Journal of Statistical Software

July 2012, Volume 50, Issue 5. *http://www.jstatsoft.org/*

GrassmannOptim: An R Package for Grassmann Manifold Optimization

Kofi Placid Adragni
University of Maryland

R. Dennis Cook
University of Minnesota

Seongho Wu
Fannie Mae

Abstract

The optimization of a real-valued objective function $f(\mathbf{U})$, where $\mathbf{U}$ is a $p \times d, p > d$, semi-orthogonal matrix such that $\mathbf{U}^\top\mathbf{U} = \mathbf{I}_d$, and f is invariant under right orthogonal transformation of $\mathbf{U}$, is often referred to as a Grassmann manifold optimization. Manifold optimization appears in a wide variety of computational problems in the applied sciences. In this article, we present **GrassmannOptim**, an R package for Grassmann manifold optimization. The implementation uses gradient-based algorithms and embeds a stochastic gradient method for global search. We describe the algorithms, provide some illustrative examples on the relevance of manifold optimization and finally, show some practical usages of the package.

Keywords: Grassmann manifold, constrained optimization, simulated annealing.

1. Introduction

This article presents **GrassmannOptim**, an R (R Development Core Team 2012) package for orthogonally constrained optimization. Let $\mathbf{U}$ be a $p \times d$, $d < p$, semi-orthogonal matrix, $\mathbf{U}^\top\mathbf{U} = \mathbf{I}_d$. The package aims to maximize a real-valued function $f(\mathbf{U})$ that is invariant under right orthogonal transformations of $\mathbf{U}$: for any $d \times d$ orthogonal matrix $\mathbf{O}$, $f(\mathbf{U}) = f(\mathbf{UO})$. This invariance of f means that the argument of f can be any representative orthonormal basis of the subspace spanned by the d columns of $\mathbf{U}$. Thus, the optimization of f is performed over the set of all d-dimensional linear subspaces of $\mathbb{R}^p$. This set, denoted here by $\mathcal{G}_{(d,p)}$, is called a Grassmann manifold or Grassmannian.

The set of all one-dimensional subspaces of $\mathbb{R}^2$, $\mathcal{G}_{(1,2)}$, is simply the set of all lines passing through the origin, the set of all two-dimensional subspaces of $\mathbb{R}^3$, $\mathcal{G}_{(2,3)}$ can be thought of as the set of all planes, and the collection of all d-dimensional subspaces of $\mathbb{R}^p$, $\mathcal{G}_{(d,p)}$, can be similarly viewed as the set of all d-dimensional hyperplanes. A single subspace in $\mathcal{G}_{(d,p)}$ can

be described uniquely by $d(p-d)$ real angles, one angle being required to pick a line in $\mathbb{R}^2$ and two angles being required to uniquely specify a plane in $\mathbb{R}^3$. The function $f(\mathbf{U})$ could conceivably be parameterized and maximized in terms of these $d(p-d)$ angles. However, using the geometry of a Grassmann manifold allows one to move continuously and smoothly through the subspaces and to develop efficient algorithms based on derivatives on the manifold.

For example, consider approximating a collection of n centered p dimensional vectors $\mathbf{X}_1$, $\ldots$,$\mathbf{X}_n$, $\sum_{i=1}^n \mathbf{X}_i = 0$, with a corresponding collection of vectors $\mathbf{Z}_i$ that lie in a d-dimensional subspace of $\mathbb{R}^p$. The approximating vectors can be represented in the form $\mathbf{Z}_i = \mathbf{U}\mathbf{C}_i$, where $\mathbf{U}$ is a semi-orthogonal basis matrix for the d-dimensional subspace and $\mathbf{C}_i$ holds the coordinates of $\mathbf{Z}_i$ in terms of $\mathbf{U}$. Using the Euclidean norm we wish to minimize $\sum_{i=1}^n \|\mathbf{X}_i - \mathbf{U}\mathbf{C}_i\|^2$. This norm is minimized over $\mathbf{C}_i$ with $\mathbf{U}$ fixed by the value $\mathbf{C}_i = \mathbf{B}^\top \mathbf{X}_i$ and thus we need to minimize $\sum_{i=1}^n \|\mathbf{X}_i - \mathbf{U}\mathbf{U}^\top \mathbf{X}_i\|^2$; equivalently, we need to maximize $f(\mathbf{U}) = \mathrm{Tr}(\mathbf{U}\mathbf{U}^\top \widehat{\mathbf{\Sigma}})$, where $\widehat{\mathbf{\Sigma}}$ is the sample covariance matrix of the $\mathbf{X}_i$'s. This $f(\mathbf{U})$ is invariant under right orthogonal transformations of $\mathbf{U}$ and so we have reduced the original problem to maximizing over a Grassmann manifold. In this case it is known that $f(\mathbf{U})$ is maximized by choosing the column of $\mathbf{U}$ to be the first d eigenvectors of $\widehat{\mathbf{\Sigma}}$. However, in more complicated problems an analytic solution will not be available and numerical optimization becomes necessary.

Many different manifolds are found in the literature. While several are unnamed, the often encountered are *oblique manifold, Riemannian manifold, Stiefel manifold* and *Grassmann manifold*. Defining algorithms on manifolds requires endowing these manifolds with a differentiable structure so that the computation of a gradient or the evaluation of an objective function is meaningful. An element of a given Grassmann manifold is a subspace. The differential geometry of Grassmann manifolds has been well studied (Wong 1967; Edelman, Arias, and Smith 1998; Chikuse 2003; Absil, Mahony, and Sepulchre 2004), and computations are often carried out using matrices to represent corresponding subspaces. A subspace can be specified non-uniquely by a basis. It also can be specified uniquely by a projection matrix. In this article, we adopt the representation by a basis of the subspace.

Grassmann manifolds and Stiefel manifolds (set of all $p \times d$ orthonormal matrices in $\mathbb{R}^p$) have been used in practice in various fields. Gruber and Theis (2006) proposed a generalized concept of k-means on a Grassmann manifold. Nilsson, Sha, and Jordan (2007) studied the problem of discovering a manifold that best preserves information relevant to a non-linear regression. The problem of computing eigenspaces and eigenvalues that is ubiquitous in engineering can be identified as manifold optimization.

Grassmann manifolds are used to accommodate orthogonality constraints. These constraints often add a layer of challenge to the optimization compared to unconstrained problems. Methods abound in the literature to deal with optimization problems with linear constraints (Fletcher 1987; Nocedal and Wright 1999), and the objective function to optimize is often modified to allow unconstrained optimization. To our knowledge, techniques related to the Lagrange multiplier and the Kuhn-Tucker theorem, simulated annealing and sequential procedures are often used in the search for orthonormal bases in constrained optimizations. However, using the intrinsic geometry of Grassmann manifold allows for efficiency. Liu, Srivastava, and Gallivan (2004) proposed a gradient algorithm that guided the development of **GrassmannOptim**. It was originally developed for image analysis to find the optimal linear representations of images for use in appearance-based object recognition. But the algorithm finds applications in other research fields.

Without extra care, a Grassmann optimization method may converge to a local optimizer and this may not be the global optimizer. To illustrate this, let $(\mathbf{U}, \mathbf{V})$ be an orthogonal matrix and consider the problem of maximizing $f(\mathbf{U}) = \mathrm{Tr}(\mathbf{U}\mathbf{U}^\top \widehat{\mathbf{\Sigma}})$ mentioned previously. The derivative on the manifold is the ordinary unconstrained derivative multiplied by $\mathbf{V}$ (See Section 2): $(\nabla f(\mathbf{U}))^\top \mathbf{V} = 2\mathbf{U}^\top \widehat{\mathbf{\Sigma}}\mathbf{V}$ which is 0 when the columns of $\mathbf{U}$ are any set of d eigenvectors of $\widehat{\mathbf{\Sigma}}$. The function $f(\mathbf{U})$ then has many stationary points which leads to the problem of local optimizers. The geometry of Grassmann manifolds allows for Markov chain Monte Carlo type algorithms for optimization. Although Liu *et al.* (2004) proposed a stochastic gradient method to help locate the global maximization, we found a simpler version proposed by Gallivan, Srivastava, Liu, and Dooren (2003), which is available as an option in **GrassmannOptim**. As a gradient-based algorithm, it does not make use of the Hessian of the objective function that is typical to Newton-type algorithms. But the gradient is required, although its explicit analytical expression is optional.

