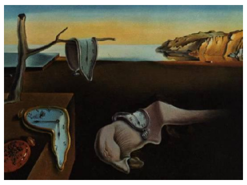

Journal of Statistical Software

MMMMMM YYYY, Volume VV, Issue II. *http://www.jstatsoft.org/*

SpaSM: A Matlab Toolbox for Sparse Statistical Modeling

Karl Sjöstrand
Technical University of Denmark
EXINI Diagnostics AB

Line Harder Clemmensen
Technical University of Denmark

Rasmus Larsen
Technical University of Denmark

Bjarne Ersbøll
Technical University of Denmark

Abstract

Applications in biotechnology such as gene expression analysis and image processing have led to a tremendous development of statistical methods with emphasis on reliable solutions to severely underdetermined systems, and interpretation, solutions where the surplus of inputs have been reduced to a concise model. At the core of this development are methods which augments the standard linear models for regression, classification and decomposition such that sparse solutions are obtained. This toolbox aims at making public carefully implemented and well-tested variants of the most popular such methods for the Matlab programming environment. The toolbox builds on code made public in 2005 and which has since been used in several studies.

Keywords: Least angle regression, LASSO, elastic net, sparse principal component analysis, sparse discriminant analysis, Matlab.

1. Introduction

The introduction of the Least Angle Regression method for regularized/sparse regression (Efron, Hastie, Johnstone, and Tibshirani 2004) marked the starting point of a series of important contributions to the statistical computing community with the following common properties:

- Solutions are obtained sequentially along a path of gradually changing amounts of regularization. This paper focuses on methods where the method coefficients are piecewise linear functions of the regularization parameter, and where algorithms proceed by finding the next piecewise linear breakpoint,

- For sufficient amounts of regularization, solutions are *sparse*, i.e., some of the coefficients of the model are exactly zero, leading to more compact models which are easier to interpret,
- Methods are *efficient*, meaning they perform on a par with competing statistical methods when performance is measured on a test data set.

Examples of contributions are (Zou and Hastie 2005), (Zou, Hastie, and Tibshirani 2006), (Rosset and Zhu 2007), (Hastie, Rosset, Tibshirani, and Zhu 2004), (Park and Hastie 2007), (Friedman, Hastie, and Tibshirani 2010), of which the first three are detailed in this paper. The methods cover regression (the LASSO and the Elastic Net with ridge regression as a special case), classification (sparse discriminant analysis with penalized linear discriminant analysis as a special case), and unsupervised modeling (sparse principal component analysis). The goal of this paper is to provide reference Matlab (The MathWorks Inc. 2010) implementations of these basic regularization-path oriented methods. We also present previously unpublished developments of the algorithm for sparse principal component analysis and provide some evidence that performance is somewhat higher, while the computational complexity is significantly lowered. The implementation strikes a balance between performance and readability, making this toolbox a good starting point for learning the details of the methods. For this reason, the code is written as pure Matlab scripts which closely follows the algorithms provided here. All methods have been fully described and validated in their respective publications; despite this we provide terse but relatively complete derivations of each algorithm such that the paper can be read and the algorithms understood without having all references at hand.

2. In the toolbox

The toolbox consists of a series of Matlab (The MathWorks Inc. 2010) scripts to build and apply various statistical models for both supervised and unsupervised analyses. Below are listings of each method, file, subfunction and utility.

2.1. Methods

Forward Selection A variant of stepwise regression in which variables are included one-by-one based on their correlation with the current residual vector. Provides a baseline algorithm for other sparse methods for regression in this toolbox.

Least Angle Regression Provides a more gentle version of the classical approach of forward selection regression. The algorithm is the basis for all other methods in the toolbox. The method is also an interesting statistical method in its own right.

LASSO This method adds l_1 (1-norm) regularization to ordinary least squares regression, yielding solutions which are sparse in terms of the regression coefficients. This may lead to efficient suppression of noise and aids in interpretation.

Elastic Net Combining the algorithmic ideas of Least Angle Regression, the computational benefits of ridge regression and the tendency towards sparse solutions of the LASSO, this versatile method is applicable for many data sets, also when the number of predictor variables far exceed the number of observations. The corresponding LARS-EN algorithm is used in the implementation of the following two algorithms.

Sparse Principal Component Analysis Principal component analysis is a powerful tool for compacting a data set and for recovering latent structures in data, but solutions are difficult to interpret as they involve all the original predictor variables. Sparse principal component analysis approximates the behavior of regular principal component analysis but models each component as a linear combination of a subset of the original variables.

Sparse Linear Discriminant Analysis Linear discriminant analysis is a standard tool for classification of observations into one of two or more groups. Further, the data can be visualized along the obtained discriminative directions. As with principal component analysis, these directions are combinations of all predictor variables. Sparse discriminant analysis reduces this to a subset of variables which may improve performance as well interpretability.

2.2. Files

`forwardselection.m` A baseline algorithm for variable selection. Based on the algorithm in `lar.m`.

`lar.m` An implementation of the LARS algorithm for least angle regression described by Efron *et al.* (2004)

`lasso.m` The LASSO method of Tibshirani (1996), implemented using a combination of the algorithms of Efron *et al.* (2004) and Rosset and Zhu (2007)

`elasticnet.m` The Elastic Net algorithm of Zou and Hastie (2005), with elements from Rosset and Zhu (2007)

`spca.m` The sparse principal component algorithm based on the work by Zou *et al.* (2006), with modification described below

`slda.m` The sparse discriminant analysis of Clemmensen, Hastie, Witten, and Ersbøll (2011).

2.3. Sub-functions and utilities

`larsen.m` The actual implementation of the Elastic Net algorithm. The functions `lasso.m`, `elasticnet.m`, `spca.m` and `slda.m` depend on this function; however it is not intended for direct use

`cholinsert.m` Update of the Cholesky factorization of $\mathbf{X}^T\mathbf{X} + \delta\mathbf{I}$. Used in `lar.m` and `larsen.m`.

`choldelete.m` Downdate of the above Cholesky factorization. Used in `larsen.m`.

`center.m` Convenience function for centering (removing the mean observation) a data matrix or response vector.

`normalize.m` Convenience function for centering and normalizing a data matrix or response matrix such that variables have unit Euclidean length.

3. Methods and algorithms

This section presents the principles behind each method in the toolbox, and outlines their algorithms. The basic building block is the LARS-EN algorithm (Zou and Hastie 2005) which encompasses regression via ordinary least squares, ridge regression, the LASSO and the Elastic Net. These are based on the linear model $\mathbf{y} = \mathbf{X}\beta + \boldsymbol{\varepsilon}$ where $\mathbf{y}$ $(n \times 1)$ is the observed response variable, $\mathbf{X}$ $(n \times p)$ is the data matrix where the ith column represents the ith predictor variable, β $(p \times 1)$ is the set of model coefficients which determines the load on each predictor variable, and $\boldsymbol{\varepsilon}$ are the residual errors. Unless stated otherwise, $\mathbf{y}$ is assumed centered and $\mathbf{X}$ is assumed centered and normalized such that each variable has zero mean and unit Euclidean length. A sparse method for regression estimates a coefficient vector β with many zero elements, giving an estimate $\hat{\mathbf{y}}$ of $\mathbf{y}$ which is a linear combination of a subset of available variables in $\mathbf{X}$. Sparse solutions may be preferred to full counterparts if the latent linear model can be assumed to be sparse, or when interpretation of the results is important. The set $\mathcal{A}$ denotes the indices in β corresponding to non-zero elements; we refer to this as the *active set*. The set $\mathcal{I}$ is called the *inactive set* and denotes the complement of $\mathcal{A}$. We use these sets also to denote submatrices such as the $(n \times |\mathcal{A}|)$ matrix $\mathbf{X}_{\mathcal{A}}$, consisting of the columns (variables) of $\mathbf{X}$ corresponding to the indices in $\mathcal{A}$. All algorithms proceed in iterations and we indicate iteration number by a parenthesized superscript number, e.g., $\hat{\beta}^{(k)}$ for the regression coefficients calculated in the kth iteration. It is further convenient to define an operator $\min^{+}(\cdot)$ which finds the smallest strictly positive value of the (vector-valued) input.

The methods for regression described below proceed in an iterative manner, adding or subtracting variables in the model in each step. The methods start with the trivial constant model, then move towards the full representation which corresponds to ordinary least squares regression or ridge regression, depending on the type of regularization. To put the presentation of these algorithms into perspective, we begin with a quick review of one of the simplest algorithms of this kind.

In *forward selection*, a variant of *stepwise regression*, variables are added one-by-one until some goodness-of-fit criterion is fulfilled. The next variable to include in this scheme can be chosen based on a number of criteria. The methods in this toolbox generally pick the variable that has the highest absolute correlation with the current residual vector. To fix the terminology and to give a simple baseline algorithm we state a forward selection algorithm in Algorithm 1.

In this algorithm, we move to the least squares solution using all currently active variables in each step. This approach may be overly greedy; there may be inactive variables that are beneficial to include before the least squares solution is reached. The following sections

Algorithm 1 Forward Selection

```
1: Initialize the active set $\mathcal{A} = \emptyset$ and the inactive set $\mathcal{I} = \{1 \dots p\}$
2: Initialize the coefficient vector $\beta^{(0)} = \mathbf{0}$
3: for $k \in \{0 \dots p-1\}$ do
4:     Find variable maximally correlated with the current residual $i = \arg\max_{i \in \mathcal{I}} \mathbf{x}_i^T(\mathbf{y} - \mathbf{X}\beta^{(k)})$
5:     Move $i$ from $\mathcal{I}$ to $\mathcal{A}$.
6:     Update the active set coefficients $\beta_{\mathcal{A}}^{(k+1)} = (\mathbf{X}_{\mathcal{A}}^T \mathbf{X}_{\mathcal{A}})^{-1} \mathbf{X}_{\mathcal{A}}^T \mathbf{y}$
7: end for
8: Output the series of coefficients $\mathbf{B} = [\beta^{(0)} \dots \beta^{(p)}]$.
```

cover less greedy variations on the forward selection scheme which result in algorithms with generally better performance and which are able to handle more difficult data sets.

