasing on average) which is bounded almost surely by h_2. This implies that $\mathbf{G}_t$ is almost surely confined within distance $\sqrt{h_2}$ from the origin and provides almost sure bounds on all continuous functions of $\mathbf{G}_t$. In Bottou (1998), confinement is essentially based on (A3) and (A4). To ensure almost sure confinement of $\mathbf{G}_t$, we rely on the fact that $\mathbb{E}_{\mathbf{X},y}\{\|\text{grad} f(\mathbf{G}) / \max(\|\mathbf{G}\|_F^2, 1)\|_F^2\} \leq k_1 + k_2\|\mathbf{G}\|_F^2$.

Second, the Lyapunov process $v_t = F(\mathbf{G}_t) \geq 0$ is proved to converge almost surely. Convergence of $F(\mathbf{G}_t)$ is then used to show that $w_t = \text{grad } F(\mathbf{G}_t)$ tends to zero almost surely. Technical details are adapted from the paper of Bottou (1998). □

In practice, saddle points and local maxima are unstable solutions while convergence to asymptotic plateaus is excluded by (A4). As a result, almost sure convergence to a local minimum of the expected cost is obtained.

Proof of Proposition 5.3.1

Let $\gamma : t \mapsto (\mathbf{G}\mathbf{M}(t)^{-1}, \mathbf{H}\mathbf{M}(t)^T)$ with $\mathbf{M}(t) \in \mathrm{GL}(r)$ be a curve along the equivalence class $[(\mathbf{G}, \mathbf{H})]$ passing through $(\mathbf{G}, \mathbf{H})$ at $t = 0$. The derivative $\dot{\gamma}(t)$ evaluated in $t = 0$ gives us the expression of vertical vectors at $(\mathbf{G}, \mathbf{H})$,

$$\dot{\gamma}(t)|_{t=0} = (-\mathbf{G}\mathbf{\Lambda}, \mathbf{H}\mathbf{\Lambda}^T), \quad \mathbf{\Lambda} \in \mathbb{R}^{r\times r}.$$

Hence,

$$\mathcal{V}_{(\mathbf{G},\mathbf{H})}\mathcal{F}(r, d_1, d_2) = \{(-\mathbf{G}\mathbf{\Lambda}, \mathbf{H}\mathbf{\Lambda}^T) : \ \mathbf{\Lambda} \in \mathbb{R}^{r\times r}\} \tag{A.2}$$

Horizontal vectors $(\bar{\xi}_{\mathbf{G}}, \bar{\xi}_{\mathbf{H}})$ at $(\mathbf{G}, \mathbf{H})$ are canonically chosen to be orthogonal to vertical vectors according to the chosen metric (5.7). Horizontal vectors $(\bar{\xi}_{\mathbf{G}}, \bar{\xi}_{\mathbf{H}})$ must then satisfy the condition

$$\mathrm{Tr}((\mathbf{G}^T\mathbf{G})^{-1}\bar{\xi}_{\mathbf{G}}^T(-\mathbf{G}\mathbf{\Lambda})) + \mathrm{Tr}((\mathbf{H}^T\mathbf{H})^{-1}\bar{\xi}_{\mathbf{H}}^T(\mathbf{H}\mathbf{\Lambda}^T)) = 0, \quad \forall \mathbf{\Lambda} \in \mathbb{R}^{r\times r},$$

which holds if and only if

$$\bar{\xi}_{\mathbf{G}}^T\mathbf{G}(\mathbf{H}^T\mathbf{H}) = (\mathbf{G}^T\mathbf{G})\mathbf{H}^T\bar{\xi}_{\mathbf{H}}.$$

One readily checks that (5.9) satisfies (5.8). Furthermore, the number of constraints imposed by (5.8) matches the number of degrees of freedom in (5.8). We thus conclude that the two representations (5.9) and (5.8) are equivalent.

Proof of Proposition 5.3.2

Let $f : (\mathbb{R}_*^{d_1\times r} \times \mathbb{R}_*^{d_2\times r})/\mathrm{GL}(r) \to \mathbb{R}$ be an arbitrary smooth function, and define

$$\bar{f} \triangleq f \circ \pi : (\mathbb{R}_*^{d_1\times r} \times \mathbb{R}_*^{d_2\times r}) \to \mathbb{R}.$$

Consider the mapping

$$h : (\mathbf{G}, \mathbf{H}) \mapsto (\mathbf{G}\mathbf{M}^{-1}, \mathbf{H}\mathbf{M}^T),$$

where $\mathbf{M} \in \mathrm{GL}(r)$. Since $\pi(h(\mathbf{G}, \mathbf{H})) = \pi(\mathbf{G}, \mathbf{H})$ for all $(\mathbf{G}, \mathbf{H})$, we have

$$\bar{f}(h(\mathbf{G}, \mathbf{H})) = \bar{f}(\mathbf{G}, \mathbf{H}), \text{ for all } (\mathbf{G}, \mathbf{H}).$$

By taking the differential of both sides,

$$D\bar{f}(h(\mathbf{G}, \mathbf{H}))[Dh(\mathbf{G}, \mathbf{H})[(\bar{\xi}_{\mathbf{G}}, \bar{\xi}_{\mathbf{H}})]] = D\bar{f}(\mathbf{G}, \mathbf{H})[(\bar{\xi}_{\mathbf{G}}, \bar{\xi}_{\mathbf{H}})]. \tag{A.3}$$

By definition of $(\bar{\xi}_{\mathbf{G}}, \bar{\xi}_{\mathbf{H}})$, we have

$$D\bar{f}(\mathbf{G}, \mathbf{H})[(\bar{\xi}_{\mathbf{G}}, \bar{\xi}_{\mathbf{H}})] = Df(\pi(\mathbf{G}, \mathbf{H}))[\xi_{[(\mathbf{G},\mathbf{H})]}].$$

Moreover, we have

$$Dh(\mathbf{G}, \mathbf{H})[(\bar{\xi}_{\mathbf{G}}, \bar{\xi}_{\mathbf{H}})] = (\bar{\xi}_{\mathbf{G}}\mathbf{M}^{-1}, \bar{\xi}_{\mathbf{H}}\mathbf{M}^T).$$

Thus, (A.3) yields

$$D\bar{f}(\mathbf{G}\mathbf{M}^{-1}, \mathbf{H}\mathbf{M}^T)[(\bar{\xi}_{\mathbf{G}}\mathbf{M}^{-1}, \bar{\xi}_{\mathbf{H}}\mathbf{M}^T)] = Df(\pi(\mathbf{G}\mathbf{M}^{-1}, \mathbf{H}\mathbf{M}^T))[\xi_{[(\mathbf{G},\mathbf{H})]}].$$

Since this equality is valid for any smooth function f, it implies that

$$D\pi(\mathbf{G}\mathbf{M}^{-1}, \mathbf{H}\mathbf{M}^T)[(\bar{\xi}_{\mathbf{G}}\mathbf{M}^{-1}, \bar{\xi}_{\mathbf{H}}\mathbf{M}^T)] = \xi_{[(\mathbf{G},\mathbf{H})]}.$$

Finally, observe that $(\bar{\xi}_{\mathbf{G}}\mathbf{M}^{-1}, \bar{\xi}_{\mathbf{H}}\mathbf{M}^T)$ is a horizontal vector at $(\mathbf{G}\mathbf{M}^{-1}, \mathbf{H}\mathbf{M}^T)$, since

$$(\bar{\xi}_{\mathbf{G}}\mathbf{M}^{-1})^T\mathbf{G}\mathbf{M}^{-1}\mathbf{M}\mathbf{H}^T\mathbf{H}\mathbf{M}^T = \mathbf{M}^{-T}\mathbf{G}^T\mathbf{G}\mathbf{M}^{-1}\mathbf{M}\mathbf{H}^T(\bar{\xi}_{\mathbf{H}}\mathbf{M}^T),$$

and thus

$$\bar{\xi}_{\mathbf{G}}^T \mathbf{G}(\mathbf{H}^T\mathbf{H}) = (\mathbf{G}^T\mathbf{G})\mathbf{H}^T\bar{\xi}_{\mathbf{H}}.$$