Other algorithms are found in the literature. Edelman *et al.* (1998) published a seminal manuscript on Stiefel and Grassmann manifold optimization of real-valued functions, where Newton-type and conjugate gradient algorithms were developed. Their algorithms perform a series of steps along a geodesic. Manton (2002) proposed algorithms similar to those of Edelman *et al.* (1998), with complex-valued orthogonality constraints. He broke from moving along geodesics and reformulated the constrained optimization problem as an unconstrained one on a suitable manifold. To our knowledge, there is no publicly available R routine for Grassmann manifold optimization. However, an existing package, called **sg_min**, is available for MATLAB (The MathWorks, Inc. 2010). It was written by Lippert (2004) and was adapted from Edelman *et al.* (1998).

The remainder of this article is organized as follows. In Section 2, we give a brief description of the algorithms proposed by Gallivan *et al.* (2003) and Liu *et al.* (2004) that guided the programming of **GrassmannOptim**. In Section 3, we present three examples of problems where manifold optimization can be used. Section 4 presents the usage of the package through some examples. We present an application to a dataset in Section 5 and a discussion follows in Section 6. The package is available from the Comprehensive R Archive Network at `http://CRAN.R-project.org/package=GrassmannOptim`.

2. The algorithm

2.1. The basic gradient algorithm

Let $\mathbf{U}$ be a $p \times d$ semi-orthogonal matrix and let $f(\mathbf{U})$ be a real-valued function that is invariant under right orthogonal transformation: for any $d \times d$ orthogonal matrix $\mathbf{O}$, $f(\mathbf{U}\mathbf{O}) = f(\mathbf{U})$. Let $\mathcal{S}_\mathbf{U}$ denote the subspace spanned by the columns of $\mathbf{U}$. We have

$$\mathcal{S}_\mathbf{U} = \{\mathbf{U}\mathbf{O} | \mathbf{O} \in \mathbb{R}^{d \times d}, \mathbf{O}^\top \mathbf{O} = \mathbf{I}_d\} \in \mathcal{G}_{(d,p)},$$

where $\mathcal{G}_{(d,p)}$ is the Grassmann manifold of all d-dimensional subspaces in $\mathbb{R}^p$. The goal is to find the subspace $\widehat{\mathcal{S}}_\mathbf{U}$ such that

$$\widehat{\mathcal{S}}_\mathbf{U} = \arg \max_{\mathcal{S}_\mathbf{U} \in \mathcal{G}_{(d,p)}} f(\mathbf{U}).$$

Steepest ascent is the closest numerical analogy to the Grassmann manifold algorithm described in this article. Our algorithm is an iterative procedure that, given a point $\mathcal{S}_{\mathbf{U}}$, computes an ascent direction where an increase in f is found. However, this procedure on a Grassmann manifold is not as straightforward as a steepest ascent method could be in a Euclidean space.

En route to presenting the algorithm, a few terms need to be defined. Let $\mathbf{V}$ be a $p \times (p-d)$ semi-orthogonal matrix that is a completion of $\mathbf{U}$, such that $\mathbf{Q} = (\mathbf{U}, \mathbf{V})$ is a $p \times p$ orthogonal matrix. Let $\mathbf{B} = (\nabla f(\mathbf{U}))^\top \mathbf{V}$, the $d \times (p-d)$ matrix that is the directional derivative of f, which is the rate of change of $f(\mathbf{U})$ in the direction of $\mathbf{V}$. And finally, let $\mathbf{A}$ be a $p \times p$ skew-symmetric matrix define as

$$\mathbf{A} = \begin{pmatrix} 0_d & \mathbf{B} \\ -\mathbf{B}^\top & 0_{p-d.} \end{pmatrix}$$

In a nutshell, the algorithm works by rotating the starting basis $\mathbf{Q}$ to a new basis by right multiplication by an orthogonal matrix. A single step of the algorithm for a step size $\delta \in (0, 1)$ involves the update

$$\mathbf{Q}_{t+1} = \mathbf{Q}_t \exp\{\delta \mathbf{A}\}. \tag{1}$$

The matrix $\mathbf{A}$ depends on the directional derivative $\mathbf{B}$ that changes at each iteration, and $\mathbf{A}$ is updated during each iteration until a stopping criterion is met.

For $\widehat{\mathcal{S}}_{\mathbf{U}}$ to be the a maximizer of the objective function f, a necessary condition is that for any tangent vector at $\widehat{\mathcal{S}}_{\mathbf{U}}$, the directional derivative of f in the direction of that vector should be zero. Practically, that means that $\|\mathbf{B}\|$, the norm of $\mathbf{B}$ should be sufficiently small. To summarize, the algorithm is the following:

1. Provide an initial matrix $p \times p$ matrix $\mathbf{Q}_0$ at $t = 0$ and then
2. Do until $\|\mathbf{B}\| < \varepsilon$
    a. Compute the directional derivative $\mathbf{B}$ and form the matrix $\mathbf{A}$
    b. Update $\mathbf{Q}_{t+1} = \mathbf{Q}_t \exp\{\delta \mathbf{A}\}$ such that $f(\mathbf{U}_{t+1}) > f(\mathbf{U}_t)$.
3. $\widehat{\mathcal{S}}_{\mathbf{U}}$ is the subspace spanned by the first d columns of $\mathbf{Q}$ at the last iteration.

Numerically, the choice of δ is not trivial. In our implementation, a discrete optimization of f over the range of δ is used. Specifically at each iteration, a sequence of values of $\delta \in (0, 1)$ is used to generate candidates $\mathbf{Q}_{t+1}$. The candidate that yields the largest $f(\mathbf{U}_{t+1})$ greater than $f(\mathbf{U}_t)$ is used.

This gradient method requires the directional derivative $\mathbf{B} = (\nabla f(\mathbf{U}))^\top \mathbf{V}$ of the objective function. In some applications, an explicit analytical expression for $\mathbf{B}$ can be obtained. As an example, consider the objective function $f(\mathbf{U}) = \mathrm{Tr}\{\mathbf{U}^\top \mathbf{W} \mathbf{U}\}$ where $\mathrm{Tr}\{\mathbf{S}\}$ represents the trace of the square matrix $\mathbf{S}$. Differentiating f with respect to the elements of $\mathbf{U}$ yields $(\mathbf{W} + \mathbf{W}^\top)\mathbf{U}$. The analytical expression of the directional derivative is then $\mathbf{U}^\top(\mathbf{W} + \mathbf{W}^\top)\mathbf{V}$. In many other cases an approximation of the directional derivative can be computed. Liu *et al.* (2004) provided a method to approximate $\mathbf{B} = (\alpha_{ij})$ at any $\mathcal{S}_{\mathbf{U}} \in \mathcal{G}_{(d,p)}$ where

$$\alpha_{ij} \approx \frac{f(\widetilde{\mathbf{U}}) - f(\mathbf{U})}{\epsilon}, \quad i = 1, ..., d;\ j = 1, ..., p - d \tag{2}$$

for a small value of $\epsilon > 0$ and $\widetilde{\mathbf{U}} = \mathbf{Q}\exp\{\epsilon\mathbf{E}_{ij}\}\mathbf{J}$. The term $\mathbf{E}_{ij}$ is a $p \times p$ skew-symmetric matrix with all entries being 0 except 1 in position (i, j) and -1 in position (j, i) and $\mathbf{J}$ is the $p \times d$ matrix of the first d columns of the $p \times p$ identity matrix.