3.1. Least angle regression

Least Angle Regression (LAR) is a regression method that provides a more gentle version of forward selection. Conceptually, LAR modifies Algorithm 1 on only one account. Instead of choosing a step size which yields the (partial) least squares solution in each step, we shorten the step length such that we stop when any inactive variable becomes equally important as the active variable in terms of correlation with the residual vector. That variable is then included in the active set and a new direction is calculated. Recall that all active variables are uncorrelated with the residual vector at the least squares solution, the step length will therefore always be as short or shorter at the point where we find the next active variable to include than that of the least squares solution.

The algorithm starts with the empty set of active variables. The correlation between each variable and the response is measured, and the variable with the highest correlation becomes the first variable included into the model. The first direction is then towards the least squares solution using this single active variable. Walking along this direction, the angles between the variables and the residual vector are measured. Along this walk, the angles will change; in particular, the correlation between the residual vector and the active variable will shrink linearly towards 0. At some stage before this point, another variable will obtain the same correlation with respect to the residual vector as the active variable. The walk stops and the new variable is added to the active set. The new direction of the walk is towards the least squares solution of the two active variables, and so on. After p steps, the full least squares solution will be reached.

The LAR algorithm is efficient since there is a closed form solution for the step length at each stage. Denoting the model estimate of $\mathbf{y}$ at iteration k by $\hat{\mathbf{y}}^{(k)}$ and the least squares solution including the newly added active variable $\hat{\mathbf{y}}_{\text{OLS}}^{(k+1)}$, the walk from $\hat{\mathbf{y}}^{(k)}$ towards $\hat{\mathbf{y}}_{\text{OLS}}^{(k+1)}$ can be formulated $(1-\gamma)\hat{\mathbf{y}}^{(k)} + \gamma\hat{\mathbf{y}}_{\text{OLS}}^{(k+1)}$ where $0 \leq \gamma \leq 1$. Estimating $\hat{\mathbf{y}}^{(k+1)}$, the position where the next active variable is to be added, then amounts to estimating γ. We seek the smallest positive γ where correlations become equal, that is

$$\mathbf{x}_{i \in \mathcal{I}}^T(\mathbf{y} - (1-\gamma)\hat{\mathbf{y}}^{(k)} - \gamma\hat{\mathbf{y}}_{\text{OLS}}^{(k+1)}) = \mathbf{x}_{j \in \mathcal{A}}^T(\mathbf{y} - (1-\gamma)\hat{\mathbf{y}}^{(k)} - \gamma\hat{\mathbf{y}}_{\text{OLS}}^{(k+1)}). \qquad (1)$$

Solving this expression for γ, we get

$$\gamma_{i\in\mathcal{I}} = \frac{(\mathbf{x}_i - \mathbf{x}_j)^T(\mathbf{y} - \hat{\mathbf{y}}^{(k)})}{(\mathbf{x}_i - \mathbf{x}_j)^T(\hat{\mathbf{y}}_{\text{OLS}}^{(k+1)} - \hat{\mathbf{y}}^{(k)})} = \frac{(\mathbf{x}_i - \mathbf{x}_j)^T\boldsymbol{\varepsilon}}{(\mathbf{x}_i - \mathbf{x}_j)^T\mathbf{d}}, \tag{2}$$

where $\mathbf{d} = \hat{\mathbf{y}}_{\text{OLS}}^{(k+1)} - \hat{\mathbf{y}}^{(k)}$ is the direction of the walk, and $j \in \mathcal{A}$. Now, $\mathbf{d}$ is the orthogonal projection of $\boldsymbol{\varepsilon}$ onto the plane spanned by the variables in $\mathcal{A}$, therefore we have $\mathbf{x}_j^T\boldsymbol{\varepsilon} = \mathbf{x}_j^T\mathbf{d} \equiv c$, representing the angle at the current breakpoint $\hat{\mathbf{y}}^{(k)}$. Furthermore, the sign of the correlation between variables is irrelevant. Therefore, we have

$$\gamma = \min_{i\in\mathcal{I}}\left\{\frac{\mathbf{x}_i^T\boldsymbol{\varepsilon} - c}{\mathbf{x}_i^T\mathbf{d} - c}, \quad \frac{\mathbf{x}_i^T\boldsymbol{\varepsilon} + c}{\mathbf{x}_i^T\mathbf{d} + c}\right\}, \quad 0 < \gamma \le 1, \tag{3}$$

where the two terms are for correlations/angles of equal and opposite sign respectively. The coefficients at this next step are given by

$$\beta^{(k+1)} = (1-\gamma)\beta^{(k)} + \gamma\beta_{\text{OLS}}^{(k+1)}. \tag{4}$$

Given these key pieces of the LAR algorithm, we state the entire procedure in Algorithm 2.

Algorithm 2 Least Angle Regression

```
 1: Initialize the coefficient vector $\beta^{(0)} = \mathbf{0}$ and the fitted vector $\hat{\mathbf{y}}^{(0)} = \mathbf{0}$,
 2: Initialize the active set $\mathcal{A} = \emptyset$ and the inactive set $\mathcal{I} = \{1 \dots p\}$
 3: for $k = 0$ to $p - 2$ do
 4:    Update the residual $\boldsymbol{\varepsilon} = \mathbf{y} - \hat{\mathbf{y}}^{(k)}$
 5:    Find the maximal correlation $c = \max_{i\in\mathcal{I}} |\mathbf{x}_i^T\boldsymbol{\varepsilon}|$
 6:    Move variable corresponding to $c$ from $\mathcal{I}$ to $\mathcal{A}$.
 7:    Calculate the least squares solution $\beta_{\text{OLS}}^{(k+1)} = (\mathbf{X}_{\mathcal{A}}^T\mathbf{X}_{\mathcal{A}})^{-1}\mathbf{X}_{\mathcal{A}}^T\mathbf{y}$
 8:    Calculate the current direction $\mathbf{d} = \mathbf{X}_{\mathcal{A}}\beta_{\text{OLS}}^{(k+1)} - \hat{\mathbf{y}}^{(k)}$
 9:    Calculate the step length $\gamma = \min^{+}_{i\in\mathcal{I}}\left\{\frac{\mathbf{x}_i^T\boldsymbol{\varepsilon}-c}{\mathbf{x}_i^T\mathbf{d}-c}, \frac{\mathbf{x}_i^T\boldsymbol{\varepsilon}+c}{\mathbf{x}_i^T\mathbf{d}+c}\right\}$, $0 < \gamma \le 1$
10:    Update regression coefficients $\beta^{(k+1)} = (1-\gamma)\beta^{(k)} + \gamma\beta_{\text{OLS}}^{(k+1)}$
11:    Update the fitted vector $\hat{\mathbf{y}}^{(k+1)} = \hat{\mathbf{y}}^{(k)} + \gamma\mathbf{d}$
12: end for
13: Let $\beta^{(p)}$ be the full least squares solution $\beta^{(p)} = (\mathbf{X}^T\mathbf{X})^{-1}\mathbf{X}^T\mathbf{y}$
14: Output the series of coefficients $\mathbf{B} = [\beta^{(0)} \dots \beta^{(p)}]$
```

Each step of Algorithm 2 adds a covariate to the model until the full least squares solution is reached. It is natural to parameterize this process by the size of the coefficients at each step as well as in between steps of the algorithm. The algorithm returns the following parametrization,

$$s(\beta) = \|\beta\|_1 = \sum_{i=1}^{p} |\beta_i|. \tag{5}$$

Picking a suitable model for a particular analysis thus means selecting a suitable value of $s \in (0, \|\beta_{OLS}\|_1)$. Cross-validation or an independent validation data set are obvious choices for this purpose, however, the algorithm provides information which substitute or complement this process.

Degrees of freedom Efron *et al.* (2004) showed that the number of degrees of freedom at each step of the LAR algorithm is well approximated by the number of non-zero elements of β. The algorithm therefore returns the following sequence,

$$df_{LAR}^{(k)} = |\mathcal{A}| = k, \quad k = 0 \dots p. \tag{6}$$

Mallow's C_p Given the above measure of the number of degrees of freedom, we can calculate a number of model selection criteria. Mallow's C_p measure is defined as (Zou, Hastie, and Tibshirani 2007)

$$C_p^{(k)} = \frac{1}{\sigma_\varepsilon^2}\|\mathbf{y} - \mathbf{X}\beta^{(k)}\|^2 - n + 2df^{(k)}. \tag{7}$$

Akaike's Information Criterion Akaiket's information criterion is similar to Mallow's C_p and is defined as

$$AIC^{(k)} = \|\mathbf{y} - \mathbf{X}\beta^{(k)}\|^2 + 2\sigma_\varepsilon^2 df^{(k)}. \tag{8}$$

Bayesian Information Criterion The Bayesian information criterion tends to choose a more sparse model than both AIC and C_p and is defined as

$$BIC^{(k)} = \|\mathbf{y} - \mathbf{X}\beta^{(k)}\|^2 + \log(n)\sigma_\varepsilon^2 df^{(k)}. \tag{9}$$

The latter three criteria can be used to pick a suitable model, typically indicated by the smallest value of each criterion. Alternatively, one can choose the sparsest model for which more complex models lead to scant improvements in the relevant model selection criterion. The measure σ_ε^2 represents the residual variance of a low-bias model which is here defined as

$$\sigma_\varepsilon^2 = \frac{1}{n}\|\mathbf{y} - \mathbf{X}^\dagger\mathbf{y}\|^2, \tag{10}$$

where $\mathbf{X}^\dagger$ is the Moore-Penrose pseudo-inverse of $\mathbf{X}$, equivalent to a ridge regression solution $\arg\min \|\mathbf{y} - \mathbf{X}\beta\|^2 + \lambda\|\beta\|^2$ in the limit $\lambda \to 0$. Note that in cases where $p > n$, this measure of the residual variance will be zero which in effect turns the information criteria defined above into a measure of training error only. We therefore recommend using these criteria for model selection only in cases where n is well above p.