Therefore, $(\bar{\xi}_{\mathbf{G}}\mathbf{M}^{-1}, \bar{\xi}_{\mathbf{H}}\mathbf{M}^T)$ is the unique horizontal lift of $\xi_{[(\mathbf{G},\mathbf{H})]}$ at $(\mathbf{G}\mathbf{M}^{-1}, \mathbf{H}\mathbf{M}^T)$, and

$$(\bar{\xi}_{\mathbf{G}\mathbf{M}^{-1}}, \bar{\xi}_{\mathbf{H}\mathbf{M}^T}) = (\bar{\xi}_{\mathbf{G}}\mathbf{M}^{-1}, \bar{\xi}_{\mathbf{H}}\mathbf{M}^T).$$

Using this result, we have, for all $\mathbf{M} \in \mathrm{GL}(r)$,

$$\bar{g}_{(\mathbf{G}\mathbf{M}^{-1},\mathbf{H}\mathbf{M}^T)}(\bar{\xi}_{(\mathbf{G}\mathbf{M}^{-1},\mathbf{H}\mathbf{M}^T)}, \bar{\zeta}_{(\mathbf{G}\mathbf{M}^{-1},\mathbf{H}\mathbf{M}^T)}) = \bar{g}_{(\mathbf{G},\mathbf{H})}(\bar{\xi}_{(\mathbf{G},\mathbf{H})}, \bar{\zeta}_{(\mathbf{G},\mathbf{H})}).$$

Indeed, for the previous equation, we have

$$\begin{aligned}
\text{LHS} &= \mathrm{Tr}((\mathbf{M}^{-T}\mathbf{G}^T\mathbf{G}\mathbf{M}^{-1})^{-1}(\bar{\xi}_{\mathbf{G}\mathbf{M}^{-1}})^T(\bar{\zeta}_{\mathbf{G}\mathbf{M}^{-1}})) + \mathrm{Tr}((\mathbf{M}\mathbf{H}^T\mathbf{H}\mathbf{M}^T)^{-1}(\bar{\xi}_{\mathbf{H}\mathbf{M}^T})^T(\bar{\zeta}_{\mathbf{H}\mathbf{M}^T})) \\
&= \mathrm{Tr}(\mathbf{M}(\mathbf{G}^T\mathbf{G})^{-1}\mathbf{M}^T\mathbf{M}^{-T}\bar{\xi}_{\mathbf{G}}^T\bar{\zeta}_{\mathbf{G}}\mathbf{M}^{-1}) + \mathrm{Tr}(\mathbf{M}^{-T}(\mathbf{H}^T\mathbf{H})^{-1}\mathbf{M}^{-1}\mathbf{M}\bar{\xi}_{\mathbf{H}}^T\bar{\zeta}_{\mathbf{H}}\mathbf{M}^T) \\
&= \mathrm{Tr}((\mathbf{G}^T\mathbf{G})^{-1}\bar{\xi}_{\mathbf{G}}^T\bar{\zeta}_{\mathbf{G}}) + \mathrm{Tr}((\mathbf{H}^T\mathbf{H})^{-1}\bar{\xi}_{\mathbf{H}}^T\bar{\zeta}_{\mathbf{H}}) \\
&= \text{RHS}.
\end{aligned}$$

Proof of Proposition 6.2.3

Any tangent vector $(\xi_{\mathbf{G}}, \xi_{\mathbf{H}}) \in \mathbb{R}^{d_1\times r} \times \mathbb{R}^{d_2\times r}$ in the total space admits a decomposition

$$(\xi_{\mathbf{G}}, \xi_{\mathbf{H}}) = P^{\mathcal{H}}_{(\mathbf{G},\mathbf{H})}(\xi_{\mathbf{G}}, \xi_{\mathbf{H}}) + P^{\mathcal{V}}_{(\mathbf{G},\mathbf{H})}(\xi_{\mathbf{G}}, \xi_{\mathbf{H}}),$$

where $P^{\mathcal{H}}(\cdot)_{(\mathbf{G},\mathbf{H})}$ is the sought projection onto the horizontal space at $(\mathbf{G}, \mathbf{H})$ and $P^{\mathcal{V}}_{(\mathbf{G},\mathbf{H})}(\cdot)$ extracts the component that lies in the vertical space. We thus have

$$P^{\mathcal{H}}_{(\mathbf{G},\mathbf{H})}(\xi_{\mathbf{G}}, \xi_{\mathbf{H}}) = (\xi_{\mathbf{G}} + \mathbf{G}\boldsymbol{\Lambda}, \xi_{\mathbf{H}} - \mathbf{H}\boldsymbol{\Lambda}^T).$$

The value of $\boldsymbol{\Lambda}$ can be determined by plugging this expression into condition (6.8) that is satisfied by any horizontal vectors at $(\mathbf{G}, \mathbf{H})$,

$$(\xi_{\mathbf{G}} + \mathbf{G}\boldsymbol{\Lambda})^T\mathbf{G}(\mathbf{H}^T\mathbf{H}) = (\mathbf{G}^T\mathbf{G})\mathbf{H}^T(\xi_{\mathbf{H}} - \mathbf{H}\boldsymbol{\Lambda}^T).$$

Rearranging the terms of the previous equation leads to the Sylvester equation

$$(\mathbf{H}^T\mathbf{H})(\mathbf{G}^T\mathbf{G})\boldsymbol{\Lambda} + \boldsymbol{\Lambda}(\mathbf{H}^T\mathbf{H})(\mathbf{G}^T\mathbf{G}) = \xi_{\mathbf{H}}^T\mathbf{H}(\mathbf{G}^T\mathbf{G}) - (\mathbf{H}^T\mathbf{H})\mathbf{G}^T\xi_{\mathbf{G}}.$$

A solution $\boldsymbol{\Lambda} \in \mathbb{R}^{r\times r}$ exists and is unique provided that $(\mathbf{H}^T\mathbf{H})(\mathbf{G}^T\mathbf{G})$ has no eigenvalue equal to zero. This is the case since $\det((\mathbf{H}^T\mathbf{H})(\mathbf{G}^T\mathbf{G})) = \det(\mathbf{H}^T\mathbf{H})\det(\mathbf{G}^T\mathbf{G}) > 0$.

Proof of Proposition 5.3.3

Let $\gamma : t \mapsto (\mathbf{U}\mathbf{O}(t), \mathbf{O}(t)^T\mathbf{B}\mathbf{O}(t), \mathbf{V}\mathbf{O}(t))$ with $\mathbf{O}(t) \in \mathcal{O}(r)$ be a curve along the equivalence class $[(\mathbf{U}, \mathbf{B}, \mathbf{V})]$ passing through $(\mathbf{U}, \mathbf{B}, \mathbf{V})$ at $t = 0$. The set of vertical vectors at point $(\mathbf{U}, \mathbf{B}, \mathbf{V})$ is given by the derivative $\dot{\gamma}(t)$ evaluated in $t = 0$,

$$\dot{\gamma}(t)|_{t=0} = (\mathbf{U}\boldsymbol{\Omega}, \mathbf{B}\boldsymbol{\Omega} - \boldsymbol{\Omega}\mathbf{B}, \mathbf{V}\boldsymbol{\Omega}), \quad \boldsymbol{\Omega}^T = -\boldsymbol{\Omega} \in \mathbb{R}^{r\times r}.$$

Hence,

$$\mathcal{V}_{(\mathbf{U},\mathbf{B},\mathbf{V})}\mathcal{F}(r, d_1, d_2) = \{(\mathbf{U}\boldsymbol{\Omega}, \mathbf{B}\boldsymbol{\Omega} - \boldsymbol{\Omega}\mathbf{B}, \mathbf{V}\boldsymbol{\Omega}) : \ \boldsymbol{\Omega}^T = -\boldsymbol{\Omega} \in \mathbb{R}^{r\times r}\}.$$

Horizontal vectors $(\bar{\xi}_{\mathbf{U}}, \bar{\xi}_{\mathbf{B}}, \bar{\xi}_{\mathbf{V}})$ are then defined by the condition

$$\mathrm{Tr}(\bar{\xi}_{\mathbf{U}}^T(\mathbf{U}\boldsymbol{\Omega})) + \mathrm{Tr}(\mathbf{B}^{-1}\bar{\xi}_{\mathbf{B}}\mathbf{B}^{-1}(\mathbf{B}\boldsymbol{\Omega} - \boldsymbol{\Omega}\mathbf{B})) + \mathrm{Tr}(\bar{\xi}_{\mathbf{V}}^T(\mathbf{V}\boldsymbol{\Omega})) = 0, \quad \forall\, \boldsymbol{\Omega}^T = -\boldsymbol{\Omega} \in \mathbb{R}^{r\times r},$$

which holds if and only if

$$\bar{\xi}_{\mathbf{U}}^T\mathbf{U} + \mathbf{B}^{-1}\bar{\xi}_{\mathbf{B}} - \bar{\xi}_{\mathbf{B}}\mathbf{B}^{-1} + \bar{\xi}_{\mathbf{V}}^T\mathbf{V} = \mathbf{U}^T\bar{\xi}_{\mathbf{U}} + \bar{\xi}_{\mathbf{B}}\mathbf{B}^{-1} - \mathbf{B}^{-1}\bar{\xi}_{\mathbf{B}} + \mathbf{V}^T\bar{\xi}_{\mathbf{V}}. \tag{A.4}$$

