Sparse optimization models with robust sketching and applications

Vu Pham

Sparse optimization models with robust sketching and applications

by

Vu Pham

A dissertation submitted in partial satisfaction of the
requirements for the degree of
Doctor of Philosophy

in

Engineering – Electrical Engineering and Computer Sciences

in the

Graduate Division
of the
University of California, Berkeley

Committee in charge:

Professor Laurent El Ghaoui, Chair
Professor Marti Hearst
Professor Ming Gu

Fall 2016

Sparse optimization models with robust sketching and applications

Abstract

Sparse optimization models with robust sketching and applications

by

Vu Pham

Doctor of Philosophy in Engineering – Electrical Engineering and Computer Sciences

University of California, Berkeley

Professor Laurent El Ghaoui, Chair

Sparse machine learning has recently emerged as powerful tool to obtain models of high-dimensional data with high degree of interpretability, at low computational cost. The approach has been successfully used in many areas, such as signal and image processing. In sparse learning classification, for example, the prediction accuracy or some other classical measure of performance is not the sole concern: we also wish to be able to better understand which few features are relevant as markers for classification. Furthermore, many of sparse learning tasks in practice, including cross-validation, parameter search, or leave-one-out analysis, involve multiple instances of similar problems, each instance sharing a large part of learning data with the others. In this thesis, we introduce a robust framework for solving these multiple sparse regressions in the form of square-root LASSO problems, based on a sketch of the learning data that uses low-rank approximations. Our approach allows a dramatic reduction in computational effort, while not sacrificing—sometimes even improving—the statistical performance.

We present our technique by first studying sparse optimization with applications in different domain of interests, from text analytics to system design, and then developing theories for robust solutions for sparse regression in multi-instance setting. We also provide comparisons with other heuristics to obtain sparse models in various applications. In more detail, our central contributions from this thesis include:

- Identifying key tasks in domains of interests under real-world setting,
- Suggesting models that are suitable for these tasks along the axes of computational complexity and model understandability,
- Exploiting problem structures when working with multiple instances to robustly improve computation while maintaining high learning performance, and
- Proposing applications of our robust solutions in high-dimensional setting.

To my family

Contents

List of Figures

List of Tables

Acknowledgments

I owe credit to so many people for this thesis and I would like to apologize if I forget to mention some of you in my acknowledgement. If I do, please forgive me and I would be very happy to invite you out for dinner.

First of all, I'm greatly indebted to my doctoral advisor at Berkeley, Prof. Laurent El Ghaoui, for supporting me throughout graduate study. More importantly, Prof. El Ghaoui has put me into contact with many areas, given me the opportunity to work with many world-leading scholars, and constantly been my role model as an example of excellence and professionalism in academia and mentorship. Without my doctoral advisor, I would not be able to explore diverse directions in my work, and pursue the topic I'm most interested in. I always remember how my advisor has always made his invaluable time to meet with me every week throughout many years of my PhD. Together with Prof. El Ghaoui, there are a number of other faculty who have greatly supported me in my PhD study. I would like to express my most sincere gratitude to Prof. Ming Gu, Prof. Marti Hearst, and Prof. Michael Mahoney, who have given me extremely valuable feedback to my results and suggested possible future directions for me to pursue.

Berkeley has been an amazing environment for me since the first day of my PhD. I would like to thank Prof. Peter Bickel, Prof. Bin Yu, Prof. Ashok Srivastava, Prof. Vu Duong, and Prof. Abigail De Kosnik for your great guidance, even before I joined Berkeley for graduate school. Without their support, it would not be possible for me to have the courage to follow this journey. I would like to express my deepest appreciation to them and other amazing friends from the focus research group, including Sharmodeep Bhattacharyya, Fu Shi, Siqi Wu, and many more. Additionally, I've been very fortunate to be surrounded by wonderful peers and colleagues during my PhD in EECS and Statistics department: Mert Pilanci, Andrew Godbehere, Jerome Thai, Walid Krichene, Arturo Fernandez, Guan-Cheng Li. Moreover, during time at Cal, I've also had the best chance to collaborate with many great graduate and undergraduate students, some of whom have moved on to pursue graduate study and doctoral research in other institutions, and some have started as young Professors in top-tier research universities.

No words can describe how much I fall in love with the culture at Cal, and most importantly, the people of Berkeley. I'm fortunate to take part in amazing groups at Cal; I cannot forget the time with the incredible Culture show team, playing board games with housemates, and practicing sports with teammates at the RSF. It is thanks to diverse groups at Berkeley where I learn to appreciate the beauty of other disciplines, such as economics, physics, and management. You all are the amazing people whom I've loved being with and whom I've always wanted to spend great time altogether.

Last but not least, I cannot stress enough how important my parents have been to me during graduate school. Thank you Mom for encouraging me to pursue the best education I can possibly get, and always reminding me to focus on the fundamentals. Thank you Dad who always asks me to be myself and pursue what I'm truly passionate about. Without your support, I would not have the courage to do what I love, and I could

not be the person who I am today. I always remember every moment since my childhood with you, and I would like to dedicate this thesis to you. Wish you both good health and great happiness from my loving heart.

Chapter 1

Introduction

Modern techniques in machine learning are essential to obtain models in the context of big data and have been successfully used in many areas, such as artificial intelligence systems, signal and image processing [40, 73]. Sparse machine learning has recently emerged as a powerful tool to identify models of high-dimensional setting with a high degree of interpretability, at low computational cost. Specifically, sparse learning seeks a trade-off between a model's accuracy measure and sparsity of its result, which in turn provides better insight into important building blocks of the model. An example is in binary classification of data with genes as features; a researcher may wish to not only obtain a high-precision classifier but also one that involves only a few genes, allowing biologists to pivot future research efforts on these specific factors. There is an extensive body of literature on the topic of sparse learning, with terms such as compressed sensing or l_1-norm penalty [5, 18, 31]. In this thesis, we will study sparse learning under multi-instance setting with perspectives from optimization theory. Broadly, we identify sparse optimization models that, in addition to traditional probabilistic graphical models, provide insights to experts on domains of interest at very low computational cost. In particular, we consider engineering designs and text analytics tasks along the axes of model computation and interpretability. One may ask why one needs to develop multi-instance approaches with sparsity as an emphasis, and whether we can just simply employ traditional machine learning methods and enforce sparsity as one last step? We will develop our response to this alternative approach with practical examples throughout this thesis.