The key computational burden of Algorithm 2 lies in Step 7 where the OLS solution involving the variables in $\mathcal{A}$ is calculated. Two techniques are used to alleviate this. For problems where n is at least ten times larger than p, we calculate the full Gram matrix $\mathbf{X}^T\mathbf{X}$ once and use the submatrix $\mathbf{X}_\mathcal{A}^T\mathbf{X}_\mathcal{A}$ to find $\beta_{\text{OLS}}^{(k+1)}$, thus avoiding an $\mathcal{O}(|\mathcal{A}|^2 n)$ matrix multiplication. When $p > 1000$, this method is not preferred since the memory footprint of the resulting $p \times p$ Gram matrix may pose a problem. In cases where $10n < p$, or when a pre-computed Gram matrix is impractical, we maintain a matrix $\mathbf{R}$ of the Cholesky factorization of the current Gram (sub)matrix $\mathbf{X}_\mathcal{A}^T\mathbf{X}_\mathcal{A}$ such that $\mathbf{R}^T\mathbf{R} = \mathbf{X}_\mathcal{A}^T\mathbf{X}_\mathcal{A}$. As variables join the active set, $\mathbf{R}$ can be updated with low computational cost. The conditions chosen for selecting between the two methods are not exact, but we have gathered evidence through simulation studies that they work well on most standard computers.

In practice one frequently has a notion of the sparsity of the desired solution when running Algorithm 2. To avoid unnecessary computations, the algorithm can be stopped prematurely, either when the active set reaches a certain size, or when the l_1 norm of the coefficients in $\beta^{(k)}$ exceeds a preset threshold. Optionally, the algorithm stores and returns the solution fulfilling the specified sparsity criterion only in order to save computer resources.

3.2. The LASSO

The LASSO (Tibshirani 1996) represents the most basic augmentation of the ordinary least squares solution which implements coefficient shrinkage and selection. The sum of squared residuals loss function $L(\hat{\beta}(\lambda))$ is combined with a penalty function $J(\hat{\beta}(\lambda))$ based on the l_1 norm as,

$$\hat{\beta}(\lambda) = \arg\min_{\beta} L(\hat{\beta}(\lambda)) + \lambda J(\hat{\beta}(\lambda)) = \arg\min_{\beta} \|\mathbf{y} - \mathbf{X}\beta\|^2 + \lambda\|\beta\|_1. \tag{11}$$

The l_1 penalty will promote sparse solutions. This means that as λ is increased, elements of $\hat{\beta}(\lambda)$ will become exactly zero. Due to the non-differentiability of the penalty function, there are no closed-form solutions to Equation 11. A number of algorithms have been proposed (e.g., (Fu 1998; Osborne, Presnell, and Turlach 2000; Friedman *et al.* 2010)) including the quadratic programming approach on an expanded space of variables outlined in the original LASSO paper of Tibshirani (1996). The algorithm presented here is due to Rosset and Zhu (2007) who derived a sufficient condition for piecewise linear coefficient paths on which they based several LASSO-type methods. The LASSO algorithm described here is a special case of their work. Efron *et al.* (2004) arrived at an equivalent algorithm by showing that a small modification to the Least Angle Algorithm yields LASSO solutions.

The goal of this section is to derive an expression for how the solutions of Equation 11 change with λ. The solution set $\hat{\beta}(\lambda)$ will hit a non-differentiability point when coefficients either go from non-zero to zero (join $\mathcal{I}$), or the other way around (join $\mathcal{A}$). Assume first that we are in a region of values of λ where variables are neither joining nor leaving $\mathcal{A}$. The normal equations to Equation 11 around λ and around a nearby point $\lambda + \epsilon$ are then

$$-2\mathbf{X}_{\mathcal{A}}^T(\mathbf{y} - \mathbf{X}_{\mathcal{A}}\hat{\beta}_{\mathcal{A}}(\lambda)) + \lambda \cdot \mathrm{sign}(\hat{\beta}_{\mathcal{A}}(\lambda)) = 0 \tag{12}$$

$$-2\mathbf{X}_{\mathcal{A}}^T(\mathbf{y} - \mathbf{X}_{\mathcal{A}}\hat{\beta}_{\mathcal{A}}(\lambda+\epsilon)) + (\lambda+\epsilon) \cdot \mathrm{sign}(\hat{\beta}_{\mathcal{A}}(\lambda+\epsilon)) = 0. \tag{13}$$

We now write Equation 13 as a first order Taylor expansion around $\hat{\beta}(\lambda)$. The general form of a multivariate Taylor expansion of $\mathbf{f}(x)$ around a is

$$\mathbf{f}(x) = \sum_{k=0}^{\infty} \frac{\nabla^{(k)}\mathbf{f}(a)}{k!}(x-a)^k = \mathbf{f}(a) + \nabla\mathbf{f}(a)(x-a) + \frac{1}{2}\nabla^2\mathbf{f}(a)(x-a)^2 + \dots. \tag{14}$$

We have,

$$\mathbf{f}(\hat{\beta}(\lambda)) = -2\mathbf{X}_{\mathcal{A}}^T(\mathbf{y} - \mathbf{X}_{\mathcal{A}}\hat{\beta}_{\mathcal{A}}(\lambda)) + (\lambda+\epsilon) \cdot \mathrm{sign}(\hat{\beta}_{\mathcal{A}}(\lambda)) \tag{15}$$

$$\nabla\mathbf{f}(\hat{\beta}(\lambda)) = 2\mathbf{X}_{\mathcal{A}}^T\mathbf{X}_{\mathcal{A}} \tag{16}$$

$$\nabla^{(k)}\mathbf{f}(\hat{\beta}(\lambda)) = 0 \quad \text{for } k = 2\dots\infty. \tag{17}$$

The complete expansion of Equation 13 becomes

$$-2\mathbf{X}_{\mathcal{A}}^T(\mathbf{y}-\mathbf{X}_{\mathcal{A}}\hat{\beta}_{\mathcal{A}}(\lambda)) + (\lambda+\epsilon)\cdot\operatorname{sign}(\hat{\beta}_{\mathcal{A}}(\lambda)) + 2\mathbf{X}_{\mathcal{A}}^T\mathbf{X}_{\mathcal{A}}\left(\hat{\beta}_{\mathcal{A}}(\lambda+\epsilon)-\hat{\beta}_{\mathcal{A}}(\lambda)\right) = 0. \quad (18)$$

Using Equation 12, we can rearrange this expression to

$$\frac{\hat{\beta}_{\mathcal{A}}(\lambda+\epsilon)-\hat{\beta}_{\mathcal{A}}(\lambda)}{\epsilon} = -(2\mathbf{X}_{\mathcal{A}}^T\mathbf{X}_{\mathcal{A}})^{-1}\operatorname{sign}(\hat{\beta}_{\mathcal{A}}(\lambda)), \quad (19)$$

which approaches $\nabla\hat{\beta}(\lambda)$ as $\epsilon\to 0$ (using $\nabla\hat{\beta}_{\mathcal{I}}(\lambda)=0$). This is a constant function which means that the coefficient paths between *events* (changes to $\mathcal{A}$ and $\mathcal{I}$) are piecewise linear, similarly to Least Angle Regression.

Now that an expression for the change in $\hat{\beta}(\lambda)$ between events has been established, we focus on finding the values of λ for which changes to $\mathcal{A}$ and $\mathcal{I}$ take place. It is beneficial here to consider Equation 11 on an expanded set of β-values, chosen such that $\beta_j=\beta_j^+ + \beta_j^-$ where $\beta_j^+\geq 0$ and $\beta_j^-\geq 0$, $\forall j$,

$$\arg\min_{\beta^+,\beta^-}\|\mathbf{y}-\mathbf{X}(\beta^+-\beta^-)\|^2+\lambda\|(\beta^++\beta^-)\|_1 = L(\hat{\beta}(\lambda))+\lambda\|(\beta^++\beta^-)\|_1 \quad (20)$$
$$\text{such that} \quad \beta_j^+\geq 0, \beta_j^-\geq 0, \forall j.$$

This formulation of the LASSO is differentiable, at the price of having to deal with twice as many variables. The Lagrange primal function is

$$L(\hat{\beta}(\lambda))+\lambda\|(\beta^++\beta^-)\|_1-\sum_{j=1}^p\lambda_j^+\beta_j^+-\sum_{j=1}^p\lambda_j^-\beta_j^-, \quad \lambda_j^+\geq 0, \lambda_j^-\geq 0, \forall j,$$

where we have introduced the Lagrange multipliers λ_j^+ and λ_j^-. The Karush-Kuhn-Tucker conditions are

$$(\nabla L(\beta))_j+\lambda-\lambda_j^+=0 \quad (21)$$
$$-(\nabla L(\beta))_j+\lambda-\lambda_j^-=0 \quad (22)$$
$$\lambda_j^+\beta_j^+=0 \quad (23)$$
$$\lambda_j^-\beta_j^-=0. \quad (24)$$

From these conditions, a number of useful properties arise. First, we note that setting $\lambda=0$ indeed gives us (using Equation 21 and Equation 22) $\nabla L(\beta)=0$ as expected. For positive values of λ we have,

$$\beta_j^+ > 0+ \Rightarrow +\lambda_j^+=0\Rightarrow\nabla L(\beta)=-\lambda\Rightarrow\lambda_j^->0\Rightarrow\beta_j^-=0 \quad (25)$$
$$\beta_j^- > 0+ \Rightarrow +\lambda_j^-=0\Rightarrow\nabla L(\beta)=\lambda\Rightarrow\lambda_j^+>0\Rightarrow\beta_j^+=0. \quad (26)$$

Elements in $\mathcal{A}$ have either $\beta_j^+>0$ or $\beta_j^->0$, but cannot both be non-zero. That is,

$$|(\nabla L(\beta))_j| = \lambda, \quad j\in\mathcal{A} \quad (27)$$
$$|(\nabla L(\beta))_j| \leq \lambda, \quad j\in\mathcal{I}.$$

for $j \in \mathcal{A}$, $|(\nabla L(\beta))_j| = \lambda$ while for elements $j \in \mathcal{I}$, $|(\nabla L(\beta))_j| \leq \lambda$. We are seeking the value of $\gamma > 0$ for which a variable in $\mathcal{I}$ joins $\mathcal{A}$ or vice versa. We have arrived at the following conditions,

$$j \in \mathcal{A} \to \mathcal{I}: \quad \hat{\beta}_j^{(k)} + \gamma \nabla \hat{\beta}_j^{(k)} = 0, \quad j \in \mathcal{A} \tag{28}$$

$$j \in \mathcal{I} \to \mathcal{A}: \quad |(\nabla L(\hat{\beta}^{(k)} + \gamma \nabla \hat{\beta}^{(k)}))_i| = |(\nabla L(\hat{\beta}^{(k)} + \gamma \nabla \hat{\beta}^{(k)}))_j|, \quad j \in \mathcal{A}, i \in \mathcal{I}. \tag{29}$$

The first of these expressions defines the distances $\{\gamma\}$ at which active variables hit zero and join $\mathcal{I}$. The second expression defines the distances at which inactive variables violate the second condition in Equation 27 and thus must join $\mathcal{A}$. Note that any element in $\mathcal{A}$ can be chosen to calculate the RHS of Equation 29, they all equal λ. The smallest value γ_{min} of the distances $\{\gamma\}$ is where the next event will happen. The coefficients can now be updated by

$$\hat{\beta}^{(k+1)} = \hat{\beta}^{(k)} + \gamma_{min} \nabla \hat{\beta}^{(k)}. \tag{30}$$

We have arrived at Algorithm 3 for the LASSO.