2.2. Stochastic gradient

The basic gradient algorithm has the potential drawback to yield a local optimum at convergence when the starting value is not properly chosen. To help avoid this possibility and to reach a global optimum, we adapted the Markov chain-type simulated annealing process proposed by Gallivan *et al.* (2003).

Let $\mathbf{U}_0$ be a representative of a starting subspace $\mathcal{S}_{\mathbf{U}_0} \in \mathcal{G}_{(d,p)}$. Set the iteration index $t = 0$ and choose an initial "temperature" T_0.

1. Repeat m times
   (a) Calculate the directional derivative matrix $\mathbf{B}$ of f.
   (b) Generate $d(p - d)$ independent realizations, w_{ij}'s, from the standard normal distribution. Calculate a candidate value $\mathbf{Y}$ following Equation 1, starting from $\mathbf{Q}_t = (\mathbf{U}_t, \mathbf{V}_t)$ in the direction of $\mathbf{B} + \sqrt{T_t}\mathbf{W}$ where $\mathbf{W}$ is the $d \times (p - d)$ matrix of (w_{ij}).
   (c) Compute $f(\mathbf{Y})$, $f(\mathbf{U}_t)$, and set $df = f(\mathbf{Y}) - f(\mathbf{U}_t)$.
   (d) Set $\mathbf{U}_{t+1} = \mathbf{Y}$ with probability $\min\{\exp\{df/T_t\}, 1\}$, else set $\mathbf{U}_{t+1} = \mathbf{U}_t$.
2. Decrease the temperature T_t to T_{t+1}, with $T_{t+1} = T_t/\tau$ where $\tau > 1$ is the cooling ratio.
3. Set $t = t + 1$ and go to Step 1 if T_{t+1} is greater than a threshold. Stop otherwise.

Simulating annealing can be quite time-consuming. The user provides the maximum number of iterations m, the cooling ratio τ specified in this algorithm and the initial temperature T_0. Ideally, m should be large enough to allow the Markov chain at each temperature to be approximately in its stationary distribution before cooling. The cooling schedule should be slow. Various cooling schedules are suggested in the literature. Nourani and Andresen (1998) studied the performance of linear, logarithmic and exponential cooling schedules. Givens and Hoeting (2005, Section 3.4.1.2) also mentioned various cooling schedules. In general, a good cooling strategy should decrease the temperature rapidly at first and slowly over time. But in the above algorithm, a simple linear cooling schedule is adopted. Here, we suggest setting the cooling ratio τ close to 1. The value of the initial temperature T_0 is problem-dependent, but it must be chosen properly. A useful strategy is to choose $T_0 > 0$ so that $\exp\{[f(\mathbf{Y}_1) - f(\mathbf{Y}_2)]/T_0\}$ is close to 1 for any pair of candidates matrices $\mathbf{Y}_1$ and $\mathbf{Y}_2$.

3. Relevance

Computing eigenspaces is a task that occurs in many research areas, including signal processing (Comon and Golub 1990), data mining (Berry, Drmac, and Jessup 1999), and control theory (Patel, Laub, and Van Dooren 1994). The overarching goal is to reduce the complexity of a problem by focusing on a subset of relevant quantities that are dependent on eigenspaces.

In statistics, a small number of principal components may be computed to capture the important variability in the data. Many eigenvalue problems are optimization problems on manifolds (Absil *et al.* 2004; Edelman *et al.* 1998). In this section, we present three problems that can be phrased as manifold optimization problems. The first is related to linear discriminant analysis, the second is about independent components analysis and the third describes Cook's principal fitted components models.

3.1. Discriminant analysis

In a typical classification problem, let us suppose there are K classes that need to be separated in p-dimensional space. When p is large, the information relevant to the separation of the classes may be largely contained in a few directions $\boldsymbol{\alpha}_1, ..., \boldsymbol{\alpha}_d \in \mathbb{R}^p$ where $d < p$. One goal of discriminant analysis is to identify these directions. Fisher's linear discriminant analysis (LDA) problem amounts to maximizing the function

$$f(\boldsymbol{\alpha}) = \frac{\boldsymbol{\alpha}^\top \mathbf{B} \boldsymbol{\alpha}}{\boldsymbol{\alpha}^\top \mathbf{W} \boldsymbol{\alpha}} \tag{3}$$

over $\boldsymbol{\alpha} \in \mathbb{R}^p$, where $\mathbf{B}$ is the between-class covariance matrix and $\mathbf{W}$ is the within-class covariance matrix. Equivalently, we can find $\arg\max_{\boldsymbol{\alpha}} \boldsymbol{\alpha}^\top \mathbf{B} \boldsymbol{\alpha}$ subject to $\boldsymbol{\alpha}^\top \mathbf{W} \boldsymbol{\alpha} = 1$. To estimate the d directions, standard practice in multivariate statistics is to use sequential maximization of the objective function: Supposing that $\{\boldsymbol{\alpha}_1, ..., \boldsymbol{\alpha}_{m-1}\}$ are the first $m-1$ discriminant directions that maximize Equation 3, the next is $\boldsymbol{\alpha}_m = \arg\max_{\boldsymbol{\alpha}} f(\boldsymbol{\alpha})$ subject to $\boldsymbol{\alpha}^\top \mathbf{W} \boldsymbol{\alpha}_j = 0, j = 1, ..., m-1$. In fact, the sequential optimization of the function $f(\boldsymbol{\alpha})$, identified as a Rayleigh quotient, is a generalized eigenvalue problem that can be solved simultaneously on a manifold. To see this, let $\mathbf{U} = (\boldsymbol{\alpha}_1, \ldots, \boldsymbol{\alpha}_d)$ and consider the generalized Rayleigh quotient $f(\mathbf{U}) = \mathrm{Tr}\{\mathbf{U}^\top \mathbf{B} \mathbf{U} (\mathbf{U}^\top \mathbf{W} \mathbf{U})^{-1}\}$. With the change of variables $\mathbf{Y} = \mathbf{W}^{1/2} \mathbf{U}$ and $\mathbf{B}^* = \mathbf{W}^{-1/2} \mathbf{B} \mathbf{W}^{-1/2}$, maximizing $f(\mathbf{U})$ is equivalent to maximizing $\mathrm{Tr}\{\mathbf{Y}^\top \mathbf{B}^* \mathbf{Y}\}$ subject to $\mathbf{Y}^\top \mathbf{Y} = \mathbf{I}$. This maximum is achieved when $\mathrm{span}(\mathbf{Y})$ equals the span of the first d eigenvectors of $\mathbf{B}^*$, which is the same as $\mathrm{span}(\boldsymbol{\alpha}_1, \ldots, \boldsymbol{\alpha}_d)$ from the usual sequential procedure. Viewed as manifold optimization, there is nothing special about $\boldsymbol{\alpha}_1, \ldots, \boldsymbol{\alpha}_d$ other than the subspace they span.