1.1 Goals and contributions

The focus of this thesis is to propose, via examples in engineering, machine learning, and data analytics, approaches for sparse optimization with modest computation and high sparsity for model understanding. By identifying key tasks from real-life case studies, we exemplify our approach with real data sets such as energy profiles and safety reports to evaluate the potential of the models in practice. Most importantly, via real-life examples, we observe the multi-instance setting when working with large datasets, and propose our method to exploit this structure. Based on these observation, we derive robust solutions with fast algorithms without sacrificing statistical learning performance, while keeping the problem structure simple. In particular, the goals of this thesis are to:

1. Motivate sparse optimization study with applications in different domain of interests, from system design to text analytics, and

2. Develop theories for robust solutions for sparse regression in multi-instance setting, where data are often shared between many instances.

More specifically, the important contributions of the thesis are the following:

- We identify key tasks in these domains of interest under real-world setting,

- We employ models that are suitable for these tasks along the axes of computational complexity and model understandability,

- We exploit problem structure when working with many instances to improve computation while maintaining high learning performance,

- We discuss applications of our robust solutions in high-dimensional settings.

A common theme in our presentation is our emphasis on sparsity. We posit that by pivoting our model on sparsity, we do not make the optimization algorithm harder, but in fact, more tractable in the development of the model and more useful to experts with the end-result. We also provide comparisons with other heuristic methods which enforce sparsity as the final step of the computation process. Additionally we discuss safe feature elimination techniques or screening rules for sparse optimization under very high dimensional settings. In short, this thesis motivates sparsity in practical applications and exploitation of problem structures in multi-instance and large-scale learning to improve computational performance that can benefit human experts without requiring excessive resources.

1.2 Organization of the thesis

The results of this thesis are based on joint work of me and collaborators. I'll outline the rest of this thesis while briefly describe our collaborations and previous results in each portion. Chapter 2 of this thesis provides an overview on sparse models, with an example from text analytics tasks. The presentation in this chapter serves as an introduction to the theme of our thesis. We present our case study in Chapter 3, of which our results are obtained from collaborations with Laurent El Ghaoui, Ashok Srivastava, Kanishka Bhaduri, Guan-Cheng Li, and Viet-An Duong [38]. Comparative studies in this chapter were inspired from previous work by Ashok Srivastava and Mehran Sahami [94].

Chapter 4 moves beyond text analytics to enter the area of energy system design. Despite being a complex application in practice, engineering problems often benefit from insights into dynamical systems, aiming to improve the performance, such as to minimize temperature deviation from an ideal range. Our work on sparse models is based on collaboration with a Fortune 500 company, Électricité de France (EDF), whose R&D departments are active in the area of model predictive approaches in "energy management". The results from this chapter are also from our joint work with Chris Meissen, Andy Packard, Laurent El Ghaoui, Giuseppe Calafiore, and Carlo Novara [19].

Examples from Chapter 3 and Chapter 4 serve as motivation for a main theme of this thesis: sparse learning often occurs in multiple instances. In Chapter 5, we consider a robust approach to sparse modeling under multi-instance setting. With an observation that many instances in fact share most of the data, we take advantage of a pre-processing step for data approximation ("sketching"), and exploit this structure in all instances. Examples in this chapter tie back to the case studies discussed in previous chapters, which summarize applications of our theoretical approach in practice. The development of this chapter receives many inputs from Laurent El Ghaoui, Mert Pilanci, Vincent Leclère [85]. Our work was inspired from the seminal paper 20 years ago on robust least squares by Laurent El Ghaoui and Hervé Lebret [37]. Lastly, we conclude this thesis with our discussion on notable results in Chapter 6.

1.3 Notation

In this section, we provide our basic notation before proceeding to the rest of the thesis. Our notation is more or less standard; we will clarify our notation in the remaining chapters when needed to avoid confusion.

Space. We use $\mathbf{R}$ to denote the set of real numbers, $\mathbf{R}_+$ to denote the set of nonnegative real numbers, and $\mathbf{R}_{++}$ to denote the set of strictly positive real numbers. The set of real vectors of dimension n is denoted $\mathbf{R}^n$, and the set of real matrices of dimension $m \times n$ matrices is denoted $\mathbf{R}^{m\times n}$, where m is the number of rows and n is the number columns. Note that recently in high-dimensional settings, some authors may choose to denote $\mathbf{R}^{n\times d}$ where n is the number of rows (or observations), and d is the number of features (or dimensions).

Vector. We delimit vectors and matrices with square brackets; all vector notations in this thesis are column vectors, i.e.

$$x := \begin{bmatrix} a \\ b \\ c \end{bmatrix} = \begin{bmatrix} a & b & c \end{bmatrix}^T \in \mathbf{R}^3$$

where $a, b, c \in \mathbf{R}$. The special symbol $\mathbf{1}$ denotes a vector of all ones (whose dimensions are determined from context). The comparison $x \geq 0$ is a component-wise comparison, i.e. $x \in \mathbf{R}^n_+$. We refer to the ith component of a vector x by x_i, even though occasionally we will use the same notation as the ith block of vector x.

For block notation, we will use both column stacking, and row stacking, i.e., for $x \in \mathbf{R}^m$ and $y \in \mathbf{R}^n$, we have:

$$\begin{bmatrix} x \\ y \end{bmatrix} \in \mathbf{R}^{m+n}$$

and

$$\begin{bmatrix} x^T & y^T \end{bmatrix} \in \mathbf{R}^{1\times(m+n)}.$$

Norm. We denote norms of vectors and matrices using a standard notation. The l_1-norm, l_2-norm (Euclidean norm), and l_∞-norm of a vector are defined as

$$\begin{array}{lcl} \|x\|_1 & \triangleq & \sum_{i=1}^n |x_i| \\ \|x\|_2 & \triangleq & \sqrt{\sum_{i=1}^n x_i^2} \\ \|x\|_\infty & \triangleq & \max_{i=1}^n |x_i| \end{array}$$

The cardinality of a vector x is the number of non-zero components, denoted $\|x\|_0$.