Algorithm 3 LASSO (Rosset and Zhu 2007)

```
1: Initialize $\beta^{(0)} = \mathbf{0}$, $\mathcal{A} = \arg\max_j |\mathbf{x}_j^T \mathbf{y}|$, $\nabla \hat{\beta}_{\mathcal{A}}^{(0)} = -\text{sign}(\mathbf{x}_{\mathcal{A}}^T \mathbf{y})$, $\nabla \hat{\beta}_{\mathcal{I}}^{(0)} = 0$, $k = 0$
2: while $\mathcal{I} \neq \emptyset$ do
3:     $\gamma_j = \min^+_{j \in \mathcal{A}} \; -\beta_j^{(k)} / \nabla \hat{\beta}_j^{(k)}$
4:     $\gamma_i = \min^+_{i \in \mathcal{I}} \{ \frac{(\mathbf{x}_i + \mathbf{x}_j)^T (\mathbf{y} - \mathbf{X}\hat{\beta}^{(k)})}{(\mathbf{x}_i + \mathbf{x}_j)^T (\mathbf{X} \nabla \hat{\beta}^{(k)})}, \frac{(\mathbf{x}_i - \mathbf{x}_j)^T (\mathbf{y} - \mathbf{X}\hat{\beta}^{(k)})}{(\mathbf{x}_i - \mathbf{x}_j)^T (\mathbf{X} \nabla \hat{\beta}^{(k)})} \}$ where $j$ is any index in $\mathcal{A}$
5:     $\gamma = \min\{\gamma_j, \gamma_i\}$
6:     if $\gamma = \gamma_j$ then
7:         Move $j$ from $\mathcal{A}$ to $\mathcal{I}$
8:     else
9:         Move $i$ from $\mathcal{I}$ to $\mathcal{A}$
10:    end if
11:    $\hat{\beta}^{(k+1)} = \hat{\beta}^{(k)} + \gamma \nabla \hat{\beta}^{(k)}$
12:    $\nabla \hat{\beta}_{\mathcal{A}}^{(k+1)} = -(2\mathbf{X}_{\mathcal{A}}^T \mathbf{X}_{\mathcal{A}})^{-1} \cdot \text{sign}(\hat{\beta}_{\mathcal{A}}^{(k+1)})$
13:    $k = k + 1$
14: end while
15: Output the series of coefficients $\mathbf{B} = [\beta^{(0)} \ldots \beta^{(k)}]$
```

One of the benefits with this particular algorithm is that the coefficient path can be parameterized either in terms of $\|\hat{\beta}^{(k)}\|_1$, the size of the penalty at iteration k, or the regularization parameter λ. The latter is seldom explicitly specified in path-following algorithms but here, the first identity in Equation 27 provides a way of directly calculating λ as a function of $\hat{\beta}$,

$$\lambda = 2|\mathbf{x}_{j \in \mathcal{A}}^T (\mathbf{y} - \mathbf{X}\hat{\beta})|. \tag{31}$$

Any element in $\mathcal{A}$ will do for this calculation. To minimize the risk of numerical problems, we calculate this value for all elements in $\mathcal{A}$ and pick the median.

If asked for, the algorithm returns the same information as Algorithm 2. The LASSO solution path can be parameterized either in terms of $s(\beta)$ (cf., Equation 5), or in terms of λ which also can be interpreted as a function of β, cf., Equation 31. Zou *et al.* (2007) show that an unbiased estimate of the degrees of freedom of a particular LASSO solution is given by $|\mathcal{A}|$, the number of non-zero components of β. Given this estimate, the various model selection criteria can be calculated as outlined in Section 3.1.

We use the same Gram matrix or Cholesky updating scheme as described in Section 3.1. As variables leave the active set, the Cholesky factorization $\mathbf{R}^T\mathbf{R} = \mathbf{X}_{\mathcal{A}}^T\mathbf{X}_{\mathcal{A}}$ is downdated by removing the contribution to $\mathbf{R}$ which is due to the dropped variable.

3.3. The elastic net

Ridge regression (Hoerl and Kennard 1970) represents an effective way of shrinking the OLS coefficients towards zero. The l_1 penalty of the LASSO is replaced with an l_2 penalty,

$$\hat{\beta}(\delta) = \arg\min_{\beta} \|\mathbf{y} - \mathbf{X}\beta\|^2 + \delta\|\beta\|^2, \tag{32}$$

which leads to the closed form solution

$$\hat{\beta}(\delta) = (\mathbf{X}^T\mathbf{X} + \delta\mathbf{I})^{-1}\mathbf{X}^T\mathbf{y}. \tag{33}$$

Although similar in formulation, ridge regression and the LASSO have important differences. The l_2 penalty of ridge regression leads to a shrinkage of the regression coefficients, much like the l_1 penalty of the LASSO, but coefficients are not forced to exactly zero for finite values of δ. However, a benefit of ridge regression is that a unique solution is available, also when the data matrix $\mathbf{X}$ is rank deficient, e.g., when there are more predictors than observations ($p > n$). This is seen in Equation 33; the addition of a sufficiently large constant value along the diagonal of the Gram matrix $\mathbf{X}^T\mathbf{X}$ ensures full rank (Petersen and Pedersen 2008). The LASSO algorithm (Algorithm 3) is terminated when the active set size $|\mathcal{A}|$ becomes larger than p since the matrix $\mathbf{X}_{\mathcal{A}}^T\mathbf{X}_{\mathcal{A}}$ in Step 12 is no longer invertible.

Elastic net regression (Zou and Hastie 2005) combines the virtues of ridge regression and the LASSO by considering solutions penalized by both an l_2 and an l_1 term,

$$\hat{\beta}(\lambda, \delta) = \arg\min_{\beta} \|\mathbf{y} - \mathbf{X}\beta\|^2 + \delta\|\beta\|^2 + \lambda\|\beta\|_1, \tag{34}$$

thus bridging the gap between the LASSO ($\delta = 0$) and ridge regression ($\lambda = 0$). The l_2 penalty ensures a unique solution also when $p > n$ and the l_1 penalty offers variable selection via a sparse vector of coefficients $\hat{\beta}$. Moreover, the l_2 leads to a *grouping effect* (Zou and Hastie 2005), a term that alludes to the characteristic that highly correlated predictors tend to have similar regression coefficients for nonzero δ. Note however that this does in general not mean that highly correlated variables are included into the active set in groups along the regularization path.

We can use the the LASSO algorithm to obtain the full regularization path of elastic net solutions. To see this, we first note that ridge regression solutions can be obtained by solving an ordinary least squares problem with an augmented set of observations,

$$\tilde{\mathbf{X}} = \begin{bmatrix} \mathbf{X} \\ \sqrt{\delta}\mathbf{I}_p \end{bmatrix}, \quad \tilde{\mathbf{y}} = \begin{bmatrix} \mathbf{y} \\ \mathbf{0} \end{bmatrix}. \tag{35}$$

Expanding the equation $\hat{\beta} = (\tilde{\mathbf{X}}^T\tilde{\mathbf{X}})^{-1}\tilde{\mathbf{X}}^T\tilde{\mathbf{y}}$ gives the ridge solution in Equation 33. For a fixed value of δ, Algorithm 3 offers solutions for all relevant values of λ. Selecting suitable values for the regularization parameters typically involves selecting the best value of λ for a discrete set of values of δ. Thus, the algorithm must be run for each value of δ.

If $p > n$, the augmented data matrix in Equation 35 has size $(n+p) \times p$, implying a system of equations that may be prohibitively large. Remarkably, it turns out that we can do without explicitly forming these augmented matrices, mainly due to the fact that any multiplications with $\tilde{\mathbf{y}}$ effectively voids the contribution of the additional rows in $\tilde{\mathbf{X}}$ since the corresponding rows of $\tilde{\mathbf{y}}$ are zero. Other computations are dot products between vectors with additional elements in $\mathcal{I}$ and vectors with additional elements in $\mathcal{A}$. Since these never coincide ($\mathcal{I} = \mathcal{A}^c$), these additional elements do not contribute to the result. The partial OLS solution calculated in Step 12 must however take into account the additional rows $\mathbf{X}$. When this equation is solved using a pre-computed Gram matrix, we simply supply the augmented Gram matrix $\mathbf{X}^T\mathbf{X} + \delta\mathbf{I}$. When the Cholesky approach is used, it is straightforward to take an additional parameter δ into account such that the Cholesky factorization of $\mathbf{X}^T\mathbf{X} + \delta\mathbf{I}$ is obtained. Except for these alterations to Step 12, the algorithm is run as usual. The LASSO and the Elastic Net therefore use the same underlying function (`larsen.m`) which take the additional parameter δ used in Step 12. For LASSO solutions δ is simply set to zero.