Plugging the expressions of generic tangent vectors

$$\begin{aligned}\bar{\xi}_{\mathbf{U}} &= \mathbf{U}\mathrm{Skew}(\mathbf{A}) + \mathbf{U}_\perp \in T_{\mathbf{U}}\mathrm{St}(r, d_1)\\ \bar{\xi}_{\mathbf{B}} &= \mathrm{Sym}(\mathbf{D}) \qquad\qquad\quad \in T_{\mathbf{B}}S_{++}(r)\\ \bar{\xi}_{\mathbf{V}} &= \mathbf{V}\mathrm{Skew}(\mathbf{C}) + \mathbf{V}_\perp \in T_{\mathbf{V}}\mathrm{St}(r, d_2)\end{aligned}$$

into the previous equation yields

$$2(\mathbf{B}^{-1}\bar{\xi}_{\mathbf{B}} - \bar{\xi}_{\mathbf{B}}\mathbf{B}^{-1}) = 2(\mathrm{Skew}(\mathbf{A}) + \mathrm{Skew}(\mathbf{C})),$$

which can be rewritten as

$$\mathbf{B}(\mathrm{Skew}(\mathbf{A}) + \mathrm{Skew}(\mathbf{C}))\mathbf{B} = \bar{\xi}_{\mathbf{B}}\mathbf{B} - \mathbf{B}\bar{\xi}_{\mathbf{B}}.$$

Proof of Proposition 5.3.4

The proof is analogous to the proof of Proposition 5.3.2. The mapping h is now

$$h : (\mathbf{U}, \mathbf{B}, \mathbf{V}) \mapsto (\mathbf{UO}, \mathbf{O}^T\mathbf{BO}, \mathbf{VO}), \quad \mathbf{O} \in \mathcal{O}(r).$$

Observe that a horizontal vector $(\bar{\xi}_{\mathbf{UO}}, \bar{\xi}_{\mathbf{O}^T\mathbf{BO}}, \bar{\xi}_{\mathbf{VO}})$ at $(\mathbf{UO}, \mathbf{O}^T\mathbf{BO}, \mathbf{VO})$ defined as

$$(\bar{\xi}_{\mathbf{UO}}, \bar{\xi}_{\mathbf{O}^T\mathbf{BO}}, \bar{\xi}_{\mathbf{VO}}) = (\bar{\xi}_{\mathbf{U}}\mathbf{O}, \mathbf{O}^T\bar{\xi}_{\mathbf{B}}\mathbf{O}, \bar{\xi}_{\mathbf{V}}\mathbf{O}),$$

satisfies (A.4) at $(\mathbf{UO}, \mathbf{O}^T\mathbf{BO}, \mathbf{VO})$, which means that $(\bar{\xi}_{\mathbf{U}}\mathbf{O}, \mathbf{O}^T\bar{\xi}_{\mathbf{B}}\mathbf{O}, \bar{\xi}_{\mathbf{V}}\mathbf{O})$ is a horizontal vector at $(\mathbf{UO}, \mathbf{O}^T\mathbf{BO}, \mathbf{VO})$. Finally, one readily checks that $\forall \mathbf{O} \in \mathcal{O}(r)$,

$$\bar{g}_{(\mathbf{UO},\mathbf{O}^T\mathbf{BO},\mathbf{VO})}(\bar{\xi}_{(\mathbf{UO},\mathbf{O}^T\mathbf{BO},\mathbf{VO})}, \bar{\zeta}_{(\mathbf{UO},\mathbf{O}^T\mathbf{BO},\mathbf{VO})}) = \bar{g}_{(\mathbf{U},\mathbf{B},\mathbf{V})}(\bar{\xi}_{(\mathbf{U},\mathbf{B},\mathbf{V})}, \bar{\zeta}_{(\mathbf{U},\mathbf{B},\mathbf{V})}).$$

Proof of Proposition 6.2.4

Any given $(\xi_{\mathbf{U}}, \xi_{\mathbf{B}}, \xi_{\mathbf{V}}) \in T_{(\mathbf{U},\mathbf{B},\mathbf{V})}(\mathrm{St}(r, d_1) \times S_{++}(r) \times \mathrm{St}(r, d_2))$ admits a unique decomposition

$$(\xi_{\mathbf{U}}, \xi_{\mathbf{B}}, \xi_{\mathbf{V}}) = P^{\mathcal{V}}_{(\mathbf{U},\mathbf{B},\mathbf{V})}(\xi_{\mathbf{U}}, \xi_{\mathbf{B}}, \xi_{\mathbf{V}}) + P^{\mathcal{H}}_{(\mathbf{U},\mathbf{B},\mathbf{V})}(\xi_{\mathbf{U}}, \xi_{\mathbf{B}}, \xi_{\mathbf{V}}),$$

where $P^{\mathcal{V}}(\cdot)$ extracts the component that lies in the vertical space. We have

$$P^{\mathcal{H}}_{(\mathbf{U},\mathbf{B},\mathbf{V})}(\xi_{\mathbf{U}}, \xi_{\mathbf{B}}, \xi_{\mathbf{V}}) = (\xi_{\mathbf{U}} - \mathbf{U}\boldsymbol{\Omega}, \xi_{\mathbf{B}} - \mathbf{B}\boldsymbol{\Omega} + \boldsymbol{\Omega}\mathbf{B}, \xi_{\mathbf{V}} - \mathbf{V}\boldsymbol{\Omega}).$$

Substituting this expression into (A.4), one arrives at (6.11). A solution $\boldsymbol{\Omega}$ exists and is unique since $\mathbf{B}^2 \succ 0$ and $-\mathbf{B}^2 \prec 0$ have no common eigenvalues.

Proof of Theorem 5.4.1

Let us define $\mathbf{G}(\alpha) = \mathbf{G}\exp(-\alpha\mathbf{T})$, $\mathbf{H}(\alpha) = \mathbf{H}\exp(\alpha\mathbf{T})$ and $\mathbf{T} = \mathbf{G}^T\mathbf{G} - \mathbf{H}^T\mathbf{H}$, and consider the Lyapunov function

$$\varphi(\alpha) \triangleq \varphi(\mathbf{G}(\alpha), \mathbf{H}(\alpha)) = \mathrm{Tr}(\mathbf{G}(\alpha)^T\mathbf{G}(\alpha) + \mathbf{H}(\alpha)^T\mathbf{H}(\alpha)).$$

We now study the variation

$$\begin{aligned}
\Delta\varphi(\alpha) &\triangleq \varphi(\alpha) - \varphi(0) \\
&= \mathrm{Tr}(\mathbf{G}^T\mathbf{G}(\exp(-2\alpha\mathbf{T}) - \mathbf{I}) + \mathbf{H}^T\mathbf{H}(\exp(2\alpha\mathbf{T}) - \mathbf{I})) \\
&= \sum_{j=1}^{\infty} \mathrm{Tr}\left(\frac{(2\alpha)^j}{j!}(\mathbf{G}^T\mathbf{G} + (-1)^j\mathbf{H}^T\mathbf{H})(-\mathbf{T})^j\right) \\
&= \sum_{j=1}^{\infty}\left\{\frac{(2\alpha)^{2j-1}}{(2j-1)!}\mathrm{Tr}(-(\mathbf{G}^T\mathbf{G} - \mathbf{H}^T\mathbf{H})\mathbf{T}^{2j-1}) + \frac{(2\alpha)^{2j}}{(2j)!}\mathrm{Tr}((\mathbf{G}^T\mathbf{G} + \mathbf{H}^T\mathbf{H})\mathbf{T}^{2j})\right\} \\
&= \sum_{j=1}^{\infty} \mathrm{Tr}\left((\mathbf{G}^T\mathbf{G} + \mathbf{H}^T\mathbf{H} - \frac{2j}{2\alpha}\mathbf{I})\frac{(2\alpha\mathbf{T})^{2j}}{2j!}\right) \\
&\leq \sum_{j=1}^{\infty} \mathrm{Tr}\left((\mathbf{G}^T\mathbf{G} + \mathbf{H}^T\mathbf{H} - \frac{1}{\alpha}\mathbf{I})\right)\frac{(2\alpha\mathbf{T})^{2j}}{2j!}.
\end{aligned}$$