The Frobenius norm of a matrix $A \in \mathbf{R}^{m\times n}$ is similar to the l_2-norm of vectors:

$$\|A\|_F = \sqrt{\sum_{i=1}^m \sum_{j=1}^n A_{ij}^2}$$

The maximum singular value of that matrix is denoted $\|A\|_2$.

Optimization. We denote our optimization problem with notation:

$$\min_{x\in\mathcal{C}} f(x)$$

where $\mathcal{C}$ is the domain for the optimization variable x. We will omit the set $\mathcal{C}$ when x is a real vector with dimension implicitly determined from context.

Note that a subset $\mathcal{C}$ of some vector space is *convex*, if the line segment between any two points $a, b \in \mathcal{C}$ is contained entirely in $\mathcal{C}$:

$$[a, b] := \{\lambda a + (1-\lambda)\, b \mid 0 \leq \lambda \leq 1\} \subseteq \mathcal{C}.$$

A real-valued function $f : \mathcal{C} \to \mathbf{R}$ is *convex* if the line segment connecting the function values at any two points $a, b \in \mathcal{C}$ lie above the graph of f in that range:

$$f(\lambda a + (1-\lambda)\, b) \leq \lambda f(a) + (1-\lambda)\, f(b),\ \forall 0 \leq \lambda \leq 1.$$

Chapter 2

Sparse optimization in text analytics

Sparse learning has been successfully used in many areas of modeling, including dynamical systems, optimal control, signal and image processing. In this chapter, we present an application of these methods that can also be useful in analyzing large collections of text documents, without requiring user expertise in machine learning. Our proposed approach relies on three main ingredients: (i) multi-document text summarization; (ii) comparative summarization of two corpora, both using sparse regression or classification; (iii) sparse principal components and sparse graphical models for unsupervised analysis and visualization of large text corpora. We first give an overview of sparse learning methods, cover fundamental sparse models, and elaborate how to apply the concept of sparse optimization into this area. This chapter serves as an introduction before we delve into the journey of a case study in text analytics with sparse optimization approaches.

2.1 Sparse learning overview

Sparse machine learning refers to a set of learning algorithms that seek a trade-off between some goodness-of-fit measure and sparsity of the result, the latter property allowing better interpretability. In a sparse learning classification task, for example, the prediction accuracy or some other classical measure of performance is not the sole concern: we also wish to be able to better identify which few features are relevant as markers for classification. Thus, if a binary classification task involves, for instance, data with genes as features, one wishes to provide not only a high-performance classifier but also one that only involves a few genes, allowing biologists to focus their further research efforts on those specific genes. Binary classification algorithms often provide a weight for each feature, hence if the weight vector is sparse (it contains many zero components), then the features with nonzero weights are the ones that are the most relevant in understanding the difference between the two classes. Similar benefits are derived from sparsity in the context of unsupervised learning, as discussed in more detail later.

There is an extensive literature on the topic of sparse machine learning, with terms such as compressed sensing, l_1-norm penalties and convex optimization [5, 18, 20, 31, 98, 110], often associated with the topic. Successful applications of sparse methods have been reported, mostly in image and signal processing, for example in [40, 73, 77]. Owing to the intensity of research in this area, many very efficient algorithms have been developed for sparse machine learning in the recent past. Despite an initial agreement that sparse learning problems are more computationally difficult than their non- sparse counterparts, a new consensus might soon emerge that sparsity constraints or penalties actually help reduce the computational burden involved in learning.

In this chapter we propose that sparse learning methods can be very useful to understand large text databases. Of course, machine learning methods in general have already been successfully applied to text classification and clustering, as evidenced by a large body of literature, for example, by Thorsten Joachims [58]. We show that sparsity is an important added property that is a crucial component in any tool aiming at providing interpretable statistical analysis, allowing in particular efficient multi-document summarization, comparison, and visualization of huge-scale text corpora. More classical algorithms, such as naive Bayes for supervised learning and latent Dirichlet allocation (LDA) [14] for unsupervised learning can also be applied for such tasks. However, these algorithms do not incorporate sparsity directly into the model, and applying them for the text processing tasks considered here requires a final 'thresholding' step to make the result interpretable, as discussed in detail later. The experiments in the following chapter indicate that the sparse learning approach provides an efficient alternative to these popular models, and in the case of LDA, at a fraction of the computational cost, much better readability of the code.

2.2 Sparse learning models

In this section, we review some of the main algorithms of sparse machine learning, and then explain how these models can be used for some generic tasks arising in text analysis.

2.2.1 Sparse classification and regression

2.2.1.1 LASSO regression.

Perhaps the most well known example of sparse learning is the variant of least-squares known as the LASSO [96], which takes the form:

$$\min_{\beta} \quad \|X^T\beta - y\|_2^2 + \lambda\|\beta\|_1 \tag{2.1}$$

where X is a $n \times m$ data matrix (with each row a specific feature, each column a specific data point), y is a m-dimensional response vector, and $\lambda > 0$ is a parameter. The l_1-norm penalty encourages the regression coefficient vector β to be sparse, bringing interpretability to the result. Indeed, if each row is a feature, then a zero element in β at the optimum of (2.1) implies that that particular feature is absent from the optimal model. If λ is large, then the optimal β is very sparse, and the LASSO model then allows to select the few features that are the best predictors of the response vector.

The LASSO problem looks more complicated than its classical least-squares counterpart. However, there is mounting evidence that, contrary to intuition, the LASSO is substantially easier to solve than least-squares, at least for high values of the penalty parameter λ. As shown later, in typical applications to text classification, a high value of λ is desired, which is precisely the regime where the LASSO is computationally very easy to solve. The so-called safe feature elimination procedure, introduced in [39], allows the algorithm to cheaply detect that some of the components of β will be zero at optimum. This in turn enables treating data sets having millions of terms and documents, at least for high values of λ.