Zou and Hastie (2005) argue and provide some evidence that the double shrinkage introduced by the l_1 and l_2 has an unfortunate effect on prediction accuracy. They propose to compensate for this by multiplying the solutions $\mathbf{B}$ by a factor $(1+\delta)$, and refer to the unadjusted solutions as the Naïve Elastic Net. In some cases, the naïve solution is preferred, which consequently are obtained either by calling `larsen.m` directly or by dividing the Elastic Net solutions by $(1+\delta)$.

The Elastic Net algorithm outputs the same model selection criteria as the LAR and LASSO algorithms. Computationally, the difference lies in the estimation of the number of degrees of freedom and the residual variance σ_ε^2. For non-zero δ, the corresponding ridge regression solution is used as a low-bias model in the estimation of the latter. Zou (2005) shows that an unbiased estimate of the number of degrees of freedom of Elastic Net solutions can be obtained by

$$\operatorname{tr}\left(\mathbf{X}_{\mathcal{A}}(\mathbf{X}_{\mathcal{A}}^T\mathbf{X}_{\mathcal{A}} + \delta\mathbf{I})^{-1}\mathbf{X}_{\mathcal{A}}^T\right), \tag{36}$$

which we solve efficiently using a singular value decomposition of $\mathbf{X}_{\mathcal{A}}$.

3.4. Sparse principal component analysis

Principal component analysis (PCA) is a linear transformation $\mathbf{S} = \mathbf{XL}$ of a mean-zero data matrix $\mathbf{X}$ where the *loading vectors* (columns) of $\mathbf{L}$ provide an orthonormal basis which successively maximizes the variance of the projected data in $\mathbf{S}$, where the *principal components* (columns) of $\mathbf{S}$ are uncorrelated, see e.g., Hastie, Tibshirani, and Friedman (2009). PCA is optimal in the sense that no linear transformation can produce a more compact representation of data given $K < p$ basis vectors. The successive maximization of variance means that the few first principal components are usually sufficient to accurately describe the data. However, each principal component is a linear combination of *all* variables in $\mathbf{X}$ and is therefore difficult to interpret and assign a meaningful label. To alleviate this, sparse PCA (SPCA) aims at upholding some or all of the properties of PCA — successive maximization of variance, independence of the loading vectors and uncorrelated principal components — while enforcing sparsity of the loading vectors such that each principal component is a linear combination of only a few of the original variables.

The algorithm for computing sparse loading vectors used in this toolbox is detailed in (Zou *et al.* 2006), and uses the Elastic Net in a regression-like framework for PCA. In the spirit of this paper, we start by formulating regular PCA as the solution to a regression problem, and then add suitable constraints to obtain sparse solutions.

Viewing PCA from a compression standpoint, the objective is to find the rank-K subspace projection $\mathbf{A}\mathbf{A}^T$ such that $\mathbf{A}^T\mathbf{A} = \mathbf{I}$ ($\mathbf{A}$ is $p \times K$) which reconstructs a data point $\mathbf{x}$ as well as possible. This amounts to the following criterion,

$$\arg\min_{\mathbf{A}} \|\mathbf{X} - \mathbf{X}\mathbf{A}\mathbf{A}^T\|_F^2, \quad \text{such that} \quad \mathbf{A}^T\mathbf{A} = \mathbf{I}. \tag{37}$$

The solution is readily available via a singular value decomposition; let $\mathbf{X} = \mathbf{U}\mathbf{D}\mathbf{V}^T$, and set $\mathbf{A} = \mathbf{V}$.

Zou *et al.* (2006) show that this criterion can be relaxed into the following l_2-penalized formulation,

$$\arg\min_{\mathbf{A},\mathbf{B}} \|\mathbf{X} - \mathbf{X}\mathbf{A}\mathbf{B}^T\|_F^2 + \delta\|\mathbf{B}\|_F^2, \quad \text{such that} \quad \mathbf{A}^T\mathbf{A} = \mathbf{I}, \tag{38}$$

where $\mathbf{B}$ is $p \times K$ and $\|\mathbf{B}\|_F^2 = \sum_{k=1}^K \|\beta_k\|_2^2$. After normalization such that each column of $\mathbf{B}$ has unit length, the optimal solution is $\mathbf{A} = \mathbf{B} = \mathbf{V}$, the loading matrix of PCA, irrespective of the choice of δ. The role of the ridge penalty on $\mathbf{B}$ is to provide unique solutions also when $p > n$, in which case δ must be non-zero. Since the loading vectors in $\mathbf{B}$ are orthogonal, we can estimate them sequentially by

$$\arg\min_{\alpha_k,\beta_k} \|\mathbf{X} - \mathbf{X}\beta_k\alpha_k^T\|_F^2 + \delta\|\beta_k\|_2^2, \quad \text{subject to} \quad \mathbf{A}_k^T\mathbf{A}_k = \mathbf{I}, \tag{39}$$

where $\mathbf{A}_k$ denotes the matrix $[\alpha_1 \dots \alpha_k]$. Aiming at an algorithm for computing a sparse matrix of loadings, we will now state an alternating algorithm for optimizing the above criterion for α_k and β_k. For this purpose, we have the following result.

Lemma 3.1 *Assume $\alpha_k^T\alpha_k = 1$ and fix α_k, $\mathbf{X}$ and $\mathbf{Y}$. Then, the problems*

$$\arg\min_{\beta_k} \|\mathbf{Y} - \mathbf{X}\beta_k\alpha_k^T\|_F^2 \tag{40}$$

$$\arg\min_{\beta_k} \|\mathbf{Y}\alpha_k - \mathbf{X}\beta_k\|_2^2 \tag{41}$$

$$\tag{42}$$

have the same minimizer $\hat{\beta}_k = (\mathbf{X}^T\mathbf{X})^{-1}\mathbf{X}^T\mathbf{Y}\alpha_k$.

This result (with $\mathbf{Y} = \mathbf{X}$ and adding the l_2 penalty) shows that for fixed α_k, the optimal β_k is given by $\hat{\beta}_k = (\mathbf{X}^T\mathbf{X} + \delta\mathbf{I})^{-1}\mathbf{X}^T\mathbf{X}\alpha_k$. If we instead fix β_k, the optimal α_k is given by the following result.

Lemma 3.2 *Let $\mathbf{A}_{(k-1)}$ with $\mathbf{A}_{(k-1)}^T\mathbf{A}_{(k-1)} = \mathbf{I}$ be the $(p \times k - 1)$ matrix containing the first $k - 1$ columns of $\mathbf{A}$. The "fix β_k, solve for α_k"-problem can then be formulated as,*

$$\hat{\alpha}_k = \arg\min_{\alpha_k} \|\mathbf{X} - \mathbf{X}\beta_k\alpha_k^T\|_F^2 \quad \textit{subject to} \quad \alpha_k^T\alpha_k = 1, \; \alpha_k^T\mathbf{A}_{(k-1)} = \mathbf{0}. \tag{43}$$

Let $\mathbf{s} = \left(\mathbf{I} - \mathbf{A}_{(k-1)}\mathbf{A}_{(k-1)}^T\right)\mathbf{X}^T\mathbf{X}\beta_k$. Then, $\hat{\alpha}_k = \mathbf{s}/\sqrt{\mathbf{s}^T\mathbf{s}}$.

Appendix B.1 and B.2 contain proofs of the above results. If applied alternately until convergence for each principal component, we end up with the full PCA solution. This convergence is assured since Criterion 39 is convex and each alternating step lead to a lower function value.

Turning to the problem of estimating sparse principal components (a sparse loading matrix), an l_1 penalty is added to the formulation in Equation 39.

$$\{\hat{\alpha}_k, \hat{\beta}_k\} = \arg\min_{\alpha_k, \beta_k} \|\mathbf{X} - \mathbf{X}\beta_k\alpha_k^T\|_F^2 + \delta\|\beta_k\|_2^2 + \lambda\|\beta_k\|_1, \quad \text{subject to} \quad \mathbf{A}_k^T\mathbf{A}_k = \mathbf{I}. \tag{44}$$

Using the alternating approach defined above to optimize this criterion, we see that $\hat{\alpha}_k$ is estimated as before, while $\hat{\beta}_k$ is turned from a ridge regression problem into an elastic net problem. As before the response vector is $\mathbf{X}\alpha_k$. We arrive at Algorithm 4 for computing sparse principal components. This algorithm also handles the case where the l_2 regularization parameter δ is set to infinity. The elastic net estimation of β_k then turns into a soft-thresholding rule as described in (Zou *et al.* 2006). This leads to a computational advantage, which is why this option is popular for very high-dimensional data arising from e.g., image or gene expression data. It is our experience that this option also provides better solutions (in terms of explained variance for a fixed level of sparsity) in such cases.

Algorithm 4 SPCA (Zou *et al.* 2006)

```
1: Let $K < p$ be the number of sparse principal loading vectors to estimate
2: Let $\mathbf{A}$ be the $(p \times K)$ matrix consisting of the $K$ first ordinary principal loading vectors
3: for $k = 1, \ldots, K$ do
4:   while sparse loading vector $\beta_k$ has not converged do
5:     if $\delta = \infty$ then
6:       $\beta_k = \left(|\mathbf{X}^T\mathbf{X}\alpha_k| - \lambda\right)_+ \operatorname{sign}(\mathbf{X}^T\mathbf{X}\alpha_k)$ (Soft thresholding)
7:     else
8:       Solve the elastic net problem $\beta_k = \arg\min_\beta \|\mathbf{X}\alpha_k - \mathbf{X}\beta\|^2 + \delta\|\beta\|^2 + \lambda\|\beta\|_1$
9:     end if
10:    $\beta_k = \beta_k / \sqrt{\beta_k^T\beta_k}$ (Normalize to unit length)
11:    $\alpha_k = (\mathbf{I} - \mathbf{A}_{(k-1)}\mathbf{A}_{(k-1)}^T)\mathbf{X}^T\mathbf{X}\beta_k$ (Update $k$th column of projection matrix $\mathbf{A}$)
12:    $\alpha_k = \alpha_k / \sqrt{\alpha_k^T\alpha_k}$ (Normalize to unit length)
13:  end while
14: end for
15: Output the coefficients $\mathbf{B} = [\beta_1 \ldots \beta_K]$
```

Note that this algorithm is no longer convex, and may converge to local minima.