With the choice α of (5.16), we have $\Delta\varphi(\alpha) < 0$ and therefore $\varphi(\mathbf{G}_{t+1}, \mathbf{H}_{t+1}) < \varphi(\mathbf{G}_t, \mathbf{H}_t)$. Convergence follows from the classical Lyapunov stability theorem. Moreover, a fixed point $(\mathbf{G}_t, \mathbf{H}_t)$ satisfying $\varphi(\mathbf{G}_{t+1}, \mathbf{H}_{t+1}) = \varphi(\mathbf{G}_t, \mathbf{H}_t)$ is obtained if and only if

$$\sum_{j=1}^{\infty} \mathrm{Tr}\left(\frac{(2\alpha_t)^{2j}}{(2j)!}(\mathbf{G}_t^T\mathbf{G}_t - \mathbf{H}_t^T\mathbf{H}_t)^{2j}\right) = 0$$

The latter condition is equivalent to $\mathbf{G}_t^T\mathbf{G}_t = \mathbf{H}_t^T\mathbf{H}_t$ since $\alpha_t > 0$.

Proof of Proposition 6.2.4

Expression of horizontal vectors

Let $\gamma : t \mapsto (\mathbf{U}\mathbf{O}(t), \mathbf{O}(t)^T\mathbf{R}^2\mathbf{O}(t))$ with $\mathbf{O}(t) \in \mathcal{O}(r)$ be a curve along the equivalence class $[(\mathbf{U}, \mathbf{R}^2)]$ passing through $(\mathbf{U}, \mathbf{R}^2)$ at $t = 0$. The set of vertical vectors at point $(\mathbf{U}, \mathbf{R}^2)$ is given by the derivative $\dot\gamma(t)$ evaluated in $t = 0$,

$$\dot\gamma(t)|_{t=0} = (\mathbf{U}\mathbf{\Omega}, \mathbf{R}^2\mathbf{\Omega} - \mathbf{\Omega}\mathbf{R}^2), \quad \mathbf{\Omega}^T = -\mathbf{\Omega} \in \mathbb{R}^{r\times r}.$$

Hence,

$$\mathcal{V}_{(\mathbf{U},\mathbf{R}^2)}(\mathrm{St}(r,d) \times S_{++}(r))/\mathcal{O}(r) = \{(\mathbf{U}\mathbf{\Omega}, \mathbf{R}^2\mathbf{\Omega} - \mathbf{\Omega}\mathbf{R}^2) : \ \mathbf{\Omega}^T = -\mathbf{\Omega} \in \mathbb{R}^{r\times r}\}.$$

Horizontal vectors $(\bar\xi_{\mathbf{U}}, \bar\xi_{\mathbf{R}^2})$ are then defined by the condition

$$\mathrm{Tr}(\bar\xi_{\mathbf{U}}^T(\mathbf{U}\mathbf{\Omega})) + \mathrm{Tr}(\mathbf{R}^{-2}\bar\xi_{\mathbf{R}^2}\mathbf{R}^{-2}(\mathbf{R}^2\mathbf{\Omega} - \mathbf{\Omega}\mathbf{R}^2)) = 0, \quad \forall\, \mathbf{\Omega}^T = -\mathbf{\Omega} \in \mathbb{R}^{r\times r},$$

which holds if and only if

$$\bar\xi_{\mathbf{U}}^T\mathbf{U} + \mathbf{R}^{-2}\bar\xi_{\mathbf{R}^2} - \bar\xi_{\mathbf{R}^2}\mathbf{R}^{-2} = \mathbf{U}^T\bar\xi_{\mathbf{U}} + \bar\xi_{\mathbf{R}^2}\mathbf{R}^{-2} - \mathbf{R}^{-2}\bar\xi_{\mathbf{R}^2}. \tag{A.5}$$

Plugging the expressions of generic tangent vectors

$$\begin{aligned}
\bar\xi_{\mathbf{U}} &= \mathbf{U}\mathrm{Skew}(\mathbf{A}) + \mathbf{U}_\perp &&\in T_{\mathbf{U}}\mathrm{St}(r,d_1) \\
\bar\xi_{\mathbf{R}^2} &= \mathrm{Sym}(\mathbf{D}) &&\in T_{\mathbf{R}^2}S_{++}(r)
\end{aligned}$$

into the previous equation yields

$$2(\mathbf{R}^{-2}\bar\xi_{\mathbf{R}^2} - \bar\xi_{\mathbf{R}^2}\mathbf{R}^{-2}) = 2\mathrm{Skew}(\mathbf{A}),$$

which can be rewritten as

$$\mathrm{Skew}(\mathbf{A}) = \mathbf{R}^{-2}\bar\xi_{\mathbf{R}^2} - \bar\xi_{\mathbf{R}^2}\mathbf{R}^{-2}.$$

Invariance of the metric along the fibers

The proof is analogous to the proof of Proposition 5.3.2. The mapping h is now

$$h : (\mathbf{U}, \mathbf{R}^2) \mapsto (\mathbf{UO}, \mathbf{O}^T\mathbf{R}^2\mathbf{O}), \quad \mathbf{O} \in \mathcal{O}(r).$$

Observe that a horizontal vector $(\bar{\xi}_{\mathbf{UO}}, \bar{\xi}_{\mathbf{O}^T\mathbf{R}^2\mathbf{O}})$ at $(\mathbf{UO}, \mathbf{O}^T\mathbf{R}^2\mathbf{O})$ defined as

$$(\bar{\xi}_{\mathbf{UO}}, \bar{\xi}_{\mathbf{O}^T\mathbf{R}^2\mathbf{O}}) = (\bar{\xi}_{\mathbf{U}}\mathbf{O}, \mathbf{O}^T\bar{\xi}_{\mathbf{R}^2}\mathbf{O}),$$

satisfies (A.5) at $(\mathbf{UO}, \mathbf{O}^T\mathbf{R}^2\mathbf{O})$, which means that $(\bar{\xi}_{\mathbf{U}}\mathbf{O}, \mathbf{O}^T\bar{\xi}_{\mathbf{R}^2}\mathbf{O})$ is a horizontal vector at $(\mathbf{UO}, \mathbf{O}^T\mathbf{R}^2\mathbf{O})$. Finally, one readily checks that $\forall \mathbf{O} \in \mathcal{O}(r)$,

$$\bar{g}_{(\mathbf{UO},\mathbf{O}^T\mathbf{R}^2\mathbf{O})}(\bar{\xi}_{(\mathbf{UO},\mathbf{O}^T\mathbf{R}^2\mathbf{O})}, \bar{\zeta}_{(\mathbf{UO},\mathbf{O}^T\mathbf{R}^2\mathbf{O})}) = \bar{g}_{(\mathbf{U},\mathbf{R}^2)}(\bar{\xi}_{(\mathbf{U},\mathbf{R}^2)}, \bar{\zeta}_{(\mathbf{U},\mathbf{R}^2)}).$$

Appendix B

Omitted derivations

Riemannian connection on $S_{++}(r)$ for the affine-invariant metric

The Riemannian connection $\overline{\nabla}_\xi \zeta$ on $S_{++}(r)$ associated with the affine-invariant metric

$$g_{\mathbf{R}^2}(\xi_{\mathbf{R}^2}, \zeta_{\mathbf{R}^2}) = \mathrm{Tr}(\mathbf{R}^{-2}\xi_{\mathbf{R}^2}\mathbf{R}^{-2}\zeta_{\mathbf{R}^2}),$$

is obtained from Koszul formula (3.26). For all vector fields ξ, ζ, ν on $T_{\mathbf{R}^2}S_{++}(r)$, we must have

$$\begin{aligned} 2\bar{g}_{\mathbf{R}^2}(\overline{\nabla}_\xi \zeta, \nu) = D\bar{g}_{\mathbf{R}^2}(\zeta, \nu)[\xi] + D\bar{g}_{\mathbf{R}^2}(\xi, \nu)[\zeta] - D\bar{g}_{\mathbf{R}^2}(\zeta, \xi)[\nu] \\ + \bar{g}_{\mathbf{R}^2}(\nu, [\xi, \zeta]) + \bar{g}_{\mathbf{R}^2}(\zeta, [\nu, \xi]) - \bar{g}_{\mathbf{R}^2}(\xi, [\zeta, \nu]). \end{aligned} \tag{B.1}$$