Many algorithms have been proposed for LASSO; one example is the Alternating Direction Method of Multipliers [16]. We present the pseudocode of this algorithm below and show a Matlab implementation in the Appendix.

```
$\alpha^0 := 0$
$\beta^0 := 0$
$\gamma^0 := 0$
for t := 0, 1, 2, ..., T do
    $\alpha^{t+1} := \underset{\alpha}{\operatorname{argmin}} \lambda\|\alpha\|_1 + \frac{\rho}{2}\|\alpha - \beta^t + \gamma^t\|_2^2$
    $= \underset{\alpha}{\operatorname{argmin}} \frac{\lambda}{\rho}\|\alpha\|_1 + \frac{1}{2}\|\alpha - \beta^t + \gamma^t\|_2^2$
    $\beta^{t+1} := \underset{\beta}{\operatorname{argmin}} \|X^T\beta - y\|_2^2 + \frac{\rho}{2}\|\alpha^{t+1} - \beta + \gamma^t\|_2^2$
    $\gamma^{t+1} := \gamma^t + \alpha^{t+1} - \beta^{t+1}$
end
```

Algorithm 1: ADMM algorithm for the LASSO.

There also exist alternative algorithms in solving the LASSO. With sparse input matrix X, a simple method based on minimizing the objective function of (2.1) one coordinate of β at a time is also very competitive as shown by Friedman and Nesterov[44, 79].

2.2.1.2 Other loss functions.

Similar models arise in the context of support vector machines (SVM) for binary classification, where the sparse version takes the form (e.g. see[13]):

$$\min_{\beta,b} \quad \frac{1}{m}\sum_{i=1}^m h(y_i(x_i^T\beta + b)) + \lambda\|\beta\|_1 \tag{2.2}$$

where y is now the vector of ± 1's indicating class membership, and h is the so-called hinge loss function, with values $h(t) = \max(0, 1 - t)$. At optimum of problem (2.2), the above model parameters (β, b) yield a classification rule, i.e. predict a label $\hat{y}$ for a new data point x, as follows: $\hat{y} = \mathbf{sign}(x^T\beta + b)$. A smooth version of the above is sparse logistic regression, which obtains upon replacing the hinge loss with a smooth version

$h_{\text{logistic}}(t) = \log(1+e^{-t})$. Both of these models are useful but somewhat less popular than the LASSO, as state-of-the-art algorithms have not yet completely caught up. For text applications, Gawalt and Miratrix [48, 57] have found that LASSO regression, although less adapted to the binary nature of the problem, is still very efficient.

2.2.2 Sparse principal component analysis

The classical Principal Component Analysis (PCA) method allows reduction of the dimension of data sets by performing a low-rank approximation to the data matrix, and projecting data points on the corresponding subspace. Sparse principal component analysis (sparse PCA, see [108, 111] and references therein) is a variant of PCA that can find sparse directions of high variance. The sparse PCA problem can be formulated in many different ways, one of which (see [71, 91]) involves a low-rank approximation problem where the sparsity of the low-rank approximation is penalized:

$$\begin{array}{ll} \min\limits_{p,q} & \|M - pq^T\|_F^2 \\ \text{s.t} & \|p\|_0 \le k,\ \|q\|_0 \le h \end{array} \tag{2.3}$$

where M is the $m \times n$ data matrix, $\|\cdot\|_F$ is the Frobenius norm. In the above, the notation $\|\cdot\|_0$ stands for the cardinality, that is, the number of non-zeros in its vector argument, and $k \le m$, $h \le n$ are parameters that constrain the cardinality of the solution (p, q). The classical original PCA is obtained with $k = m,\ h = n$.

The model above results in a rank-one approximation to M (the matrix pq^T at optimum), and vectors p, q are constrained to be sparse. If M is a term-by-document matrix, the above model provides sparsity in the feature space (via p) and the document space (via a "topic model" q), allowing to pinpoint a few features and a few documents that jointly "explain" data variance.

Several algorithms have been proposed for the above problem, or related variants, see for example [27, 60, 91]. The approach in [109] is based on solving a relaxation to the problem, one column of the matrix variable at a time. Other algorithms (e.g. [91]) attempt to solve the problem directly, without any relaxation; these kinds of methods are not guaranteed to even converge to a local minimum. However, they appear to be quite efficient in practice, and extremely scalable. One such algorithm consists in solving the above problem alternatively over p, q many times [91]. This leads to a modified power iteration method:

$$p \to P(T_k(Mq)),\ q \to P(T_h(M^T p)),$$

where P is the projection on the unit circle (assigning to a non-zero vector v its scaled version $v/\|v\|_2$), and for $t \ge 0$, T_t is the "hard thresholding" operator (for a given vector v, $T_t(v)$ is obtained by zeroing out all but the t largest components of v). We provide a pseudocode for SPCA below and a MATLAB implementation in the Appendix:

```
for $k := 1, \dots, n_{topic}$ do
    $p^{(k)}, q^{(k)} := \text{PowerIteration}(M, n_w, n_d)$
    $I := \left\{i \;:\; p_i^{(k)} \neq 0\right\}$
    $J := \left\{j \;:\; q_j^{(k)} \neq 0\right\}$
    for $i \in I, j \in J$ do
        $A_{ij} := 0$
    end
    $p^{(k)}$ and $q^{(k)}$ are the $k$-th sparse principal components.
end
function PowerIteration$(M, n_w, n_d)$ :
$p := \mathbf{1}$
$q := \mathbf{1}$
while a stopping criterion has not been met do
    $p := \text{HardThresholding}\,(Aq, n_w)$
    $p := p/\|p\|_2$
    $q := \text{HardThresholding}\left(A^T p, n_d\right)$
    $q := q/\|q\|_2$
end
return $p, q$
```

Algorithm 2: Modified power iteration for SPCA with hard thresholding.

In some applications, involving for example visualization of large text databases, it is useful to distinguish positive and negative components of vectors p, q, and retain a fixed number of the largest positive and largest negative components separately. We will later elaborate on this point in our visual analysis on real datasets in the experiments to follow.

With $k = m,\ h = n$, the original power iteration method for the computation of the largest singular value of M is recovered, with optimal p, q the right- and left- singular vectors of M. The presence of cardinality constraints modifies these singular vectors to make them sparser, while maintaining the closeness of M to its rank-one approximation. The hard-thresholding version of power iteration scales extremely well with problem size, with greatest speed increases over standard power iteration for PCA when a high degree of sparsity is asked for. This is because the vectors p, q are maintained to be extremely sparse during the iterations.