In the case of Fisher's linear discriminant, sequential and simultaneous optimization yield the same subspace, but this equivalence will not always hold, as illustrated by the following two examples. The first example is the LDA projection pursuit index for classification (Lee, Cook, Klinke, and Lumley 2005). Based on Fisher's LDA, the index expression is given by

$$f(\mathbf{U}) = \begin{cases} 1 - \frac{|\mathbf{U}^\top \mathbf{W} \mathbf{U}|}{|\mathbf{U}^\top (\mathbf{W}+\mathbf{B}) \mathbf{U}|} & \text{for } |\mathbf{U}^\top (\mathbf{W}+\mathbf{B}) \mathbf{U}| \neq 0 \\ 0 & \text{for } |\mathbf{U}^\top (\mathbf{W}+\mathbf{B}) \mathbf{U}| = 0, \end{cases} \tag{4}$$

where $\mathbf{U}$ is a $(p \times d)$ semi-orthogonal matrix. Using this index, the directions provided by the subspace spanned by the columns of $\mathbf{U}$ that maximize $f(\mathbf{U})$ reveal differences between classes. For the second example, consider the optimal Bayes rule for classifying a new feature vector $\mathbf{x}$ into one of K normal populations with known means $\boldsymbol{\mu}_j$ and variances $\boldsymbol{\Sigma}_j$, $j = 1, \ldots, K$, based on d linear combinations $\boldsymbol{y} = \mathbf{U}^\top \mathbf{x}$, where $\mathbf{U}$ is again a $p \times d$ matrix with rank d. Using an optimal Bayes rule, the probability of correct classification is (Guseman, Peters, and Walker 1975)

$$f(\mathbf{U}) = \int_{\mathbb{R}^d} \max_{1 \leq j \leq K} N_p(\boldsymbol{y} | \mathbf{U}^\top \boldsymbol{\mu}_j, \mathbf{U}^\top \boldsymbol{\Sigma}_j \mathbf{U}) d\boldsymbol{y},$$

where N_p is the p-dimensional normal density evaluated at $\boldsymbol{y}$ with the indicated mean and variance. It can be seen that in both examples, $f(\mathbf{U})$ depends only on span($\mathbf{U}$) and thus maximizing f is again a Grassmann optimization problem. Sequential and simultaneous optimization will not necessarily yield the same subspace for these objective functions. This might be appreciated by recalling that sequential optimization requires an external inner product, and there does not seem to be a clear choice for that inner product in these optimization problems. Relatedly, Cook and Forzani (2010) discuss sequential versus full Grassmann optimization, and present an example illustrating the potential downside of sequential methods.

3.2. Independent components analysis

The independent component analysis (ICA) of a random vector consists of searching for a linear transformation that minimizes the statistical dependence between its components. Also known as *blind source separation* or referred to as *sources separation problem*, it was, according to Comon (1994), first proposed by J. Herault and C. Jutten around 1986. It has been used in biomedical applications like image analysis and in signal processing. A recurring example of the application of ICA is the *cocktail party problem* where the task is to recover p source signals $\mathbf{s}(t) = \{s_1(t), ..., s_p(t)\}$, supposed to be statistically independent, from recordings where they appear as linear mixtures $\mathbf{x}(t) = \{x_1, ..., x_n\}$. Mathematically, there is a mixing matrix $\mathbf{A}$ such that $\mathbf{x}(t) = \mathbf{A}\mathbf{s}(t)$ and the goal of ICA is to identify the mixing matrix.

The problem is often translated into finding an $n \times p$ de-mixing matrix $\mathbf{W}$ such that the signals $y(t) = \mathbf{W}^\top \mathbf{x}(t)$ are as independent as possible (Absil *et al.* 2004). The quantification of the independence involves a cost function f such that $f(\mathbf{WD}) = f(\mathbf{W})$ for all nonsingular diagonal matrices $\mathbf{D}$. Because of the invariance property of f, its optimization must be restrained to constrained sets corresponding to equivalence classes $\{\mathbf{WD} : \mathbf{D} \text{ diagonal}\}$. A possible choice for the constraint set is the oblique manifold

$$\mathcal{OB} = \{\mathbf{W} \in \mathbb{R}^{n \times p} : \text{Diag}(\mathbf{W}\mathbf{W}^\top) = \mathbf{I}_n\}. \tag{5}$$

Thus, algorithms for independence component analysis can be formulated in terms of manifolds (Douglas 2000; Pham 2001). See Comon (1994) and Yeredor (2002) for further details on ICA.

3.3. Principal fitted components

One of the oldest and best known methods for reducing dimensionality in multivariate problems is principal component analysis (PCA). But PCA in a regression setting does not make any use of the outcome in reducing the dimensionality of the predictors. Cook (2007) proposed principal fitted component (PFC) models that can outperform PCA as a reductive method for regression. In this section, we discuss one specific PFC model where parameter estimation is carried over a Grassmann manifold.

Let $\mathbf{X} = (X_1, ..., X_p)^\top$ be a p-vector of random predictors and Y be the response. Cook (2007), using the stochastic nature of the predictors, proposed the following model:

$$\mathbf{X}_y = \boldsymbol{\mu} + \boldsymbol{\Gamma}\boldsymbol{\beta}\mathbf{f}_y + \boldsymbol{\varepsilon}. \tag{6}$$

Here, $\mathbf{X}_y$ denotes the conditional $\mathbf{X}$ given $Y = y$, $\mathbf{f}_y \in \mathbb{R}^r$ is a function of the response, $\boldsymbol{\mu} \in \mathbb{R}^p$, $\boldsymbol{\Gamma} \in \mathbb{R}^{p \times d}$ is a semi-orthogonal matrix and $\boldsymbol{\beta} \in \mathbb{R}^{d \times r}$. The error term $\boldsymbol{\varepsilon} \in \mathbb{R}^p$ is assumed to be normally distributed with mean 0 and variance $\boldsymbol{\Delta}$.

Essentially, this model can be seen as a way to partition the information in the predictors given the response into two parts: one part ($\boldsymbol{\Gamma}\boldsymbol{\beta}\mathbf{f}_y = \mathrm{E}(\mathbf{X}_y - \mathbf{X})$) is related to the response and the other ($\boldsymbol{\mu}+\boldsymbol{\varepsilon}$) that is not. The term that is related to y suggests that the translated conditional means $\mathrm{E}(\mathbf{X}_y - \mathbf{X})$ fall in the d-dimensional subspace $\mathcal{S}_{\boldsymbol{\Gamma}}$. Under the appropriate structure on $\boldsymbol{\Delta}$, the sufficient reduction is $\boldsymbol{\Gamma}^\top\mathbf{X}$, and thus $\mathbf{X}$ can be replaced by $\boldsymbol{\Gamma}^\top\mathbf{X}$ without loss of information about the regression of Y on $\mathbf{X}$. Here, the concept of sufficient reduction follows the definition of Cook (2007): A reduction $R : \mathbb{R}^p \to \mathbb{R}^d, d \leq p$ is sufficient if $Y \perp\!\!\!\perp \mathbf{X}|R(\mathbf{X})$. However, as $\boldsymbol{\Gamma}^\top\mathbf{X}$ is a sufficient reduction, for any $d \times d$ orthogonal matrix $\mathbf{O}$, $(\boldsymbol{\Gamma}\mathbf{O})^\top\mathbf{X}$ is also sufficient. Thus $\boldsymbol{\Gamma}$ is not estimable but the subspace spanned by its columns is estimable. The goal of an analysis is then to estimate the subspace $\mathcal{S}_{\boldsymbol{\Gamma}}$ spanned by the columns of $\boldsymbol{\Gamma}$.