Since this is the first account of this sequential SPCA algorithm we give preliminary results of its performance and discuss advantages in relation to the previously proposed simultaneous approach (Zou *et al.* 2006).

A clear advantage of sequential estimation of components comes from running the algorithm once to estimate k components, and once to estimate $k + l$ components. The sequential approach will yield the exact same first k components in both cases whereas the simultaneous algorithm gives different results for all components.

Both algorithms are initialized with a matrix $\mathbf{A}$ equal to the loading matrix of regular PCA. The corresponding scores are ordered from high to low variance. The simultaneous approach often stray far from this initial solution and yields an arbitrary ordering in terms of variance of its components. In (Sjöstrand, Stegmann, and Larsen 2006) we discuss several ways of establishing a sensible ordering of oblique components. The sequential algorithm is more likely to produce components of decreasing variance, and we have therefore chosen to return the components as-is, in order of computation.

The sequential approach transforms one large non-convex optimization problem into several small. Convergence rates for each such problem are typically orders of magnitude higher than that of the simultaneous approach. To verify this, we conducted an experiment on a synthetic data set of 600 observations, created from 200 observations each of three sparse components with added Gaussian noise. The total number of variables in the data set ranged from 10 to 1500 with increments of 10, and we ran the sequential and simultaneous algorithms once for each choice of p. We extracted three sparse principal components and compared computation times and total adjusted variance. Figure 1 shows the results. The simultaneous algorithm was forced to give up after 1000 iterations, which occurred in a large proportion of runs. This is visible in the figure as a marked line of maximal computation times. Computation times for the sequential algorithm were lower by a factor 15–100 and with no premature terminations. The sequential algorithm is more restrictive than its sequential coouterpart since previous components are fixed when estimating the next component. In our experience, one pays a small price in terms of variance for this restriction; this is shown in the right plot in Figure 1. We have not yet encountered a case were this reduction is significant.

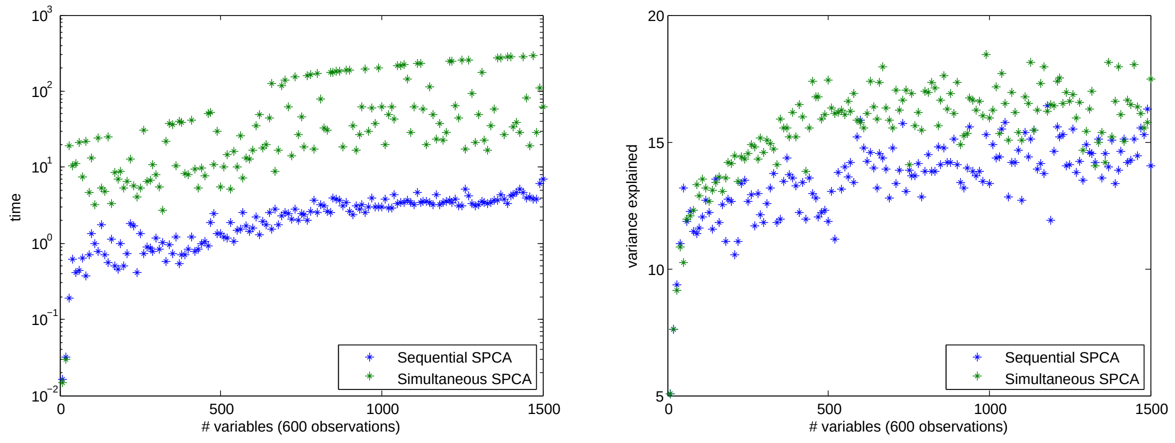

Figure 1: The left figure shows computation time for a data set with 600 observations and increasing dimensionality; 25 non-zero loadings were extracted. The sequential SPCA algorithm is faster by a factor 15–100. The right figure shows total adjusted variance for three components for the same data set. The sequential algorithm pays a small penalty for the one component at a time approach.

3.5. Sparse linear discriminant analysis

Linear Discriminant Analysis (LDA) estimates orthogonal directions β_k in which observations $\mathbf{x}_i$ belonging to one of K classes are most separated. Separation is measured as the between-class variance σ_b^2 in relation to the within-class variance σ_w^2 of the projected data $\mathbf{X}\beta_k$. Class-belongings are dummy-encoded in a $(n \times K)$ matrix $\mathbf{Y}$ where element (i, j) is 1 if the ith observation belongs to the jth class, else 0. Further, the matrix $\mathbf{D}_\pi = \frac{1}{n}\mathbf{Y}^T\mathbf{Y}$ is a diagonal matrix of class prior probabilities based on their frequency in $\mathbf{Y}$. Given these definitions, the matrix of class centroids is given by $\mathbf{M} = \frac{1}{n}\mathbf{D}_\pi^{-1}\mathbf{Y}^T\mathbf{X}$, the total covariance matrix is $\Sigma = \frac{1}{n}\mathbf{X}^T\mathbf{X}$, the between-class covariance matrix is $\Sigma_b = \mathbf{M}^T\mathbf{D}_\pi\mathbf{M} = \frac{1}{n}\mathbf{X}^T\mathbf{Y}(\mathbf{Y}^T\mathbf{Y})^{-1}\mathbf{Y}^T\mathbf{X}$, and the within-class covariance matrix is $\Sigma_w = \Sigma - \Sigma_b = \frac{1}{n}\mathbf{X}^T(\mathbf{I} - \mathbf{Y}(\mathbf{Y}^T\mathbf{Y})^{-1}\mathbf{Y}^T)\mathbf{X}$. The cost function to optimize for the kth direction is,

$$\arg\max_{\beta_k} \beta_k^T\Sigma_b\beta_k \quad \text{subject to} \quad \beta_k^T\Sigma_w\beta_k = 1, \beta_k^T\Sigma_w\beta_l = 0, \forall l < k. \tag{45}$$

Standard differentiation leads to a an eigenvalue problem with respect to the matrix $\Sigma_w^{-1}\Sigma_b$ which yields the full set of solutions $\mathbf{B} = \{\beta_k\}$. Classification of a new observation $\mathbf{x}$ is performed by finding the closest centroid in the derived space defined by $\mathbf{B}$.

LDA relies on the assumptions that (1) the data is normally distributed and (2) all classes have equal covariances. Although these assumptions are seldom met exactly, LDA often has as good or better performance compared to more flexible alternatives. This is in part due to the robustness of a method with few parameters to estimate. As with ordinary least squares in regression, there are situations where the inverse of the Gram matrix $\mathbf{X}^T\mathbf{X}$ — which turns up in the estimation of regression coefficients as well as the estimation of Σ_w^{-1} — has high variance or is computationally infeasible. Similarly to ridge regression (cf., Section 3.3) the covariance matrix (or equivalently, the Gram matrix) may be replaced by a regularized variant $\Sigma + \delta\Omega$. If Ω is a positive definite matrix, then there exists a large-enough positive value of δ such that $\Sigma + \delta\Omega$ is positive definite (Petersen and Pedersen 2008). Application of this approach to LDA leads to *penalized* (linear) discriminant analysis (PDA) (Hastie, Buja, and Tibshirani 1995); the estimate of Σ_w is simply replaced by $\Sigma_w + \delta\Omega$, otherwise the calculation and application proceeds as before. In the remainder of this section, we will use $\Omega = \mathbf{I}$ which shrinks the solutions towards those obtained by assuming a spherical common covariance matrix.

An alternative route to the solutions $\mathbf{B}$ of LDA/PDA is via *optimal scoring* (Hastie, Tibshirani, and Buja 1994). The PDA optimal scoring criterion is

$$\arg\min_{\Theta,\mathbf{B}} \|\mathbf{Y}\Theta - \mathbf{X}\mathbf{B}\|_F^2 + \delta\|\mathbf{B}\|_F^2 \quad \text{subject to} \quad \Theta^T\mathbf{D}_\pi\Theta = \mathbf{I}. \tag{46}$$

The matrix Θ is a scoring matrix, orthogonal in $\mathbf{D}_\pi$, which assigns a multiple to each column (class) in $\mathbf{Y}$. This transformation of the dummy encodings circumvents problematic situations which otherwise make a regression approach to classification difficult (Hastie *et al.* 2009). As shown in detail in (Hastie *et al.* 1995) and more succinctly in (Hastie *et al.* 1994), the optimal $\mathbf{B}$ are equivalent, up to a diagonal scaling matrix, to those obtained by PDA using the penalized within-class covariance matrix $\Sigma_w + \frac{\delta}{n}\mathbf{I}$. Differentiation, first with respect to $\mathbf{B}$ and then with respect to Θ gives the standard solution; first compute a multivariate ridge regression of $\mathbf{X}$ on $\mathbf{Y}$ yielding regression coefficients $\tilde{\mathbf{B}} = (\mathbf{X}^T\mathbf{X} + \delta\mathbf{I})^{-1}\mathbf{X}^T\mathbf{Y}$. Θ is then obtained by an eigenanalysis of the matrix $\hat{\mathbf{Y}}\mathbf{Y}$, where $\hat{\mathbf{Y}} = \mathbf{X}\tilde{\mathbf{B}}$. The final directions $\mathbf{B}$ are finally obtained by $\mathbf{B} = \tilde{\mathbf{B}}\Theta$.

Similarly to the treatment of the PCA Criterion 39, we can estimate the directions β_k sequentially since they are orthogonal. The estimation of the kth direction involves solving

$$\arg\min_{\theta_k,\beta_k} \|\mathbf{Y}\theta_k - \mathbf{X}\beta_k\|_2^2 + \delta\|\beta_k\|_2^2 \quad \text{subject to} \quad \Theta_k^T\mathbf{D}_\pi\Theta_k = \mathbf{I}, \tag{47}$$

where Θ_k contains the k first columns of Θ. We will now describe an alternating algorithm which replaces the eigenanalysis-based recipe in the previous paragraph. This algorithm then naturally extends to *sparse* discriminant analysis, where the ridge regression estimate is replaced by an elastic net estimate. In line with Section 3.4, we first state the following Lemma.