An alternative simple algorithm for solving the optimization problem above is based on solving a classical PCA problem, then thresholding the resulting singular vectors so that they have the desired level of sparsity. (we discuss "thresholded models" in more details in Section 2.2.4.) For large-scale data, PCA is typically solved with power iteration, so the "thresholded PCA" algorithm is very similar to the above thresholded power iteration for sparse PCA. The only difference is in how many times thresholding takes place. Note that in practice, the thresholded power iteration for sparse PCA is much faster than its plain counterpart, since we are dealing with much sparser vectors as we perform the power iterations.

2.2.3 Sparse graphical models

Sparse graphical modeling seeks to uncover a graphical probabilistic model for multivariate data that exhibits some sparsity characteristics. One of the main examples of this approach is the so-called sparse covariance selection problem, with a Gaussian assumption on the data (see [81], and related works such as [45, 63, 70, 75, 93, 105]). Here we start with a $n \times n$ sample covariance matrix S, and assuming the data is Gaussian, formulate a variant to the corresponding maximum likelihood problem:

$$\max_X \quad \log\det X - \mathbf{Trace}(SX) - \lambda \|X\|_1 \tag{2.4}$$

where $\lambda > 0$ is a parameter, and $\|X\|_1$ denotes the sum of the absolute values of all the entries in the $n \times n$ matrix variable X. Here, $\mathbf{Trace}(SX)$ is the scalar product between the two symmetric matrices S and X, that is, the sum of the diagonal entries in the matrix product SX. When $\lambda = 0$, and assuming S is positive-definite, the solution is $X = S^{-1}$. When $\lambda > 0$, the solution X is always invertible (even if S is not), and tends to have many zero elements in it as λ grows. A zero element in the (i, j) entry of X corresponds to the conditional independence property between nodes i and j; hence sparsity of X is directly related to that of the conditional independence graph, where the absence of an edge denotes conditional independence.

The covariance selection problem is much more challenging than its classical counterpart (where $\lambda = 0$), which simply entails inverting the sample covariance matrix. At this point it appears that one of the most competitive algorithms involves solving the above problem one column (and row) of X at a time. Each sub-problem can be interpreted as a LASSO regression problem between one particular random variable and all the others [45, 81]. In Chapter 5, we will show how to solve multiple LASSO problems efficiently. Successful applications of sparse covariance selection include Senate voting [81] and gene data analysis [30, 81].

Just as in the PCA case, there is a conceptually simple algorithm, which relies on thresholding. If the covariance matrix is invertible, we simply invert it and threshold the elements of the inverse. Some limited evidence points to the statistical superiority of the sparse approach (based on solving problem (2.4)) over its thresholded counterpart.

2.2.4 Thresholded models

The algorithms in sparse learning are built around the philosophy that sparsity should be part of the model's formulation, and not produced as an afterthought. Sparse modeling is based on some kind of direct formulation of the original optimization problem, involving, typically, an l_1-norm penalty. As a result of the added penalty, sparse models have been originally thought to be substantially more computationally challenging than their non-penalized counterparts.

In practice, sparse results can be obtained after the use of almost any learning algorithm, even one that is not necessarily sparsity-inducing. Sparsity is then simply obtained via thresholding the result. This is the case for example with naïve Bayes classification: since a naïve Bayes classifier assigns weights to each feature, we can simply zero out the smaller weights to obtain a sparse classification rule. The same is true for unsupervised algorithms such as Latent Dirichlet Allocation (LDA, see [15]). In the case of LDA, the result is a probability distribution on all the terms in the dictionary. Only the terms with the highest weights are retained, which amounts in effect to threshold the probability distribution. The notion of *thresholded models* refers to the approach of applying a learning algorithm and obtaining sparsity with a final step of thresholding.

The question about which approach, "direct" sparse modeling or sparse modeling via thresholding, works better in practice, is a natural one. Since direct sparse modeling appears to be more computationally challenging, why bother? Extensive research in the least-squares case shows that thresholding is actually often sub-optimal [48]. Similar evidence has been reported on the PCA case [108]. Our own experiments in Chapter 3 support this viewpoint.

There is an added benefit to direct sparse modeling—a computational one. Originally thresholding was considered as a computational shortcut, a quick way to find sparse models. As we argued above for least-squares, SVM, logistic regression, and PCA, sparse models can be actually surprisingly easier to solve than classical models; at least in those cases, there is no fundamental reason for insisting on thresholded models, although they can produce good results. For the case of covariance selection, the situation is still unclear, since direct sparse modeling via problem (2.4) is still computationally challenging.

The above motivates many researchers to "sparsify" existing statistical modeling methodologies, such as LDA. Note that LDA also encodes a notion of sparsity, not in the feature space, but on the document (data) space: it assumes that each document is a mixture of a small number of topics, where the topic distribution is assumed to have a Dirichlet prior. Thus, depending on the concentration parameter of this prior, a document comprised of a given set of words may be effectively restricted to having a small number of topics.

This notion of sparsity (document-space sparsity) does not constrain the number of features active in the model, and does not limit overall model complexity. As a result, in LDA, the inclusion of terms that have little discrimination power between topics (such as "and" or "the"). may fall into multiple topics unless they are eliminated by hand. Once a set of topics is identified the most descriptive words are depicted as a list in order of highest posterior probability given the topic. As with any learning method, thresholding can be applied to this list to reveal the top most descriptive words given a topic. It may be possible to eliminate this thresholding step using a modified objective function with an appropriate sparsity constraint. This is an area of very active research, as evidenced by Eisenstein, Ahmed, and Xing [36].

2.2.5 Text analytics tasks

In this section, we review some of the text analytics-specific tasks that can be addressed using sparse learning methods.

2.2.5.1 Topic summarization

Topic summarization is an extensive area of research in natural language processing and text understanding. For a recent survey on the topic, see [26]. There are many instances of this problem, depending on the precise task that is addressed. For example the focus could be to summarize a single unit of text, or summarize multiple documents, or summarize two classes of documents in order to produce the summaries that offer the best contrast. Some further references to summarization include [49, 54, 78].

The approach introduced by Jia et al. in 2014 [57] relies on LASSO regression to produce a summary of a particular topic as treated in multiple documents. This is part of the extraction task within a summarization process, where relevant terms are produced and given verbatim [26]. Using predictive models for topic summarization has a long history, see for example [89]; the innovation is the systematic reliance on sparse regression models.