The estimation of $\mathcal{S}_{\boldsymbol{\Gamma}}$ depends on the structure of $\boldsymbol{\Delta}$. Various structures for $\boldsymbol{\Delta}$ can be modeled, including isotropic ($\sigma^2\mathbf{I}$), diagonal ($\sigma_1^2, ..., \sigma_p^2$), and general unstructured $\boldsymbol{\Delta} > 0$ cases. The data structure dictates the variance structure to be used. While an isotropic structure of the variance may be appropriate for conditionally independent predictors that are on the same numerical scale, a diagonal structure may fit better when on different measurement scales, and an unstructured variance could be considered with predictors that are conditionally dependent. An interesting case arises with heterogeneous errors yielding

$$\boldsymbol{\Delta} = \boldsymbol{\Gamma}\boldsymbol{\Omega}\boldsymbol{\Gamma}^\top + \boldsymbol{\Gamma}_0\boldsymbol{\Omega}_0\boldsymbol{\Gamma}_0^\top, \tag{7}$$

where $\boldsymbol{\Gamma}_0$ is the orthogonal completion of $\boldsymbol{\Gamma}$ such that $(\boldsymbol{\Gamma}, \boldsymbol{\Gamma}_0)$ is a $p \times p$ orthogonal matrix. The matrices $\boldsymbol{\Omega} \in \mathbb{R}^{d\times d}$ and $\boldsymbol{\Omega}_0 \in \mathbb{R}^{(p-d)\times(p-d)}$ are assumed to be symmetric and full-rank. Under model (6) with covariance structure (7), Y is independent of $\mathbf{X}$ given $\boldsymbol{\Gamma}^\top\mathbf{X}$ and consequently $\boldsymbol{\Gamma}^\top\mathbf{X}$ carries all the predictive information that $\mathbf{X}$ has about Y.

We now turn to the maximum likelihood estimation of $\mathcal{S}_{\boldsymbol{\Gamma}}$ which involves a Grassmann manifold optimization. Assuming that a set of n data points is observed, let $\bar{\mathbf{X}}$ be the sample mean of $\mathbf{X}$ and let $\mathbb{X}$ denote the $n \times p$ matrix with rows $(\mathbf{X} - \bar{\mathbf{X}})^\top$. Let $\mathbb{F}$ denote the $n \times r$ matrix with rows $(\mathbf{f}_y - \bar{\mathbf{f}})^\top$ and set $\mathbf{P}_{\mathbb{F}}$ to denote the linear operator that projects onto the subspace spanned by the columns of $\mathbb{F}$. Also let $\widehat{\mathbb{X}} = \mathbf{P}_{\mathbb{F}}\mathbb{X}$ denote the $n \times p$ matrix of the fitted values from the multivariate linear regression of $\mathbf{X}$ on $\mathbf{f}_y$. Setting $\widehat{\boldsymbol{\Sigma}} = \mathbb{X}^\top\mathbb{X}/n$ and $\widehat{\boldsymbol{\Sigma}}_{\text{fit}} = \widehat{\mathbb{X}}^\top\widehat{\mathbb{X}}/n$, we let $\widehat{\boldsymbol{\Sigma}}_{\text{res}} = \widehat{\boldsymbol{\Sigma}} - \widehat{\boldsymbol{\Sigma}}_{\text{fit}}$. Then the maximum likelihood estimator of $\mathcal{S}_{\boldsymbol{\Gamma}}$ is

$$\widehat{\mathcal{S}}_{\boldsymbol{\Gamma}} = \arg\max_{\mathcal{S}_{\mathbf{U}} \in \mathcal{G}_{(d,p)}} f(\mathbf{U}), \tag{8}$$

where

$$f(\mathbf{U}) = -(n/2)\log|\mathbf{U}^\top\widehat{\boldsymbol{\Sigma}}_{\text{res}}\mathbf{U}| - (n/2)\log|\mathbf{V}^\top\widehat{\boldsymbol{\Sigma}}\mathbf{V}| \tag{9}$$

and $\mathbf{V}$ is an orthogonal completion of $\mathbf{U}$. The objective function is a real-valued function with its argument being a $p \times d$ semi-orthogonal matrix. The invariant property of f, $f(\mathbf{U}\mathbf{O}) = f(\mathbf{U})$ for any $d \times d$ orthogonal matrix $\mathbf{O}$, and the orthogonal constraint $\mathbf{U}^\top\mathbf{U} = \mathbf{I}_d$ set up the optimization of f to be carried over the Grassmann manifold $\mathcal{G}_{(d,p)}$, which is the parameter space for $\mathcal{S}_{\boldsymbol{\Gamma}}$.

Optimization over a Grassmann manifold is used in several other statistical models for dimension reduction, including covariance reducing models (Cook and Forzani 2008), envelope models (Cook, Li, and Chiaromonte 2010), generalized PFC (Cook and Li 2009), and likelihood acquired directions (Cook and Forzani 2009). In Section 5, an application to a set of data is presented where LAD is reviewed briefly and used.

4. The R package GrassmannOptim

In this section we give specific examples on how to use **GrassmannOptim**. The main visible function of this package is of the same name as the package name. To use this function, the user needs to provide the objective function; providing the analytical expression of the directional derivative is not required.

The call of the function, as described in the package manual, is

```
GrassmannOptim(objfun, W, sim_anneal = FALSE, temp_init = 20,
  cooling_rate = 2, max_iter_sa = 100, eps_conv = 1e-05, max_iter = 100,
  eps_grad = 1e-05, eps_f = .Machine$double.eps, verbose = FALSE)
```

As an illustration on the use of **GrassmannOptim**, we consider the statistical model described in Section 3.3. The directional derivative $\mathbf{B}$ of $f(\mathbf{U})$ is

$$\mathbf{B} = 2\{(\mathbf{U}^\top \widehat{\boldsymbol{\Sigma}}_{\text{res}} \mathbf{U})^{-1} \mathbf{U}^\top \widehat{\boldsymbol{\Sigma}}_{\text{res}} \mathbf{V} - \mathbf{U}^\top \widehat{\boldsymbol{\Sigma}} \mathbf{V} (\mathbf{V}^\top \widehat{\boldsymbol{\Sigma}} \mathbf{V})^{-1}\}. \tag{10}$$

We generated a dataset following model (6), with y generated using $Y = U + e$ where $U \sim$ uniform$(-2, 2)$ and $e \sim N(0, 0.1)$. We set $\mathbf{f}_y = (y, y^2, y^3)^\top$ and used eight predictors with 300 observations. We set $\boldsymbol{\Gamma}$ to be the first three columns and $\boldsymbol{\Gamma}_0$ to be the last 5 columns of the identity matrix $\mathbf{I}_8$, $\boldsymbol{\beta} = 3\text{Diag}(1, 0.4, 0.4)$, and $\boldsymbol{\Omega}_0 = 4\mathbf{I}_5$. With $\sigma = 1.5$, $\rho = 0.5$ and $J_3 = (1, 1, 1)^\top$, $\boldsymbol{\Omega}$ was set to $\sigma^2 \mathbf{I}_3 + \rho\sigma^2(J_3 J_3^\top - \mathbf{I}_3)$. The term ρ controls the correlation between the first three predictors of $\mathbf{X}_y$. The following code is used to generate the data, using the **mvtnorm** (Genz *et al.* 2012) package:

```
R> library("mvtnorm")
R> set.seed(123)
R> p <- 8
R> nobs <- 300
R> d <- 3
R> sigma <- 1.5
R> sigma0 <- 2
R> rho <- 0.5
R> y <- array(runif(n = nobs, min = -2, max = 2), c(nobs, 1)) +
+    rnorm(n = nobs, sd = 0.1)
R> fy <- scale(cbind(y, y^2, y^3), TRUE, FALSE)
R> Gamma <- diag(p)[, 1:3]
R> Gamma0 <- diag(p)[, -(1:3)]
R> Omega <- sigma^2 * matrix(rho, ncol = 3, nrow = 3)
R> diag(Omega) <- sigma^2
R> Delta <- Gamma %*% Omega %*% t(Gamma) + sigma0^2 * Gamma0 %*% t(Gamma0)
R> Err <- t(rmvnorm(n = nobs, mean = c(rep(0, p)), sigma = Delta))
R> beta <- diag(3 * c(1, 0.4, 0.4))
R> X <- t(Gamma %*% beta %*% t(fy) + Err)
```

With this simulated data, the first three predictors are related to the response y with respectively a linear, quadratic and cubic relationship. The last 5 predictors are not related to the response. The following code is used to obtain Figure 1(left) which shows a scatter plot matrix of the first four predictors with the response.