Since $S_{++}(r)$ is an open subset of a vector space, the Lie bracket is given by

$$[\xi, \zeta] = D\zeta[\xi] - D\xi[\zeta]. \tag{B.2}$$

Moreover, we have

$$\begin{aligned} D\bar{g}_{\mathbf{R}^2}(\zeta, \nu)[\xi] = \bar{g}_{\mathbf{R}^2}(D\zeta[\xi], \nu) + \bar{g}_{\mathbf{R}^2}(\zeta, D\nu[\xi]) \\ - \mathrm{Tr}(\mathbf{R}^{-2}\xi\mathbf{R}^{-2}\zeta\mathbf{R}^{-2}\nu) - \mathrm{Tr}(\mathbf{R}^{-2}\zeta\mathbf{R}^{-2}\xi\mathbf{R}^{-2}\nu). \end{aligned} \tag{B.3}$$

Plugging identities (B.2) and (B.3) into (B.1) yields,

$$2\bar{g}_{\mathbf{R}^2}(\overline{\nabla}_\xi \zeta, \nu) = 2\bar{g}_{\mathbf{R}^2}(D\zeta[\xi], \nu) - \mathrm{Tr}(\mathbf{R}^{-2}\xi\mathbf{R}^{-2}\zeta\mathbf{R}^{-2}\nu) - \mathrm{Tr}(\mathbf{R}^{-2}\zeta\mathbf{R}^{-2}\xi\mathbf{R}^{-2}\nu).$$

Since this identity is valid for all vector fields ν, we identify

$$\overline{\nabla}_\xi \zeta = D\zeta[\xi] - \mathrm{Sym}(\xi\mathbf{R}^{-2}\zeta). \tag{B.4}$$

Riemannian connection in $\mathbb{R}_*^{d_1\times r} \times \mathbb{R}_*^{d_2\times r}$ for the balanced geometry

We derive the Riemannian connection $\overline{\nabla}_\xi \zeta$ in $\mathbb{R}_*^{d_1\times r} \times \mathbb{R}_*^{d_2\times r}$ for the Riemannian metric

$$\bar{g}_{(\mathbf{G},\mathbf{H})}((\bar{\xi}_\mathbf{G}, \bar{\xi}_\mathbf{H}), (\bar{\zeta}_\mathbf{G}, \bar{\zeta}_\mathbf{H})) = \mathrm{Tr}((\mathbf{G}^T\mathbf{G})^{-1}\bar{\xi}_\mathbf{G}^T\bar{\zeta}_\mathbf{G}) + \mathrm{Tr}((\mathbf{H}^T\mathbf{H})^{-1}\bar{\xi}_\mathbf{H}^T\bar{\zeta}_\mathbf{H}). \tag{B.5}$$

For all vector fields $\xi, \zeta, \nu \in (\mathbb{R}^{d_1\times r} \times \mathbb{R}^{d_2\times r})$, the Riemannian connection $\overline{\nabla}_\xi \zeta$ must satisfy Kozsul formula

$$\begin{aligned} 2\bar{g}_{(\mathbf{G},\mathbf{H})}(\overline{\nabla}_\xi \zeta, \nu) = D\bar{g}_{(\mathbf{G},\mathbf{H})}(\zeta, \nu)[\xi] + D\bar{g}_{(\mathbf{G},\mathbf{H})}(\xi, \nu)[\zeta] - D\bar{g}_{(\mathbf{G},\mathbf{H})}(\zeta, \xi)[\nu] \\ + \bar{g}_{(\mathbf{G},\mathbf{H})}(\nu, [\xi, \zeta]) + \bar{g}_{(\mathbf{G},\mathbf{H})}(\zeta, [\nu, \xi]) - \bar{g}_{(\mathbf{G},\mathbf{H})}(\xi, [\zeta, \nu]). \end{aligned} \tag{B.6}$$

Since $\mathbb{R}_*^{d_1\times r}\times\mathbb{R}_*^{d_2\times r}$ is an open subset of a vector space, the Lie bracket is given by

$$[\xi,\zeta]=D\zeta[\xi]-D\xi[\zeta], \tag{B.7}$$

Moreover, we have

$$\begin{aligned}&D\bar{g}_{(\mathbf{G},\mathbf{H})}((\zeta_{\mathbf{G}},\zeta_{\mathbf{H}}),(\nu_{\mathbf{G}},\nu_{\mathbf{H}}))[\xi_{\mathbf{G}},\xi_{\mathbf{H}}]= \\ &\quad\bar{g}_{(\mathbf{G},\mathbf{H})}(D(\zeta_{\mathbf{G}},\zeta_{\mathbf{H}})[\xi_{\mathbf{G}},\xi_{\mathbf{H}}],(\nu_{\mathbf{G}},\nu_{\mathbf{H}}))+\bar{g}_{(\mathbf{G},\mathbf{H})}((\zeta_{\mathbf{G}},\zeta_{\mathbf{H}}),D(\nu_{\mathbf{G}},\nu_{\mathbf{H}})[\xi_{\mathbf{G}},\xi_{\mathbf{H}}]) \\ &\quad-2\mathrm{Tr}((\mathbf{G}^T\mathbf{G})^{-1}\mathrm{Sym}(\xi_{\mathbf{G}}^T\mathbf{G})(\mathbf{G}^T\mathbf{G})^{-1}\zeta_{\mathbf{G}}^T\nu_{\mathbf{G}})-2\mathrm{Tr}((\mathbf{H}^T\mathbf{H})^{-1}\mathrm{Sym}(\xi_{\mathbf{H}}^T\mathbf{H})(\mathbf{H}^T\mathbf{H})^{-1}\zeta_{\mathbf{H}}^T\nu_{\mathbf{H}}).\end{aligned} \tag{B.8}$$

Plugging identities (B.7) and (B.8) into (B.6) yields

$$\begin{aligned}\overline{\nabla}_{(\bar{\xi}_{\mathbf{G}},\bar{\xi}_{\mathbf{H}})}\bar{\zeta}_{\mathbf{G}}&=D\bar{\zeta}_{\mathbf{G}}[\bar{\xi}_{\mathbf{G}},\bar{\xi}_{\mathbf{H}}]-\bar{\zeta}_{\mathbf{G}}(\mathbf{G}^T\mathbf{G})^{-1}\mathrm{Sym}(\bar{\xi}_{\mathbf{G}}^T\mathbf{G})-\bar{\xi}_{\mathbf{G}}(\mathbf{G}^T\mathbf{G})^{-1}\mathrm{Sym}(\bar{\zeta}_{\mathbf{G}}^T\mathbf{G}) \\ &\quad+\mathbf{G}(\mathbf{G}^T\mathbf{G})^{-1}\mathrm{Sym}(\bar{\zeta}_{\mathbf{G}}^T\bar{\xi}_{\mathbf{G}}), \\ \overline{\nabla}_{(\bar{\xi}_{\mathbf{G}},\bar{\xi}_{\mathbf{H}})}\bar{\zeta}_{\mathbf{H}}&=D\bar{\zeta}_{\mathbf{H}}[\bar{\xi}_{\mathbf{G}},\bar{\xi}_{\mathbf{H}}]-\bar{\zeta}_{\mathbf{H}}(\mathbf{H}^T\mathbf{H})^{-1}\mathrm{Sym}(\bar{\xi}_{\mathbf{H}}^T\mathbf{H})-\bar{\xi}_{\mathbf{H}}(\mathbf{H}^T\mathbf{H})^{-1}\mathrm{Sym}(\bar{\zeta}_{\mathbf{H}}^T\mathbf{H}) \\ &\quad+\mathbf{H}(\mathbf{H}^T\mathbf{H})^{-1}\mathrm{Sym}(\bar{\zeta}_{\mathbf{H}}^T\bar{\xi}_{\mathbf{H}}).\end{aligned}$$

Bibliography

J. Abernethy, F. Bach, T. Evgeniou, and J.-P. Vert. A new approach to collaborative filtering: Operator estimation with spectral regularization. *Journal of Machine Learning Research*, 10 (Mar):803–826, 2009.

P.-A. Absil. *Invariant Subspace Computation: A Geometric Approach*. PhD thesis, University of Liège, 2003.