The basic idea is to divide the corpus in two classes, one that corresponds to the topic, and the other to the rest of the text corpus. One example is from [57]: to provide the summary of the topic "China" in a corpus of news articles from *The New York Times* over a specific period, we may separate all the paragraphs that mention the term "china" (or related terms such as "chinese", "china's", etc) from the rest of the paragraphs. We then form a numerical, matrix representation X (via, say, TF-IDF scores) of the data, and form a "response" vector (with 1's if the document mentions China and -1 otherwise). Solving the LASSO problem (2.1) leads to a vector β of regressor coefficients, one for each term of the dictionary. Since LASSO encourages sparsity, many elements of β are zero. The non-zero elements point to terms in the dictionary that are highly predictive of the appearance of "china" in any paragraph in the corpus.

The approach can be used to contrast two sets of documents. For example, we can use it to highlight the terms that allow to best distinguish between two authors, or two news sources on the same topic.

Topic summarization is closely related to *topic modeling* via Latent Dirichlet Allocation (LDA) [15], which finds on a latent probabilistic model to produce a probability distribution of all the words. Once the probability distribution is obtained, the few terms that have the highest probability are retained, to produce some kind of summary in an unsupervised fashion. As discussed in section 2.2.4, the overall approach can be seen as a form of indirect, thresholding method for sparse modeling.

2.2.5.2 Discrimination between several corpora

Here the basic task is to find out what terms best describe the differences between two or more corpora. We simply classify one of the corpora against all the others: the (say) positive class will contain all the documents from one corpus, and the negative class includes the documents from all the remaining corpus. We can use any sparse binary classification algorithm for the task, included the thresholded models referred to in section 2.2.4. The classification algorithm will identify the features that are most relevant in distinguishing a document from one class (the corpora under study) to one from the other class.

The resulting classifier weight vector, which is sparse, then points to a short list of terms that are most representative of the salient differences between the corpora and all the others. Of course, related methods such as multi-class sparse logistic regression can be used.

2.2.5.3 Visualization and clustering

Sparse PCA and sparse graphical models can provide insights to large text databases. PCA itself is a widely used tool for data visualization, but as noted by many researchers, the lack of interpretability of the principal components is a challenge. A famous example of this difficulty involves the analysis of Senate voting patterns. It is well-known in political science that, in that type of data, the first two principal components explain the total variance very accurately [81]. The first component simply represents party affiliation, and accounts for a high proportion of the total variance (typically, 80%). The second component is much less interpretable.

Using sparse PCA, we can provide axes that are sparse. Concretely this means that they involve only a few features in the data. Sparse PCA thus brings an interpretation, which is given in terms of which few features explain most of the variance. As mentioned before, it is possible to assign a fixed number of terms to each axis direction, one for the positive and one for the negative directions. (We illustrate this in our experiments on the ASRS data set.) Likewise, sparse graphical modeling can be very revealing for text data. Because it produces sparse graphs, it can bring an understanding as to which variables (say, terms, or sources, or authors) are related to each other and how.

Chapter 3

Case study in text analytics

We apply the methods in the previous chapter with a comparative study of the sparse PCA; also via a real-life example with a corpus of Aviation Safety Reporting System (ASRS) reports we demonstrate that they can reveal causal and contributing factors in runway incursions. In this example, we show that these methods, namely SPCA and LASSO, automatically discover main tasks that pilots perform during flight, which can aid in further understanding the causal and contributing factors to runway incursions and other drivers for aviation safety incidents. The comparative study on the other hand involves other commonly used datasets, and we report on the competitiveness of sparse optimization compared to state-of-the-art methods such as latent Dirichlet allocation (LDA).

Our interpretation of the results in the ASRS case study is from our submission to the ASRS text mining competition, with a focus on visual presentations of safety reports and issue recovery [38]. Note that, although text understanding and text classification are related research problems in language processing, it can be observed from the literature that the correlation between good model fit and good representative quality may be absent, or even negative in some experiments [23, 57]. In this chapter, although we mostly motivate the applications of sparse methods and our interpretation, we also refer to an ongoing line of work on evaluating sparse approaches such as the LASSO with human surveys and questionnaire, comparing the method to existing state-of-the-art alternatives, such as co-occurence, χ^2 log likelihood, and Delta TF-IDF (see [48, 57] and references therein). The big picture in this chapter is to motivate the multi-instance nature in sparse learning and hence lead to subsequent chapters on robust sketching.

3.1 Sparse PCA and LDA: comparative study

In this section, we perform a comparative study of the sparse PCA and LDA approaches, using databases that are commonly used in the text processing community. We use three data sets, of increasing size: the Amazon data set, which contains con-

```
NUMTOPICS=10
BETA=0.01;
ALPHA=50/NUMTOPICS;
ITERATIONS = 500; (LDA iterations)
WORDSPERTOPIC = 10;
SEED = 1000; (used for Gibbs sampler initialization)
```

Table 3.1: Table of default parameters used in the LDA code of [95].

sumer reviews for a variety of products; the Reuters news text categorization collection, which involves news articles; and the NSF data set, which contains abstracts from scientific articles dated 1999 through 2003. The three data sets can be obtained from the UCI archive [43]. These data sets range widely in size, from one to a thousand hundred documents. For each data set, we apply a basic stop-word removal before running the algorithms.

In this present study, we choose 10 principal components from the method and compare with the 10 topics revealed by LDA. The first component from sparse PCA is obtained by solving the sparse problem (2.3), then we update the data matrix M by removing the features found in the first component and solve for the second component and so on. We refer readers to the pseudo-code (Algorithm 2) and our implementation in the Appendix. There also exists alternative algorithms, such as Block Coordinate Ascent algorithm [109], which can be very efficient in large-scale text data. The LDA code we have used has been developed by Steyvers and Griffiths [95]; throughout, we have used the default values for various parameters, as detailed in Table 3.1.

3.1.1 Amazon Data Set

The Amazon data set is the smallest of the three data sets examined in this section, with 1500 documents, and 2960 unigrams[1]. It consists of user reviews, mainly on consumer products. The reviews originate from 50 of the most active users, each of whom has 30 reviews collected. The original data contains bigrams and trigrams, and also includes authors' usage of digits and punctuation. We have removed all of these, and retained only unigrams after stop-word removal to run the LDA and the SPCA.