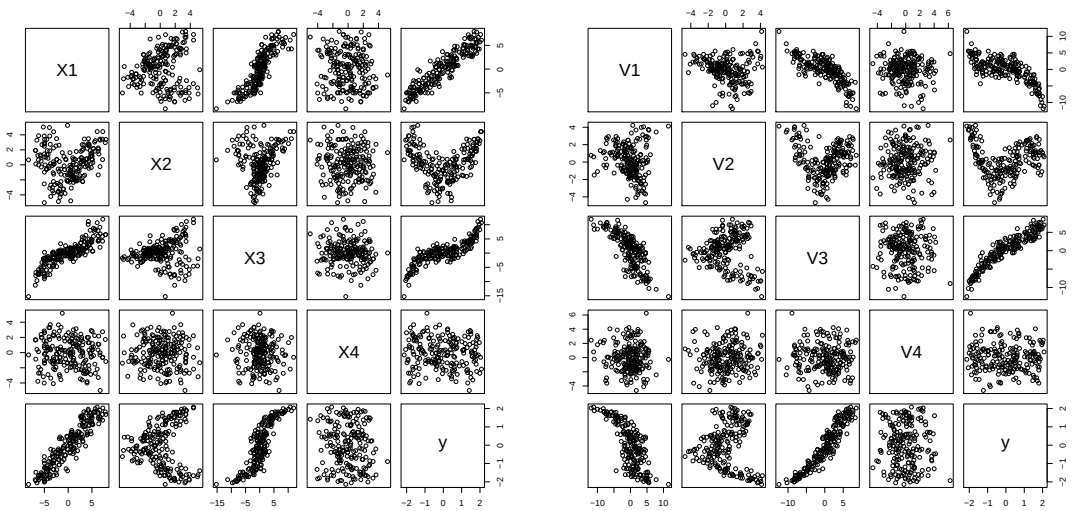

Figure 1: Scatter plots matrix of the first four predictors and the response (left), and the projected data onto the first four directions of the Grassmann optimization solution and the response (right).

```
R> x <- data.frame(cbind(X[, 1:4], y))
R> colnames(x) <- c("X1", "X2", "X3", "X4", "y")
R> pairs(x)
```

The fitting procedure requires the computation of some terms that are shown next.

```
R> P_F <- fy %*% solve(t(fy) %*% fy) %*% t(fy)
R> Xc <- scale(X, TRUE, FALSE)
R> S_fit <- t(Xc) %*% P_F %*% Xc/nobs
R> S <- t(Xc) %*% Xc/nobs
R> S_res <- S - S_fit
R> sigmas <- list(S = S, S_fit = S_fit, S_res = S_res, p = p, nobs = nobs)
```

We provided the objective function (`f`) and the directional derivative (`Gradient`) as follows:

```
R> f <- function(W) {
+    Qt <- W$Qt
+    d <- W$dim[1]
+    p <- ncol(Qt)
+    Sigmas <- W$sigmas
+    U <- matrix(Qt[, 1:d], ncol = d)
+    V <- matrix(Qt[, (d + 1):p], ncol = (p - d))
+    return(-log(det(t(V) %*% Sigmas$S %*% V)) -
+      log(det(t(U) %*% Sigmas$S_res %*% U)))
+  }
R> Gradient <- function(W){
+    Qt <- W$Qt
```

```
+    d <- W$dim[1]
+    p <- ncol(Qt)
+    Sigmas <- W$sigmas
+    U <- matrix(Qt[, 1:d], ncol = d)
+    V <- matrix(Qt[, (d + 1):p], ncol = (p-d))
+    terme1 <- solve(t(U) %*% Sigmas$S_res %*% U) %*%
+      t(U) %*% Sigmas$S_res %*% V
+    terme2 <- t(U) %*% Sigmas$S %*% V %*% solve(t(V) %*% Sigmas$S %*%V)
+    return(2 * (terme1 - terme2))
+  }
```

Both functions are called within the function called `objfun` that returns values of the objective function and the directional derivative for a specified set of arguments.

```
R> objfun <- function(W) list(value = f(W), gradient = Gradient(W))
```

The call of the function **GrassmannOptim** requires a minimum set of arguments. Other than `objfun`, the object `W`, which is of type `list`, is needed. As a list object, it may contain the optional initial starting matrix `Qt` that is a $p \times p$ orthogonal matrix. Regardless of `Qt` being provided or not, `W` must contain at least two components: the first is `dim = c(d, p)` where d is the dimension of the subspace sought and p is the number of predictors ($d < p$); the second is any list of arguments to be passed to the objective function `f` and the directional derivative `Gradient`. In the case of the current example, that list is `sigmas`.

We present three scenarios to illustrate the use of **GrassmannOptim**. In the context of this application, $d = 3$ is the number of columns of $\mathbf{\Gamma}$ and $p = 8$. In the first, no initial matrix is provided. The call is:

```
R> W <- list(dim = c(d, p), sigmas = sigmas)
R> ans <- GrassmannOptim(objfun, W, eps_conv = 1e-4, verbose = TRUE)

Initialization...
iter    Loglik          Gradient
   1    -1.0027e+01     5.2477e+00
   2    -9.9171e+00     4.5877e+00
   3    -9.5656e+00     1.4027e+00
 ...       ...             ...
  26    -8.3892e+00     1.9602e-04
  29    -8.3892e+00     4.0443e-05

R> ans$converged

[1] TRUE
```

In this case, when no starting matrix is provided, a random starting matrix is used. With `verbose = TRUE`, iteration results are shown with the iteration number `iter`; the values of the objective function are shown under `Loglik` and the norms of the directional derivative

are shown under `Gradient`. If convergence is reached, then the component `converged` of the output list is `TRUE`.

In many scenarios, there might be good guesses for starting matrices. For this particular example, it is worth using the matrix of eigenvectors of the fitted covariance matrix $\widehat{\boldsymbol{\Sigma}}_{\text{fit}}$.

```
R> Qt <- svd(S_fit)$u
R> W <- list(Qt = Qt, dim = c(d, p), sigmas = sigmas)
R> ans <- GrassmannOptim(objfun, W, eps_conv = 1e-4, verbose = TRUE)

Initialization...
iter    Loglik          Gradient
   1    -8.4266e+00     2.6095e-01
   2    -8.4010e+00     2.5774e-01
   3    -8.3905e+00     1.3484e-02
   4    -8.3895e+00     6.8860e-03
   5    -8.3893e+00     6.7825e-04
   6    -8.3892e+00     2.6774e-04
   7    -8.3892e+00     1.1446e-04
   8    -8.3892e+00     9.7860e-05

R> ans$converged

[1] TRUE
```

Here convergence is reached much faster than by using a random start. When there is no good guess for the starting matrix, the simulated annealing method may become handy and quite powerful, although it requires more computing time.

```
R> W <- list(dim = c(d, p), sigmas = sigmas)
R> ans <- GrassmannOptim(objfun, W, eps_conv = 1e-4, sim_anneal = TRUE,
+    max_iter_sa = 100, verbose = TRUE)

iter    Loglik          Gradient
   1    -8.3892e+00     3.4433e-04
   2    -8.3892e+00     1.4293e-04
   4    -8.3892e+00     1.2478e-04
   6    -8.3892e+00     7.3144e-05

R> ans$converged

[1] TRUE
```

The process converges after only a handful iterations at the cost of the long simulated annealing process. At convergence, the first d components of the matrix output `Qt` is the solution to the optimization problem. This matrix is the basis of the estimate of the three dimensional subspace $\mathcal{S}_{\boldsymbol{\Gamma}}$ in $\mathbb{R}^8$.

```
R> ans$Qt[, 1:3]

        [,1]    [,2]    [,3]
[1,] -0.2278 -0.2017 -0.9509
[2,]  0.0895 -0.9716  0.1882
[3,] -0.9694 -0.0406  0.2413
[4,]  0.0033  0.0556  0.0092
[5,] -0.0087 -0.0749  0.0208
[6,]  0.0138  0.0283  0.0208
[7,] -0.0074 -0.0134 -0.0307
[8,] -0.0056 -0.0633 -0.0142
```

The projected data onto the directions given by first three columns of `Qt` are plotted on Figure 1(right) using the following code.

```
R> x <- data.frame(cbind(X %*% ans$Qt[, 1:4], y))
R> colnames(x) <- c("V1", "V2", "V3", "V4", "y")
R> pairs(x)
```

The similarity between these two plots seems evident, where the newly obtained variables `V1`, `V2` and `V3` appear to display respectively a cubic, a quadratic, and a linear relationship with the response. The graph of the last variable `V4` on the response shows a null plot with no relationship as expected.