Lemma 3.3 *Let $\Theta_{(k-1)}$ with $\Theta_{(k-1)}^T\mathbf{D}_\pi\Theta_{(k-1)} = \mathbf{I}$ be the $(p \times k-1)$ matrix containing the first $k-1$ columns of Θ. The "fix β_k, solve for θ_k"-problem can then be formulated as,*

$$\hat{\theta}_k = \arg\min_{\theta_k} \|\mathbf{Y}\theta_k - \mathbf{X}\beta_k\|_2^2 \quad \textit{subject to} \quad \theta_k^T\mathbf{D}_\pi\theta_k = 1,\ \theta_k^T\mathbf{D}_\pi\Theta_{(k-1)} = \mathbf{0}. \tag{48}$$

Let $\mathbf{s} = \left(\mathbf{I} - \Theta_{(k-1)}\Theta_{(k-1)}^T\mathbf{D}_\pi\right)\mathbf{D}_\pi^{-1}\mathbf{Y}^T\mathbf{X}\beta_k$. Then, $\hat{\alpha}_k = \mathbf{s}/\sqrt{\mathbf{s}^T\mathbf{D}_\pi\mathbf{s}}$.

Proof Appendix B.2 gives a proof for Lemma 3.2; the proof for this lemma is equivalent, except the trivial addition of $\mathbf{D}_\pi$. The solution to this Lemma is also given — sans proof — in (Clemmensen *et al.* 2011).

The "fix θ_k, solve for β_k" problem is solved by the ridge regression estimate $\beta_k = (\mathbf{X}^T\mathbf{X} + \delta\mathbf{I})^{-1}\mathbf{X}^T\mathbf{Y}\theta_k$.

We initialize Θ to the size $(K \times K)$ identity matrix. The directions are then obtained sequentially by alternating the estimation of β_k and θ_k until convergence. Convergence to a global optimum for each direction is guaranteed by the convexity of the cost function and its constraints and since each alteration is sure to lower the cost.

With this algorithm in place, it is now straight-forward to extend it to include an l_1 penalty which promotes directions β_k which are sparse. This means that the space $\mathbf{B}$ in which the classification is carried out, consists of a subset of the available variables. As with all sparse methods, the possible benefits are ease of interpretation and (non-linear) suppression of noise. The l_1, l_2-regularized criterion is,

$$\{\hat{\theta}_k, \hat{\beta}_k\} = \arg\min_{\theta_k,\beta_k} \|\mathbf{Y}\theta_k - \mathbf{X}\beta_k\|_2^2 + \delta\|\beta_k\|_2^2 + \delta_k\|\beta_k\|_1 \quad \text{subject to} \quad \Theta_k^T\mathbf{D}_\pi\Theta_k = \mathbf{I}, \tag{49}$$

which turns the ridge regression estimation of β_k from Criterion 47 into an Elastic Net estimate. The resulting algorithm for sparse discriminant analysis is stated in Algorithm 5.

The normalization in Step 7 helps to avoid multiplicative drift towards the trivial solution where $\theta_k = \mathbf{0}$ and $\beta_k = \mathbf{0}$. To avoid additive drift we also require $\sum_i(\mathbf{D}_\pi\theta_k)_i = 0$, i.e., that θ_k is zero-mean in $\mathbf{D}_\pi$. In previous treatments of optimal scoring (see e.g., Clemmensen *et al.* (2011)), the columns of Θ are explicitly forced to be orthogonal to a vector of ones. However, it turns out that Step 6 implicitly guarantees this since $\mathbf{D}_\pi\theta_k = \mathbf{Y}^T\mathbf{X}\beta_k - \mathbf{D}_\pi\Theta_{(k-1)}\Theta_{(k-1)}^T\mathbf{D}_\pi\mathbf{D}_\pi^{-1}\mathbf{Y}^T\mathbf{X}\beta_k$. The first term is clearly mean-zero since $\mathbf{X}$ is centered. The second term is mean-zero if $\mathbf{D}_\pi\Theta_{(k-1)}\Theta_{(k-1)}^T\mathbf{D}_\pi\mathbf{D}_\pi^{-1}$ is mean zero. $\mathbf{D}_\pi\Theta_{(k-1)}\Theta_{(k-1)}^T\mathbf{D}_\pi$ is the

Algorithm 5 SLDA (Clemmensen *et al.* 2011)

```
1: Let $Q$ be the number of classes and $K < Q$ the number of discriminative directions
2: Initialize $\mathbf{Y}$ an $n \times Q$ matrix of indicator variables with $Y_{i \in \text{class}_j j} = 1$, $\mathbf{D}_\pi = \frac{1}{n}\mathbf{Y}^T\mathbf{Y}$, and
   $\Theta = \mathbf{I}_{(Q \times K)}$
3: for $k = 1, \ldots, K$ do
4:   while sparse discriminative direction $\beta_k$ has not converged do
5:     Solve the elastic net problem $\beta_k = \arg\min_\beta \|\mathbf{Y}\theta_k - \mathbf{X}\beta\|^2 + \delta\|\beta\|^2 + \delta\|\beta\|_1$
6:     $\theta_k = (\mathbf{I} - \Theta_{(k-1)}\Theta_{(k-1)}^T\mathbf{D}_\pi)\mathbf{D}_\pi^{-1}\mathbf{Y}^T\mathbf{X}\beta_k$ (Update $k$th column of $\Theta$)
7:     $\theta_k = \theta_k / \sqrt{\theta_k^T\mathbf{D}_\pi\theta_k}$ (Normalize to unit length)
8:   end while
9: end for
10: Output the coefficients $\mathbf{B} = [\beta_1 \ldots \beta_K]$
```

outer product of two mean-zero matrices and results in a mean-zero matrix. Multiplying this with the diagonal matrix $\mathbf{D}_\pi^{-1}$ does not change this property.

To allow for a more flexible model of the density of each class one may model each class a mixture of Gaussian distributions. Clemmensen *et al.* (2011) describe an extension of the described algorithm which implements this.

4. Collaboration and verification

We use tools and principles from software engineering in the development of this toolbox. A server-based repository (Apache Subversion (SVN) (Apache Software Foundation 2011)) allows toolbox authors to download (Update in SVN terms) the latest toolbox snapshot, apply changes and then upload (Commit) to the server when finished. Simultaneous editing of files is also possible where overlapping changes are merged in an intuitive way when committing changes back to the repository.

In software engineering *continuous integration* (Wikipedia 2011a) refers to the practice of committing small changes often to the repository, rather than scarce large updates. This keeps the effort required to merge changes from different authors to a minimum. To further improve the quality and effectiveness of the development, we employ *unit testing* (Wikipedia 2011b). In parallel with the development of each toolbox entity, we develop several test scripts, each testing the one part (unit) of the code. As an example, one test file asserts (using the `Matlab` `assert` command) that the elastic net with $\lambda = \delta = 0$ equals the ordinary least squares solution. Although we currently have no automated procedure, we aim to run all such unit test files each time changes are uploaded to the repository. In this way, we get a strong indication to whether the new code is working as expected, and that uploaded changes did not break code that was working in an earlier version of the toolbox. Tables 1 and 2 list all relevant unit tests.

Other means of verification we have used are *code walkthrough*, a line-by-line inspection of finished code, and deployment of beta releases of the toolbox.

A. Sparse regression with orthogonal predictors

Several methods have simple closed-form solutions in cases where the predictor variables are orthogonal and have Euclidean length 1. Often, the estimation can be split into p separate problems, one for each β_i. We will quickly review how this works for the LASSO, the treatment is similar for the Elastic Net and Least Angle Regression. We use this property for testing purposes, cf., Table 1.

$$\|\mathbf{y} - \mathbf{X}\beta\|^2 + \lambda\|\beta\|_1 \iff \|\mathbf{y} - \mathbf{x}_i\beta_i\|^2 + \lambda|\beta_i|, \forall i. \tag{50}$$

Optimizing the expression for a single $\hat{\beta}_i$. involves taking first derivatives and setting to zero,

$$-2\mathbf{x}_i^T(\mathbf{y} - \mathbf{x}_i\beta_i) + \lambda \cdot \operatorname{sign}(\beta_i) = 0, i \in \mathcal{A}. \tag{51}$$

Using $\mathbf{x}_i^T\mathbf{y} = \beta_i^{OLS}$ and $\mathbf{x}_i^T\mathbf{x}_i = 1$ we have,

$$-2\beta_i^{OLS} + 2\beta_i + \lambda \cdot \operatorname{sign}(\beta_i) = 0, i \in \mathcal{A}. \tag{52}$$

For sufficiently large values of λ, β_i will shrink to exactly zero. For any other value of λ, β_i will agree in sign with β_i^{OLS}. Therefore, we have,

$$\beta_i = \operatorname{sign}(\beta_i^{OLS})\left(|\beta_i^{OLS}| - \frac{\lambda}{2}\right)^+, \forall i, \tag{53}$$

where $(\cdot)^+$ denotes the hinge function $\max(\cdot, 0)$.

B. Proofs

B.1. Proof of Lemma 3.1

Using $\operatorname{tr}(\mathbf{AB}) = \operatorname{tr}(\mathbf{BA})$, $\operatorname{tr}(\mathbf{A}+\mathbf{B}) = \operatorname{tr}(\mathbf{A}) + \operatorname{tr}(\mathbf{B})$ and $\alpha_k^T\alpha_k = 1$ we have,

$$\|\mathbf{X} - \mathbf{X}\beta_k\alpha_k^T\|_F^2 = \tag{54}$$

$$\operatorname{tr}\left(\mathbf{X}^T\mathbf{X} + \alpha_k\beta_k^T\mathbf{X}^T\mathbf{X}\beta_k\alpha_k^T - 2\mathbf{X}^T\mathbf{X}\beta_k\alpha_k^T\right) = \tag{55}$$

$$\operatorname{tr}(\mathbf{X}^T\mathbf{X}) + \operatorname{tr}(\mathbf{X}\beta_k\alpha_k^T\alpha_k\beta_k^T\mathbf{X}^T) - 2\alpha_k^T\mathbf{X}^T\mathbf{X}\beta_k = \tag{56}$$

$$\operatorname{tr}(\mathbf{X}^T\mathbf{X}) + \operatorname{tr}(\mathbf{X}\beta_k\beta_k^T\mathbf{X}^T) - 2\alpha_k^T\mathbf{X}^T\mathbf{X}\beta_k = \tag{57}$$

$$\operatorname{tr}(\mathbf{X}^T\mathbf{X}) + \beta_k^T\mathbf{X}^T\mathbf{X}\beta_k - 2\alpha_k^T\mathbf{X}^T\mathbf{X}\beta_k, \tag{58}$$

which clearly has the same minimizing β_k as

$$\|\mathbf{X}\alpha_k - \mathbf{X}\beta_k\|_2^2 = \tag{59}$$

$$\alpha_k^T\mathbf{X}^T\mathbf{X}\alpha_k + \beta_k^T\mathbf{X}^T\mathbf{X}\beta_k - 2\alpha_k^T\mathbf{X}^T\mathbf{X}\beta_k. \tag{60}$$

Differentiation of any of the expressions gives $\hat{\beta}_k = (\mathbf{X}^T\mathbf{X})^{-1}\mathbf{X}^T(\mathbf{X}\alpha_k)$. This proof is also detailed in a slightly different context by Zou *et al.* (2006).