The results are shown in Table 3.2. For both methods, the topics show very clear word associations. In these topics, when we rank words in non-increasing order of their weights, the topic words show up as top words. For SPCA, almost all topics are very easy to interpret: topic 1 corresponds to books, topic 2 to movies, topic 4 to games, topic 6 to cells and batteries, topic 7 to hair products, topic 8 to music. Topic 9 is less clear, but likely about electronic reading devices. For LDA, we see that topic 10 corresponds to books, topic 9 to stories, topic 3 to movie and topic 1 to music. LDA shows a similar

[1]`https://archive.ics.uci.edu/ml/datasets/Amazon+Commerce+reviews+set`

TOPIC 1		TOPIC 2		TOPIC 3		TOPIC 4		TOPIC 5	
love	0.02461	case	0.01669	film	0.03878	time	0.05518	product	0.02773
music	0.02357	light	0.01298	movie	0.02676	long	0.02607	easy	0.021
year	0.01749	included	0.01247	man	0.01594	recommend	0.02211	quality	0.0209
sound	0.01655	problem	0.01226	stars	0.01273	day	0.02171	make	0.01692
beautiful	0.01383	works	0.01216	american	0.01223	makes	0.0209	bit	0.01641
cd	0.01278	video	0.01082	bad	0.01042	fun	0.01765	hand	0.01539
great	0.01267	cells	0.01061	wife	0.01032	feel	0.01664	top	0.01346
fine	0.01267	system	0.01051	past	0.00982	good	0.01643	color	0.01305
art	0.01267	cable	0.01051	school	0.00972	game	0.01633	amazon	0.01295
christmas	0.01246	time	0.01041	films	0.00912	thing	0.01552	high	0.01193
TOPIC 6		TOPIC 7		TOPIC 8		TOPIC 9		TOPIC 10	
good	0.026	good	0.02344	find	0.03938	story	0.04881	book	0.14368
dvd	0.01856	nice	0.01863	people	0.03684	stories	0.02108	read	0.03378
made	0.01834	set	0.01824	work	0.03633	life	0.01953	author	0.02054
series	0.01618	back	0.01755	found	0.02646	family	0.01797	books	0.01952
show	0.01586	small	0.01569	make	0.02595	young	0.01578	reading	0.01926
short	0.01554	great	0.01559	things	0.01618	children	0.01551	life	0.01765
version	0.01543	easily	0.01432	part	0.01598	years	0.01533	history	0.01426
style	0.01456	put	0.01402	thought	0.01343	characters	0.01478	written	0.01392
back	0.01359	buy	0.01324	information	0.0113	world	0.01277	reader	0.01256
set	0.01349	pretty	0.01285	making	0.01099	TRUE	0.01058	interesting	0.01188

TOPIC 1		TOPIC 2		TOPIC 3		TOPIC 4		TOPIC 5	
book	0.9127	film	0.6622	nice	0.3493	game	0.7527	skin	0.5492
read	0.1527	movie	0.3617	side	0.3464	games	0.3034	children	0.3864
good	0.1437	product	0.2666	lot	0.3042	fun	0.294	young	0.3028
story	0.1331	set	0.2268	price	0.2896	play	0.2388	man	0.2378
time	0.1228	made	0.2051	light	0.287	family	0.1969	written	0.2334
life	0.1207	years	0.2014	day	0.275	world	0.1619	dry	0.2098
author	0.1058	makes	0.1889	place	0.2703	characters	0.1572	beautiful	0.208
find	0.1016	long	0.1633	series	0.2688	level	0.1477	case	0.2071
people	0.1008	dvd	0.1573	works	0.2354	character	0.1275	feel	0.2044
reading	0.0867	back	0.1566	small	0.2329	played	0.1062	times	0.1997
TOPIC 6		TOPIC 7		TOPIC 8		TOPIC 9		TOPIC 10	
cells	0.7011	hair	0.8361	songs	0.4754	cover	0.8295	writing	0.5158
capacity	0.3149	recommended	0.2397	album	0.4717	wife	0.2398	products	0.4092
mah	0.2537	style	0.2042	christmas	0.3894	similar	0.1887	handle	0.3363
nimh	0.2503	brush	0.2002	cd	0.3492	told	0.1881	perfect	0.274
aa	0.2351	highly	0.18	voice	0.2577	purchased	0.187	material	0.2682
aaa	0.2276	plastic	0.1695	song	0.2347	avoid	0.162	desk	0.2139
charger	0.1924	expensive	0.149	track	0.2011	practical	0.162	short	0.2031
package	0.1769	put	0.1252	fan	0.147	paid	0.1532	color	0.2007
cell	0.1751	hold	0.1148	fine	0.1313	kindle	0.1431	lines	0.2005
rechargeable	0.1511	ingredients	0.1054	hear	0.1247	history	0.1061	review	0.1638

Table 3.2: Comparison between LDA (top) and Sparse PCA (bottom) on the Amazon data set.

good performance, although we see some non-informative words such as "good" appear in the lists.

3.1.2 Reuters Data Set

The Reuters 21578 data set contains 19043 documents and 38361 unique words. It is one of the most frequently used for text processing on news since 1999. The dataset is divided into several categories[2]. However for the purpose of this study we have discarded the labels and any categorical information, treating all the documents on equal basis. Our goal here is to ascertain if sparse learning methods can handle data that is complex by the variety of topics, as well as the presence of acronyms and abbreviations. The results of LDA and SPCA are shown in Table 3.3.

For both methods, topics in the Reuters dataset are sometimes difficult to recognize, which is perhaps due to the complexity of this data set. There are topics that both the LDA and the SPCA agree upon. For example, LDA's topic 2 and SPCA's topic 1 have similar words: "mln" (million), "dlrs" (dollars), "net", "loss", "profit", "year" and "sales". LDA's topic 9 and SPCA's topic 5 are both on agriculture exports and oil/gas prices (with terms such as "wheat", "export", "tonnes", "price"). LDA's topic 7 and SPCA's topic 9 both discuss US government issues, with terms such as "President Reagan" and "John Roberts", respectively. Of the remaining topics, the two methods either share some commonalities (for example the LDA and SPCA topic 8 can both be guessed to be related to the European zone) or involve different topics (for example LDA's topic 1 is on economic issues, while SPCA topic 3 is on market exchange).