P.-A. Absil, R. Mahony, and R. Sepulchre. Riemannian geometry of Grassmann manifolds with a view on algorithmic computation. *Acta Appl. Math.*, 80(2):199–220, 2004.

P.-A. Absil, C. G. Baker, and Kyle A. Gallivan. Trust-region methods on Riemannian manifolds. *Foundations of Computational Mathematics*, 7(3):303–330, 2007.

P.-A. Absil, R. Mahony, and R. Sepulchre. *Optimization Algorithms on Matrix Manifolds*. Princeton University Press, 2008.

P.-A. Absil, M. Ishteva, L. De Lathauwer, and S. Van Huffel. A geometric Newton method for Oja's vector field. *Neural Computation*, 21(5):1415–1433, 2009.

A. Agarwal and J. Duchi. Distributed delayed stochastic optimization. Technical report, arXiv:1104.5525v1, 2011.

Y. Amit, M. Fink, N. Srebro, and S. Ullman. Uncovering shared structures in multiclass classification. In *Proceedings of the 24th International Conference on Machine Learning*, 2007.

A. Argyriou, T. Evgeniou, and M. Pontil. Convex multi-task feature learning. *Machine Learning*, 73(3):243–272, 2007.

R. Arora. On learning rotations. In Y. Bengio, D. Schuurmans, J. Lafferty, C. K. I. Williams, and A. Culotta, editors, *Advances in Neural Information Processing Systems*, volume 22, pages 55–63. MIT Press, 2009.

V. Arsigny, P. Fillard, X. Pennec, and N. Ayache. Geometric means in a novel vector space structure on symmetric positive-definite matrices. *SIAM Journal on Matrix Analysis and Applications*, 29(1):328–347, 2007.

A. Asuncion and D.J. Newman. UCI machine learning repository. `http://www.ics.uci.edu/~mlearn/MLRepository.html`, 2007. University of California, Irvine, School of Information and Computer Sciences.

F. Bach. Consistency of trace norm minimization. *Journal of Machine Learning Research*, 9: 1019–1048, 2008.

F. Bach and M. I. Jordan. Predictive low-rank decomposition for kernel methods. In *Proceedings of the 22nd International Conference on Machine Learning (ICML)*, 2005.

L. Balzano, R. Nowak, and B. Recht. Online identification and tracking of subspaces from highly incomplete information. *arXiv:1006.4046v1*, June 2010.

H.H. Bauschke and J.M. Borwein. Legendre functions and the method of random bregman projections. *Journal of Convex Analysis*, 4:27–67, 1997.

K. Bleakley and Y. Yamanishi. Supervised prediction of drug-target interactions using bipartite local models. *Bioinformatics*, 25(18):2397–2403, 2009.

S. Bonnabel. Convergence of stochastic gradient descent algorithms on Riemannian manifolds. Technical report, Mines ParisTech, 2011.

S. Bonnabel and R. Sepulchre. Riemannian metric and geometric mean for positive semidefinite matrices of fixed rank. *SIAM Journal on Matrix Analysis and Applications*, 31(3):1055–1070, 2009.

S. Bonnabel, G. Meyer, and R. Sepulchre. Adaptive filtering for estimation of a low-rank positive semidefinite matrix. In *Proceedings of the 19th International Symposium on Mathematical Theory of Networks and Systems*, 2010.

W. M. Boothby. *An Introduction to Differentiable Manifolds and Riemannian Geometry, second edition*. Academic Press, 1986.

A. Bordes, L. Bottou, and P. Gallinari. Sgd-qn: Careful quasi-Newton stochastic gradient descent. *Journal of Machine Learning Research*, 10:1737–1754, July 2009.

I. Borg and P. Groenen. *Modern Multidimensional Scaling: Theory and Applications*. Springer, second edition, 2005.

L. Bottou. Online algorithms and stochastic approximations. In David Saad, editor, *Online Learning and Neural Networks*. Cambridge University Press, 1998.

L. Bottou. Stochastic learning. In *Advanced Lectures on Machine Learning*, number 3176 in Lecture Notes in Artificial Intelligence, LNAI-3176, pages 146–168. Springer Verlag, 2004.

L. Bottou and O. Bousquet. The tradeoffs of large scale learning. In J.C. Platt, D. Koller, Y. Singer, and S. Roweis, editors, *Advances in Neural Information Processing Systems*, volume 20, pages 161–168. MIT Press, 2007.

S. Boyd and L. Vandenberghe. *Convex Optimization*. Cambridge University Press, 2004.

S. Boyd, L. El Ghaoui, E. Feron, and V. Balakrishnan. *Linear Matrix Inequalities in Systems and Control Theory*. Volume 15 of Studies in Applied Mathematics. Society for Industrial and Applied Mathematics, 1994.

R. W. Brockett. System theory on group manifolds and coset spaces. *SIAM J. Control*, 10(2): 265–284, 1972.

R. W. Brockett. Differential geometry and the design of gradient algorithms. In Amer. Math. Soc., editor, *Proceedings of Symposia in Pure Mathematics*, volume 54, pages 69–92, 1993.

S. Burer and R. Monteiro. A nonlinear programming algorithm for solving semidefinite programs via low-rank factorization. *Mathematical Programming*, 95(2):329–357, 2003.

D. Cai, X. He, and J. Han. Efficient kernel discriminant analysis via spectral regression. In *Proceedings of the International Conference on Data Mining (ICDM07)*, 2007.

J.-F. Cai, E. J. Candes, and Z. Shen. A singular value thresholding algorithm for matrix completion. *SIAM Journal on Optimization*, 20(4):1956–1982, 2008.

E. J. Candès and B. Recht. Exact matrix completion via convex optimization. *Foundations of Computational Mathematics*, 9:717–772, 2008.

E. J. Candès and M. Wakin. An introduction to compressive sampling. *IEEE Signal Processing Magazine*, pages 21–30, 2008.

R. Chatpatanasiri, T. Korsrilabutr, P. Tangchanachaianan, and B. Kijsirikul. A new kernelization framework for Mahalanobis distance learning algorithms. *Neurocomputing*, 73(10-12): 1570–1579, 2010.

G. Chechik, V. Sharma, U. Shalit, and S. Bengio. Large scale online learning of image similarity through ranking. *Journal of Machine Learning Research*, 11(Mar), 2010.

T. Chen, Y. Hua, and W.-Y. Yan. Global convergence of Oja's subspace algorithm for principal component extraction. *IEEE Transaction Neural Networks*, 9(1):58–67, 1998.

T. F. Cox and M.A.A. Cox. *Multidimensional Scaling*. Chapman and Hall, 2001.

K. Crammer. Online tracking of linear subspaces. In *Proceedings of 19th Annual Conference on Learning Theory (COLT)*, 2006.

M. Cucuringu, Y. Lipman, and A. Singer. Sensor network localization by eigenvector synchronization over the euclidean group. *ACM Transactions on Sensor Networks (accepted)*, 2011.

J. Daniel, W.B. Gragg, L. Kaufman, and G. W. Stewart. Reorthogonalization and stable algorithms for updating the Gram-Schmidt QR factorization. *Math. Comp.*, 30:772–795, 1976.

A. d'Aspremont, L. El Ghaoui, M. I. Jordan, and G. Lanckriet. A direct formulation for sparse pca using semidefinite programming. *SIAM Review*, 49(3):434–448, 2007.

J. V. Davis and I. S. Dhillon. Structured metric learning for high dimensional problems. In *Proceedings of the 14th ACM SIGKDD conference on Knowledge Discovery and Data Mining*, 2008.

J. V. Davis, B. Kulis, P. Jain, S. Sra, and I. S. Dhillon. Information-theoretic metric learning. In *Proceedings of the 24th International Conference on Machine Learning (ICML)*, 2007.

J. P. Delmas and J. F. Cardoso. Performance analysis of an adaptive algorithm for tracking dominant subspaces. *IEEE Transactions on Signal Processing*, 46(111):3045–3057, 1998.

I. S. Dhillon and J.A. Tropp. Matrix nearness problems with Bregman divergences. *SIAM Journal on Matrix Analysis and Applications*, 29(4):1120–1146, 2007.

I. S. Dhillon, Y. Guan, and B. Kulis. Weighted graph cuts without eigenvectors: A multilevel approach. *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, 19(11): 1944–1957, 2007.

A. Edelman, T.A. Arias, and S.T. Smith. The geometry of algorithms with orthogonality constraints. *SIAM Journal on Matrix Analysis and Applications*, 20(2):303–353, 1998.