5. Application

We applied **GrassmannOptim** to a dataset used by Cook and Forzani (2009) in their development and exposition of LAD, which requires Grassmann manifold optimization. Cook and Forzani (2009) carried out the optimization using the MATLAB (The MathWorks, Inc. 2010) program **sg_min** (Lippert 2004), which enables us to contrast results from **GrassmannOptim** with those from a different code written in a different language. We briefly review the optimization portion of LAD and then turn to the example.

5.1. Likelihood acquired direction

Consider a regression in which the response Y is discrete with support $S_Y = \{1, 2, ..., h\}$. Following standard practice, continuous response can be sliced into finite categories to meet this condition. Let $\mathbf{X}_y \in \mathbb{R}^p$ denote a random vector of predictors distributed as $\mathbf{X}|(Y = y)$ and assume that $\mathbf{X}_y \sim N(\boldsymbol{\mu}_y, \boldsymbol{\Delta}_y)$, $y \in S_Y$. Let $\boldsymbol{\mu} = \mathrm{E}(\mathbf{X})$ and $\boldsymbol{\Sigma} = \mathrm{var}(\mathbf{X})$ denote the marginal mean and variance of $\mathbf{X}$ and let $\boldsymbol{\Delta} = \mathrm{E}(\boldsymbol{\Delta}_Y)$ denote the average covariance matrix. Given n_y independent observations of $\mathbf{X}_y, y \in S_Y$, the goal is to obtain the maximum likelihood estimate of the d-dimensional central subspace $\mathcal{S}_{Y|\mathbf{X}}$, which is defined informally as the smallest subspace such that Y is independent of $\mathbf{X}$ given its projection $\mathbf{P}_{\mathcal{S}_{Y|\mathbf{X}}}\mathbf{X}$ onto $\mathcal{S}_{Y|\mathbf{X}}$.

Let $\widetilde{\boldsymbol{\Sigma}}$ denote the sample covariance matrix of $\mathbf{X}$, let $\widetilde{\boldsymbol{\Delta}}_y$ denote the sample covariance matrix for the data with $Y = y$, and let $\widetilde{\boldsymbol{\Delta}} = \sum_{y=1}^{h} f_y \widetilde{\boldsymbol{\Delta}}_y$ where f_y is the fraction of cases observed

with $Y = y$. The maximum likelihood estimator of $\mathcal{S}_{Y|\mathbf{X}}$ maximizes over $\mathcal{S} \in \mathcal{G}_{(d,p)}$ the log-likelihood function

$$L(\mathcal{S}) = \frac{n}{2} \log |\mathbf{P}_{\mathcal{S}} \widetilde{\mathbf{\Sigma}} \mathbf{P}_{\mathcal{S}}|_0 - \frac{n}{2} \log |\widetilde{\mathbf{\Sigma}}| - \frac{1}{2} \sum_{y=1}^{h} n_y \log |\mathbf{P}_{\mathcal{S}} \widetilde{\mathbf{\Delta}}_y \mathbf{P}_{\mathcal{S}}|_0, \tag{11}$$

where $|\mathbf{A}|_0$ indicates the product of the non-zero eigenvalues of a positive semi-definite symmetric matrix $\mathbf{A}$ and $\mathbf{P}_{\mathcal{S}}$ indicates the projection onto the subspace $\mathcal{S}$ in the usual inner product. The desired reduction is then $\hat{\boldsymbol{\eta}}^\top \mathbf{X}$, where the columns of $\hat{\boldsymbol{\eta}}$ are a basis for the maximum likelihood estimate of $\mathcal{S}_{Y|\mathbf{X}}$.

5.2. Is it a bird, a plane or a car?

The dataset is from a pilot study to assess the possibility of distinguishing birds, planes and cars by the sounds they make. The goal of the study was the reconstruction of sonic maps that identify both the level and source of sound.

A two-hour recording was made in a city, and then five second snippets of sounds were selected. This resulted in 58 recordings identified as birds, 43 as cars and 64 as planes. Each recording was processed and ultimately represented by 13 SDMFCCs (Scale Dependent Mel-Frequency Cepstrum Coefficients). The focus was on reducing this dimension of the 13-dimensional feature vector.

We applied LAD to estimate the central subspace via Grassmann manifold optimization using our package. The first two columns of the estimated basis matrix were used to form the two LAD predictors. These two LAD predictors, shown in Figure 2, nicely separate the sound sources. Figure 2 is quite similar to the one found in Cook and Forzani (2009). The orientation here is different than that found by Cook and Forzani (2009), but this is to be expected since the goal is to estimate a subspace and not a particular basis.

Cook and Forzani (2009) and Cook, Forzani, and Tomassi (2011) adapted Lippert (2004)'s **sg_min** 2.4.1 MATLAB (The MathWorks, Inc. 2010) package for Grassmann optimization with analytical first derivatives and numerical second derivatives, using Newton-Raphson iteration on $\mathcal{G}_{(d,p)}$. With **GrassmannOptim**, we used only numerical first derivatives with no use of the second derivatives. We also used the simulated annealing procedure to help reach a global optimum. Given these distinctions and other potentially relevant differences like starting values, we found the agreement between out results and those found by Cook and Forzani (2009) to be remarkable. The full R code used to obtain Figure 2 is provided in the online supplements. (More optimization examples using **GrassmannOptim** are available at `http://www.math.umbc.edu/~kofi/GrassmannOptim/`.)

6. Discussion

We have introduced **GrassmannOptim**, an R package for Grassmann manifold optimization, and described how many subspace optimization problems can be identified as manifold optimization. The current package uses a gradient-based algorithm. Minimally, the user is required to provide the objective function and the dimensions of the basis of the subspace to be estimated. Analytical expressions of the directional derivative typically result in more efficient estimation. However, analytic derivatives are not required.

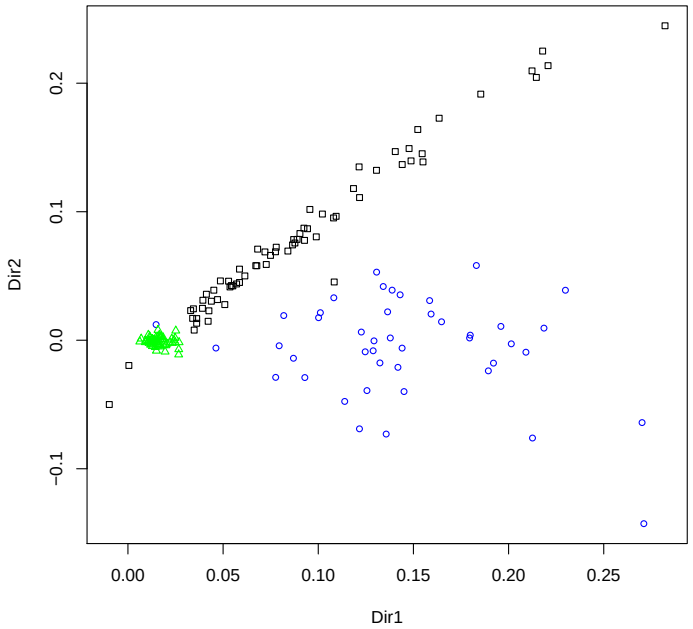

Figure 2: Plot of the first two LAD directions: cars (blue ○'s), planes (black □'s), and birds (green ▷'s).