B.2. Proof of Lemma 3.2

Incorporating the constraints into the cost function in Equation 43 using Lagrange multipliers λ (length $k-1$ vector) and γ (scalar) the problem becomes

$$\arg\min_{\alpha_k} \operatorname{tr}\left[\alpha_k\beta_k^T\mathbf{X}^T\mathbf{X}\beta_k\alpha_k^T - 2\alpha_k\beta_k^T\mathbf{X}^T\mathbf{X} + \mathbf{X}^T\mathbf{X}\right] + \alpha_k^T\mathbf{A}_{(k-1)}\lambda + \gamma(\alpha_k^T\alpha_k - 1).$$

Differentiating and setting to zero, and solving for α_k leads to

$$\hat{\alpha}_k = \frac{1}{\beta_k^T\mathbf{X}^T\mathbf{X}\beta_k + \gamma}\left[\mathbf{X}^T\mathbf{X}\beta_k - \frac{1}{2}\mathbf{A}_{(k-1)}\lambda\right], \tag{61}$$

or equivalently,

$$\hat{\alpha}_k = \frac{1}{\beta}\left[\mathbf{X}^T\mathbf{X}\beta_k - \mathbf{A}_{(k-1)}\alpha\right]. \tag{62}$$

The orthogonality constraints give

$$\begin{gathered}\mathbf{A}_{(k-1)}^T\hat{\alpha}_k = 0 \Leftrightarrow \frac{1}{\beta}\left[\mathbf{A}_{(k-1)}^T\mathbf{X}^T\mathbf{X}\beta_k - \alpha\right] = 0 \Leftrightarrow \\ \alpha = \mathbf{A}_{(k-1)}^T\mathbf{X}^T\mathbf{X}\beta_k.\end{gathered} \tag{63}$$

Inserting this expression for α into Equation 62 and simplifying gives

$$\hat{\alpha}_k = \frac{1}{\beta}\left(\mathbf{I} - \mathbf{A}_{(k-1)}\mathbf{A}_{(k-1)}^T\right)\mathbf{X}^T\mathbf{X}\beta_k \equiv \frac{1}{\beta}\mathbf{s}. \tag{64}$$

Finally, the constraint $\alpha_k^T\alpha_k = 1$ gives $\beta = \sqrt{\mathbf{s}^T\mathbf{s}}$ such that $\hat{\alpha}_k = \mathbf{s}/\sqrt{\mathbf{s}^T\mathbf{s}}$. In practice, we first calculate $\mathbf{s}$ and then normalize this vector to unit length.

References

Apache Software Foundation (2011). "Apache Subversion." http://subversion.apache.org.

Clemmensen L, Hastie T, Witten D, Ersbøll B (2011). "Sparse Discriminant Analysis." *Technometrics*, **53**(4), 406–413.

Efron B, Hastie T, Johnstone I, Tibshirani R (2004). "Least Angle Regression." *The Annals of Statistics*, **32**(2), 407–451.

Friedman JH, Hastie T, Tibshirani R (2010). "Regularization Paths for Generalized Linear Models via Coordinate Descent." *Journal of Statistical Software*, **33**(1), 1–22. ISSN 1548-7660. URL http://www.jstatsoft.org/v33/i01.

Fu WJ (1998). "Penalized Regressions: The Bridge Versus the Lasso." *Journal of Computational and Graphical Statistics*, **7**(3), 397.

Hastie T, Buja A, Tibshirani R (1995). "Penalized Discriminant Analysis." *The Annals of Statistics*, **23**, 73–102.

Hastie T, Rosset S, Tibshirani R, Zhu J (2004). "The Entire Regularization Path for the Support Vector Machine." *Journal of Machine Learning Research*, **5**, 1391–1415.

Hastie T, Tibshirani R, Buja A (1994). "Flexible Discriminant Analysis by Optimal Scoring." *Journal of the American Statistical Association*, **89**(428), 1255–1270.

Hastie T, Tibshirani R, Friedman J (2009). *The Elements of Statistical Learning: Data Mining, Inference, and Prediction.* Springer-Verlag.

Hoerl AE, Kennard RW (1970). "Ridge regression: Biased Estimation from Nonorthogonal Problems." *Technometrics*, **12**(1), 55–67. ISSN 00401706.

Osborne MR, Presnell B, Turlach BA (2000). "A new Approach to Variable Selection in Least Squares Problems." *IMA Journal of Numerical Analysis*, **20**(3), 389–403.

Park MY, Hastie T (2007). "L1 Regularization Path Algorithm for Generalized Linear Models." *Journal of the Royal Statistical Society B*, **69**(4), 659–677.

Petersen KB, Pedersen MS (2008). "The Matrix Cookbook." *Technical report*, Technical University of Denmark.

Rosset S, Zhu J (2007). "Piecewise Linear Regularized Solution Paths." *The Annals of Statistics*, **35**(3), 1012–1030.

Sjöstrand K, Stegmann MB, Larsen R (2006). "Sparse Principal Component Analysis in Medical Shape Modeling." volume 6144. SPIE.

The MathWorks Inc (2010). "Matlab version 7.11.0.584 (R2010b), 64-bit."

Tibshirani R (1996). "Regression Shrinkage and Selection via the LASSO." *Journal of the Royal Statistical Society B*, **58**(1), 267–288.

Wikipedia (2011a). "Continous Integration." http://en.wikipedia.org/wiki/Continuous_integration.

Wikipedia (2011b). "Unit Testing." http://en.wikipedia.org/wiki/Unit_testing.

Zou H (2005). *Some Perspectives of Sparse Statistical Modeling.* Ph.D. thesis, Stanford University.

Zou H, Hastie T (2005). "Regularization and Variable Selection via the Elastic Net." *Journal of the Royal Statistical Society B*, **67**(2), 301–320.

Zou H, Hastie T, Tibshirani R (2006). "Sparse Principal Component Analysis." *Journal of Computational and Graphical Statistics*, **15**(2), 265.

Zou H, Hastie T, Tibshirani R (2007). "On the "Degrees of Freedom" of the Lasso." *The Annals of Statistics*, **35**(5), 2173–2192.

Affiliation:

Karl Sjöstrand
Department of Informatics and Mathematical Modelling
Technical University of Denmark
DK-2800 Lyngby, Denmark
E-mail: `kas@imm.dtu.dk`
URL: `http://www.imm.dtu.dk/projects/spasm/`

Journal of Statistical Software `http://www.jstatsoft.org/`
published by the American Statistical Association `http://www.amstat.org/`

Volume VV, Issue II *Submitted:* yyyy-mm-dd
MMMMMM YYYY *Accepted:* yyyy-mm-dd

Unit	Test	Acceptance Criterion
LAR	Make sure the full LAR model is equal to the ordinary least squares model	Results are equal
LAR	Make sure LAR and LASSO are equal in cases where no variables are dropped in the LASSO	Results are equal
LAR	Profile code on $n \gg p$ and $p \gg n$ data sets.	Code has no apparent bottlenecks
LASSO	Make sure the full LASSO model is equal to the ordinary least squares model	Results are equal
LASSO	Run with a data set with orthogonal predictor variables. Compare to soft thresholding. Cf., Appendix A	Results are equal
LASSO	Profile code on $n \gg p$ and $p \gg n$ data sets.	Code has no apparent bottlenecks
Elastic Net	Make sure the full Elastic Net model is equal to the corresponding Ridge Regression model.	Results are equal
Elastic Net	Run with a data set with orthogonal predictor variables. Compare to soft thresholding.	Results are equal
Elastic Net	Compare results with running LASSO with an Elastic Net-style augmented data matrix	Results are equal
Elastic Net	Profile code on $n \gg p$ and $p \gg n$ data sets.	Code has no apparent bottlenecks
SPCA	Compare the full SCPA model with that of a regular PCA. Try different values of δ.	Results are equal regardless of the value of δ
SPCA	Profile code on $n \gg p$ and $p \gg n$ data sets.	Code has no apparent bottlenecks
SLDA	Assert that the resulting optimal scores $\mathbf{Z} = \mathbf{Y}\theta$ are orthogonal	$\mathbf{Z}^T\mathbf{Z}/n = \mathbf{I}$
SLDA	Compare the results of SLDA with no l_1 constraint to ridge regression on the matrix $\mathbf{Y}\theta$	Results are equal

Table 1: Toolbox unit tests (part 1)

UNIT	TEST	ACCEPTANCE CRITERION
SLDA	Compare the results of SLDA with no l_1 constraint to penalized discriminant analysis (LDA using the within-class covariance matrix $\Sigma_W + \frac{\delta}{n}\mathbf{I}$	Results are equal
SLDA	Profile code on $n \gg p$ and $p \gg n$ data sets.	Code has no apparent bottlenecks
cholinsert	Compare updates of the Cholesky factorization to a direct Cholesky factorization of the corresponding matrices	Results are equal
choldelete	Compare downdates of the Cholesky factorization to a direct Cholesky factorization of the corresponding matrices	Results are equal

Table 2: Toolbox unit tests (part 2)