T. Evgeniou, C.A. Micchelli, and M. Pontil. Learning multiple tasks with kernel methods. *Journal of Machine Learning Research*, 6(Apr):615–637, 2005.

H. Fang and D. P. Oleary. Euclidean distance matrix completion problems. Technical report, University of Minnesota, 2010.

J. Faraut and A. Koranyi. *Analysis on Symmetric Cones*. Oxford University Press, 1994.

M. Fazel. *Matrix rank minimization with applications*. PhD thesis, Stanford University, 2002.

S. Fine, K. Scheinberg, N. Cristianini, J. Shawe-Taylor, and B. Williamson. Efficient SVM training using low-rank kernel representations. *Journal of Machine Learning Research*, 2: 243–264, 2001.

Y. Freund and R. Schapire. A decision-theoretic generalization of on-line learning and an application to boosting. *Journal of Computer and System Sciences*, 55(1):119–139, 1997.

M. Friedlander and M. Schmidt. Hybrid deterministic-stochastic methods for data fitting. Technical report, arXiv:1104.2373v1, 2011.

C.F. Gauss. Theoria motus corporum coelestium in sectionibus conicis solem ambientum. 1809.

A. Globerson and S. Roweis. Metric learning by collapsing classes. In Y. Weiss, B. Schölkopf, and J. Platt, editors, *Advances in Neural Information Processing Systems*, volume 18, pages 451–458. MIT Press, 2005.

J. Goldberger, S. Roweis, G. Hinton, and R. Salakhutdinov. Neighbourhood components analysis. In L. K. Saul, Y. Weiss, and L. Bottou, editors, *Advances in Neural Information Processing Systems*, volume 17, pages 513–520. MIT Press, 2004.

G. H. Golub and C. F. Van Loan. *Matrix Computations*. The Johns Hopkins University Press, 1996.

D. Gross. Recovering low-rank matrices from few coefficients in any basis. *IEEE Transaction on Information Theory*, 57(3):1548–1566, 2011.

T. Hastie, R. Tibshirani, and J. Friedman. *The Elements of Statistical Learning: Data Mining, Inference, and Prediction*. Springer, 2009.

U. Helmke and J. Moore. *Optimization and Dynamical Systems*. Springer, 1996.

A. Ilin and T. Raiko. Practical approaches to principal component analysis in the presence of missing values. *Journal of Machine Learning Research*, 11(Jul):1957–2000, 2010.

M. Ishteva, P.-A. Absil, S. Van Huffel, and L. De Lathauwer. Best low multilinear rank approximation of higher-order tensors, based on the riemannian trust-region scheme best low multilinear rank approximation of higher-order tensors, based on the riemannian trust-region scheme. *SIAM Journal on Matrix Analysis and Applications*, 32(1):115–135, 2011.

P. Jain, B. Kulis, I. S. Dhillon, and K. Grauman. Online metric learning and fast similarity search. In D. Koller, D. Schuurmans, Y. Bengio, and L. Bottou, editors, *Advances in Neural Information Processing Systems*, volume 21, pages 761–768. MIT Press, 2008.

P. Jain, R. Meka, and I. Dhillon. Guaranteed rank minimization via singular value projection. In J. Lafferty, C. K. I. Williams, J. Shawe-Taylor, R.S. Zemel, and A. Culotta, editors, *Advances in Neural Information Processing Systems 23*, pages 937–945. 2010.

M. Journée. *Geometric Algorithms for Component Analysis with a view to gene-expression analysis*. PhD thesis, University of Liège, 2009.

M. Journée, F. Bach, P.-A. Absil, and R. Sepulchre. Low-rank optimization on the cone of positive semidefinite matrices. *SIAM Journal on Optimization*, 20(5):2327–2351, 2010.

A.J. Kearsley, R.A. Tapia, and M.W. Trosset. The solution of the metric stress and sstress problems in multidimensional scaling by Newton's method. *Computational Statistics*, 13(3): 369–396, 1998.

R. H. Keshavan and A. Montanari. Regularization for matrix completion. In *Proc. of the IEEE Int. Symposium on Inform. Theory*, 2010.

R. H. Keshavan, A. Montanari, and S. Oh. Matrix completion from noisy entries. *Journal of Machine Learning Research*, 11(Jul):2057–2078, 2010.

J. Kiefer and J. Wolfowitz. Stochastic estimation of the maximum of a regression function. *Annals of Mathematical Statistics*, 23(3):462–466, 1952.

J. Kivinen and M.K. Warmuth. Exponentiated gradient versus gradient descent for linear predictors. *Journal of Information and Computation*, 132(1):1–64, January 1997.

J. Kivinen and M.K. Warmuth. Boosting as entropy projection. In *roceedings of the 12th Conference on Computational Learning Theory (COLT)*, 1999.

B. Kulis, M. Sustik, and I. S. Dhillon. Low-rank kernel learning with Bregman matrix divergences. *Journal of Machine Learning Research*, 10:341–376, 2009.

B. Kulis, K. Saenko, and T. Darrell. What you saw is not what you get: Domain adaptation using asymmetric kernel transforms. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2011.

J. Kwok and I. Tsang. Learning with idealized kernels. *Proceedings of the 20th International Conference on Machine learning (ICML)*, 2003.

G. Lanckriet, N. Cristianini, P. Bartlett, L. El Ghaoui, and M.I. Jordan. Learning the kernel matrix with semidefinite programming. *Journal of Machine Learning Research*, 5:27–72, 2004.

Y. LeCun, B. Boser, J. S. Denker, D. Henderson, R. E. Howard, W. Hubbard, and L. D. Jackel. Handwritten digit recognition with a back-propagation network. In David Touretzky, editor, *Advances in Neural Information Processing Systems*, volume 2. Morgan Kaufman, 1989.

John M. Lee. *Introduction to smooth manifolds*, volume 218 of *Graduate Texts in Mathematics*. Springer-Verlag, New York, 2003.

Kiryung Lee and Yoram Bresler. Admira: Atomic decomposition for minimum rank approximation. *arXiv:0905.0044v2*, May 2009.

A.M. Legendre. Nouvelles méthodes pour la détermination des orbites des comètes. 1805.

G. Louppe and P. Geurts. A zealous parallel gradient descent algorithm. In *Learning on Cores, Clusters and Clouds workshop at NIPS, Vancouver, Canada*, 2010.

D. G. Luenberger. The gradient projection method along geodesics. *Management Science*, 18 (11):620–631, 1972.

S. Ma, D. Goldfarb, and L. Chen. Fixed point and bregman iterative methods for matrix rank minimiza- tion. *To appear, Mathematical Programming Series A*, 2010.

Y. Ma, J. Kosecka, and S. Sastry. Optimization criteria and geometric algorithms for motion and structure estimation. *International Journal of Computer Vision*, 44(3):219–249, 2001.

P. C. Mahalanobis. On the generalised distance in statistics. In *Proceedings of the National Institute of Sciences of India*, volume 2, pages 49–55, 1936.

Y. Matsuda and K. Yamaguchi. Global mapping analysis: stochastic gradient algorithm in SSTRESS and classical MDS stress. In *Proceedings of International Conference on Neural Information Processing*, 2001.

R. Mazumder, T. Hastie, and R. Tibshirani. Spectral regularization algorithms for learning large incomplete matrices. *Journal of Machine Learning Research*, 11(Aug):2287–2322, 2010.

Raghu Meka, Prateek Jain, and Inderjit S Dhillon. Matrix completion from power-law distributed samples. In Y. Bengio, D. Schuurmans, J. Lafferty, C. K. I. Williams, and A. Culotta, editors, *Advances in Neural Information Processing Systems 22*, pages 1258–1266. 2009.

R. Merris. Laplacian matrices of graphs : a survey. *Linear Algebra and its Applications*, 197-198: 143–176, 1994.

G. Meyer, S. Bonnabel, and R. Sepulchre. From subspace learning to distance learning: a geometrical optimization approach. In *Proceedings of the IEEE Workshop on Statistical Signal Processing (SSP2009)*, 2009.

G. Meyer, S. Bonnabel, and R. Sepulchre. Regression on fixed-rank positive semidefinite matrices: a Riemannian approach. *Journal of Machine Learning Research*, 11(Feb):593–625, 2011a.

G. Meyer, S. Bonnabel, and R. Sepulchre. Linear regression under fixed-rank constraints: a Riemannian approach. In *Proceedings of the 28th International Conference on Machine Learning*, 2011b.