Limited theoretical results are available to analyze the convergence of the algorithm. But in practice, the convergence appears linear. The dimension of the Grassmann manifold affects greatly the speed of the package. We are dealing with a space where a single object is a subspace of dimension $d \times (p - d)$. When p is large, the algorithm becomes memory-greedy and slow.

A simulated annealing method was added to the package to help a global search. Without it, there is a chance that the algorithm could convergence to a local maximum. A good starting value can also help avoid local minima. As a Markov chain method, the processing time for simulated annealing depends on many elements, including the cooling schedule, the number of iterations at each stage of the Markov chain and the dimension of the subspace to optimize. The choice of the initial temperature, the cooling schedule and the maximum number of iterations at each temperature are left to the discretion of the user.

Acknowledgments

We are thankful to the two anonymous referees whose comments helped improve the content of this article. The authors' research was supported by grant T32HL072757 from the National Heart, Lung, and Blood Institute and by grant DMS-1007547 from the National Science Foundation. The content is solely the responsibility of the authors and does not necessarily represent the official views of the National Heart, Lung, and Blood Institute, the National Institutes of Health, the National Science Foundation or Fannie Mae.

References

Absil P, Mahony R, Sepulchre R (2004). *Optimization Algorithms on Matrix Manifolds*. Princeton University Press, Princetion.

Berry MW, Drmac Z, Jessup ER (1999). "Matrices, Vector Spaces, and Information Retrieval." *SIAM Review*, **41**(2), 335–362.

Chikuse Y (2003). *Statistics on Special Manifolds*. Number 174 in Lecture Notes in Statistics. Springer-Verlag.

Comon P (1994). "Independent Component Analysis, A New Concept?" *Signal Processing*, **36**, 287–314.

Comon P, Golub GH (1990). "Tracking a Few Extreme Singular Values and Vectors in Signal Processing." *IEEE Proceedings*, **78**(8), 1327–1343.

Cook RD (2007). "Fisher Lecture: Dimension Reduction in Regression." *Statistical Science*, **22**(1), 1–26.

Cook RD, Forzani LM (2008). "Covariance Reducing Models: An Alternative to Spectral Modelling of Covariance." *Biometrika*, **95**(4), 799–812.

Cook RD, Forzani LM (2009). "Likelihood-Based Sufficient Dimension Reduction." *Journal of the American Statistical Association*, **104**(485), 197–208.

Cook RD, Forzani LM (2010). "Letter to the Editor: Reply to Zhu and Hastie." *Journal of the American Statistical Association*, **105**(490), 881.

Cook RD, Forzani LM, Tomassi DR (2011). "**LDR**: A Package for Likelihood-Based Sufficient Dimension Reduction." *Journal of Statistical Software*, **39**(3), 1–20. URL http://www.jstatsoft.org/v39/i03/.

Cook RD, Li B, Chiaromonte F (2010). "Envelope Models for Parsimonious and Efficient Multivariate Linear Regression." *Statistica Sinica*, **20**(3), 927–1010.

Cook RD, Li L (2009). "Dimension Reduction in Regression with Exponential Family Predictors." *Journal of Computational and Graphical Statistics*, **18**(3), 774–791.

Douglas SC (2000). "Self-Stabilized Gradient Algorithms for Blind Source Separation with Orthogonality Constraints." *IEEE Transactions on Neural Networks*, **11**(6), 1490–1497.

Edelman A, Arias T, Smith S (1998). "The Geometry of Algorithms with Orthogonality Constraints." *SIAM Journal of Matrix Analysis and Applications*, **20**(2), 303–353.

Fletcher R (1987). *Practical Methods of Optimization*. 2nd edition. John Wiley & Sons, Hoboken.

Gallivan K, Srivastava A, Liu X, Dooren P (2003). "Efficient Algorithms for Inferences on Grassmann Manifolds." In *Proceedings of the 12th IEEE Workshop on Statistical Signal Processing*, pp. 301–304.

Genz A, Bretz F, Miwa T, Mi X, Leisch F, Scheipl F, Hothorn T (2012). ***mvtnorm**: Multivariate Normal and t Distributions*. R package version 0.9-9992, URL http://CRAN.R-project.org/package=mvtnorm.

Givens G, Hoeting J (2005). *Computational Statistics*. John Wiley & Sons, Hoboken.

Gruber P, Theis F (2006). "Grassmann Clustering." In *Proceedings of the European Signal Processing Conference*.

Guseman LF, Peters BC, Walker HF (1975). "On the Probability of Misclassification for Linear Feature Selection." *The Annals of Statistics*, **3**, 661–668.

Lee EK, Cook RD, Klinke S, Lumley T (2005). "Projection Pursuit for Exploratory Supervised Classification." *Journal of Computational and Graphical Statistics*, **14**(4), 831–846.

Lippert R (2004). "**sg_min**: Stiefel Grassmann Optimization." MATLAB package version 2.4.1, URL http://web.mit.edu/~ripper/www/sgmin.html.

Liu X, Srivastava A, Gallivan K (2004). "Optimal Linear Representations of Images for Object Recognition." *IEEE Transactions on Pattern Analysis and Machine Intelligence*, **26**(5), 662–666.

Manton JH (2002). "Optimization Algorithms Exploiting Unitary Constraints." *IEEE Transactions on Signal Processing*, **50**(3).

Nilsson J, Sha F, Jordan M (2007). "Regression on Manifolds Using Kernel Dimension Reduction." In *Proceedings of the 24th International Conference on Machine Learning*. Corvallis.

Nocedal J, Wright S (1999). *Numerical Optimization*. Springer-Verlag, New York.

Nourani Y, Andresen B (1998). "A Comparison of Simulated Annealing Cooling Strategies." *Journal of Physics A: Mathematical and General*, **31**, 8373–8385.

Patel RV, Laub AJ, Van Dooren PM (1994). *Numerical Linear Algebra Techniques for Systems and Control*. IEEE Press, Piscataway.

Pham DT (2001). "Joint Approximate Diagonalization of Positive Definite Hermitian Matrices." *SIAM Journal of Matrix Analysis and Applications*, **22**(4), 1136–1152.

R Development Core Team (2012). *R: A Language and Environment for Statistical Computing*. R Foundation for Statistical Computing, Vienna, Austria. ISBN 3-900051-07-0, URL http://www.R-project.org/.

The MathWorks, Inc (2010). *MATLAB – The Language of Technical Computing, Version 7.10*. The MathWorks, Inc., Natick, Massachusetts. URL http://www.mathworks.com/products/matlab/.

Wong YC (1967). "Differential Geometry of Grassmann Manifolds." In *Proceedings of the National Academy of Science*, volume 57, pp. 589–594. USA.

Yeredor A (2002). "Non-Orthogonal Joint Diagonalization in the Least-Squares Sense with Application in Blind Source Separation." *IEEE Transactions on Signal Processing*, **50**(7), 1545–1553.

Affiliation:

Kofi Placid Adragni
Department of Mathematics & Statistics
University of Maryland, Baltimore County
1000 Hilltop Circle
Baltimore, MD 21250, United States of America
E-mail: kofi@umbc.edu
URL: http://www.math.umbc.edu/~kofi/
Phone: +1/410/455-2406
Fax: +1/410/455-1066

Journal of Statistical Software http://www.jstatsoft.org/
published by the American Statistical Association http://www.amstat.org/
Volume 50, Issue 5 *Submitted:* 2010-02-09
July 2012 *Accepted:* 2012-05-08