G. Meyer, S. Bonnabel, and R. Sepulchre. Linear regression on fixed-rank non-symmetric: a Riemannian approach. Technical report, University of Liège, 2011c.

B. Mishra, G. Meyer, and R. Sepulchre. Low-rank optimization for distance matrix completion. In *Submitted to the 50th IEEE Conference on Decision and Control*, 2011a.

B. Mishra, G. Meyer, and R. Sepulchre. Low-rank optimization for large-scale non-symmetric problems. Technical report, University of Liège, 2011b.

Y. Nesterov. Random gradient-free minimization of convex function. Technical report, CORE Discussion Paper (Université Catholique de Louvain), 2011.

Netflix. The Netflix prize. http://www.netflixprize.com/, 2006.

Yasunori Nishimori and Shotaro Akaho. Learning algorithms utilizing quasi-geodesic flows on the stiefel manifold. *Neurocomputing*, 67:106–135, 2005.

J. Nocedal and S. J. Wright. *Numerical Optimization, Second Edition*. Springer, 2006.

E. Oja. Principal components, minor components, and linear neural networks. *Neural Networks*, 5:927 – 935, 1992.

R. Orsi, U. Helmke, and J. B. Moore. A newton-like method for solving rank constrained linear matrix inequalities. *Automatica Journal IFAC*, 42:1875–1882, 2006.

K. Pearson, G. U. Yule, N. Blanchard, and A. Lee. The law of ancestral heredity. *Biometrika*, 2(2):211–236, 1903.

E.F. Petricoin, D.K. Ornstein, C.P. Paweletz, A.M. Ardekani, P.S. Hackett, B.A. Hitt, A. Velassco, C. Trucco, L. Wiegand, K. Wood, C.B. Simone, P.J. Levine, W.M. Linehan, M.R. Emmert-Buck, S.M. Steinberg, E.C Kohn, and L.A. Liotta. Serum proteomic patterns for detection of prostate cancer. *Journal of the National Cancer Institute*, 94(20):1576–1578, 2002.

B. Recht, M. Fazel, and P. A. Parrilo. Guaranteed minimum-rank solutions to linear matrix equations via nuclear norm minimization. *SIAM Review*, 52(3):471–501, 2010.

J. Rennie and N. Srebro. Fast maximum margin matrix factorization for collaborative prediction. In *Proceedings of the 22nd International Conference on Machine learning*, pages 713–719, 2005.

H. Robbins and S. Monro. A stochastic approximation method. *Annals of Mathematical Statistics*, 22(3):400–407, 1951.

A. Sarlette. *Geometry and Symmetries in Coordination Control*. PhD thesis, University of Liège, Belgium, 2009.

R. Schapire and Y. Singer. Improved boosting algorithms using confidence-rated predictions. *Machine Learning*, 37:297–336, 1999.

B. Schölkopf, A. J. Smola, and K.-R. Müller. Nonlinear component analysis as a kernel eigenvalue problem. *Neural Computation*, 10(3):1299–1319, 1998.

P.H. Schonemann. A generalized solution of the orthogonal procrustes problem. *Psychometrika*, 31:1–10, 1966.

S. Shalev-Shwartz, Y. Singer, and A. Ng. Online and batch learning of pseudo-metrics. In *Proceedings of the 21st International Conference on Machine Learning (ICML)*, 2004.

U. Shalit, D. Weinshall, and G. Chechik. Online learning in the manifold of low-rank matrices. In J. Lafferty, C. K. I. Williams, J. Shawe-Taylor, R.S. Zemel, and A. Culotta, editors, *Advances in Neural Information Processing Systems 23*, pages 2128–2136. 2010.

J. Shawe-Taylor and N. Cristianini. *Kernel Methods for Pattern Analysis*. Cambridge University Press, 2004.

L. Simonsson and L. Eldén. Grassmann algorithms for low rank approximation of matrices with missing values. *BIT Numerical Mathematics*, 50(1):173–191, 2010.

S.T. Smith. *Geometric Optimization Methods for Adaptive Filtering*. PhD thesis, Harvard University, Cambridge, MA, 1993.

S.T. Smith. Covariance, subspace, and intrinsic Cramér-Rao bounds. *IEEE Transactions on Signal Processing*, 53(5):1610–1630, 2005.

L. Song, A. Smola, K. Borgwardt, and A. Gretton. Colored maximum variance unfolding. In J.C. Platt, D. Koller, Y. Singer, and S. Roweis, editors, *Advances in Neural Information Processing Systems*, volume 20, pages 1385–1392. MIT Press, 2007.

A. Strehl, J. Ghosh, and R. Mooney. Impact of similarity measures on web-page clustering. In *Workshop on Artificial Intelligence for Web Search (AAAI)*, 2000.

P. Tarazaga and M. W. Trosset. An optimization problem on subsets of the symmetric positive-semidefinite matrices. *Journal of Optimization Theory and Applications*, 79:513–524, 1993.

R. Tibshirani. Regression shrinkage and selection via the lasso. *Journal of the Royal Statistical Society*, 58(1):267–288, 1996.

K.C. Toh and S. Yun. An accelerated proximal gradient algorithm for nuclear norm regularized least squares problems. *Pacific Journal of Optimization*, 6:615–640, 2010.

L. Torresani and K. Lee. Large margin component analysis. In B. Schölkopf, J. Platt, and T. Hoffman, editors, *Advances in Neural Information Processing Systems*, volume 19, pages 1385–1392. MIT Press, 2006.

J. Tsitsiklis, D. Bertsekas, and M. Athans. Distributed asynchronous deterministic and stochastic gradient optimization algorithms. *IEEE Transactions on Automatic Control*, 31(9):803–812, 1986.

K. Tsuda, G. Ratsch, and M. Warmuth. Matrix exponentiated gradient updates for on-line learning and Bregman projection. *Journal of Machine Learning Research*, 6:995–1018, 2005.

B. Vandereycken. Low-rank matrix completion by Riemannian optimization. Technical report, ANCHP-MATHICSE, Mathematics Section, Ecole Polytechnique Fédérale de Lausanne, 2011.

B. Vandereycken and S. Vandewalle. A Riemannian optimization approach for computing low-rank solutions of Lyapunov equations. *SIAM Journal on Matrix Analysis and Applications*, 31(5):2553–2579, 2010.

B. Vandereycken, P.-A. Absil, and S. Vandewalle. A Riemannian geometry with complete geodesics for the set of positive semidefinite matrices of fixed rank. Technical report, Katholieke Universiteit Leuven, 2011.

M. Warmuth. Winnowing subspaces. *Proceedings of the 24th international conference on Machine learning (ICML)*, 2007.

K. Weinberger and L. Saul. Distance metric learning for large margin nearest neighbor classification. *Journal of Machine Learning Research*, 10(Feb):207–244, 2009.

K. Weinberger, F. Sha, and L. Saul. Learning a kernel matrix for nonlinear dimensionality reduction. *Proceedings of the 21st International Conference on Machine Learning (ICML)*, pages 839–846, 2004.

E. P. Xing, A. Y. Ng, M. I. Jordan, and S. Russell. Distance metric learning with application to clustering with side-information. In S. Becker, S. Thrun, and K. Obermayer, editors, *Advances in Neural Information Processing Systems*, volume 15, pages 505–512. MIT Press, 2002.

Y. Yamanishi, M. Araki, A. Gutteridge, W. Honda, and M. Kanehisa. Prediction of drug-target interaction networks from the integration of chemical and genomic spaces. *Bioinformatics*, 24 (13):i232, 2008.

L. Yang. Distance metric learning: a comprehensive survey. Technical report, Michigan State University, 2006.

J. Yu, S. Vishwanathan, S. Günter, and N. Schraudolph. A quasi-Newton approach to nonsmooth convex optimization problems in machine learning. *Journal of Machine Learning Research*, 11:1145–1200, 2010.

G. U. Yule. On the theory of correlation. *Journal of the Royal Statistical Society*, 60(4), 1897.

J. Zhuang, I. Tsang, and S. Hoi. SimpleNPKL: simple non-parametric kernel learning. *Proceedings of the 26th International Conference on Machine Learning (ICML)*, 2009.

M. Zinkevich, M. Weimer, A. Smola, and L. Li. Parallelized stochastic gradient descent. In J. Lafferty, C. K. I. Williams, J. Shawe-Taylor, R.S. Zemel, and A. Culotta, editors, *Advances in Neural Information Processing Systems 23*, 2010.