3.1.3 NSF Data Set

The NSF datasets contain a collection of abstracts of scientific papers, written between 1990 to 2003[3]. This is the largest data set in our study, with over $120,000$ documents and over $30,000$ words. The results of LDA and SPCA are shown in Table 3.4.

In Table 3.5 we have summarized our interpretation of each topic, mostly based on the first (most heavily weighted) term for each method. (We deviated from the rule when the other terms were consistently pointing to a more specific topic, such as topic 8 for LDA or 5 for sparse PCA.)

The two methods share many topics in common. LDA's topic 9 and SPCA's topic 1 are both on university education, with terms such as "students" and "undergraduate". LDA's topic 5 and SPCA's topic 3 are both related to material science and physics. LDA's topic 7 and SPCA's topic 6 focus on molecular and cell biology, protein function and gene expression. Overall the LDA method appears to behave slightly better on this data set; sparse PCA provides a few topics (8, 9) without clear and consistent meaning. LDA does

[2] `https://archive.ics.uci.edu/ml/machine-learning-databases/reuters21578-mld/`

[3] `https://archive.ics.uci.edu/ml/machine-learning-databases/nsfabs-mld/nsfawards.data.html`

TOPIC 1		TOPIC 2		TOPIC 3		TOPIC 4		TOPIC 5	
trade	0.02585	mln	0.20871	pct	0.08466	company	0.02703	pct	0.04074
japan	0.01705	dlrs	0.13099	billion	0.07068	corp	0.02259	dlrs	0.02596
foreign	0.01362	cts	0.07385	year	0.06001	products	0.01013	april	0.02169
government	0.01222	net	0.05054	february	0.0172	companies	0.00882	bank	0.02077
economic	0.01209	loss	0.0418	january	0.01554	contract	0.00876	debt	0.01781
world	0.01123	shr	0.03412	rose	0.01427	business	0.00783	issue	0.016
told	0.01116	profit	0.02648	march	0.01305	systems	0.00773	due	0.01507
japanese	0.01046	year	0.02635	rise	0.01153	unit	0.00749	credit	0.01428
countries	0.0104	sales	0.02135	increase	0.01147	plant	0.00728	dlr	0.01412
officials	0.01015	revs	0.01973	earlier	0.01134	services	0.00714	capital	0.01367
TOPIC 6		**TOPIC 7**		**TOPIC 8**		**TOPIC 9**		**TOPIC 10**	
company	0.04847	president	0.01869	ec	0.01111	oil	0.03011	market	0.03634
shares	0.03773	chairman	0.0124	spokesman	0.01084	prices	0.02181	bank	0.03111
stock	0.03475	federal	0.01118	government	0.01006	tonnes	0.02121	exchange	0.01704
share	0.02248	house	0.00967	european	0.0098	production	0.01648	dollar	0.01644
group	0.02059	reagan	0.00898	canada	0.00966	price	0.0122	rates	0.01369
common	0.01711	told	0.00844	union	0.00943	gas	0.00885	trading	0.01356
offer	0.0169	chief	0.0075	canadian	0.00861	wheat	0.00827	banks	0.01322
corp	0.01593	executive	0.00724	meeting	0.00853	export	0.0078	rate	0.01322
dlrs	0.01371	committee	0.00673	today	0.00818	agriculture	0.00739	stg	0.01019
board	0.00988	american	0.00664	tax	0.0081	week	0.00724	money	0.0091

TOPIC 1		TOPIC 2		TOPIC 3		TOPIC 4		TOPIC 5	
mln	0.5898	common	0.0623	government	0.1019	current	0.069	sources	0.0935
cts	0.5794	payable	0.0652	oil	0.1073	ended	0.0768	compared	0.0971
net	0.3028	bank	0.0685	agreement	0.1093	extraordinary	0.0812	price	0.1035
shr	0.2863	july	0.0695	president	0.1127	sale	0.0855	fell	0.1069
dlrs	0.1987	share	0.0716	due	0.1369	credit	0.095	production	0.1088
loss	0.1853	june	0.0839	trade	0.1415	discontinued	0.1079	prices	0.1097
revs	0.1738	corp	0.0992	debt	0.1558	operations	0.1395	exports	0.1142
profit	0.0922	shares	0.1055	today	0.1651	includes	0.1695	department	0.1187
year	0.0722	stock	0.115	york	0.191	excludes	0.2029	export	0.1214
sales	0.0719	company	0.1355	offering	0.2031	gain	0.2119	total	0.1395
TOPIC 6		**TOPIC 7**		**TOPIC 8**		**TOPIC 9**		**TOPIC 10**	
years	0.0903	rates	0.076	eurobond	0.1496	john	0.0549	directors	0.0743
spokesman	0.1008	provided	0.0837	luxembourg	0.1549	robert	0.0596	paid	0.0871
air	0.1132	week	0.0843	listed	0.1586	subsidiary	0.0605	outstanding	0.0879
work	0.118	interest	0.0845	denominations	0.1735	american	0.0609	increase	0.0927
division	0.1195	central	0.0851	underwriting	0.188	elected	0.0623	offer	0.0952
awarded	0.1244	estimate	0.0951	selling	0.2023	director	0.0739	initial	0.0979
federal	0.1495	forecast	0.0969	issuing	0.2074	effective	0.0811	cash	0.1159
general	0.154	revised	0.1189	payment	0.214	resigned	0.0864	approved	0.1313
expected	0.1588	assistance	0.1208	date	0.2334	financial	0.1265	annual	0.144
international	0.2345	shortage	0.124	management	0.2385	operating	0.1584	meeting	0.1544

Table 3.3: Comparison between LDA (top) and Sparse PCA (bottom) on the Reuters data set.

TOPIC 1		TOPIC 2		TOPIC 3		TOPIC 4		TOPIC 5	
project	0.01809	species	0.01644	research	0.09839	theory	0.02179	materials	0.01826
data	0.01516	study	0.01247	university	0.04056	problems	0.0205	high	0.01393
research	0.01218	important	0.00828	program	0.02237	methods	0.01438	properties	0.0127
information	0.00947	natural	0.00822	award	0.01927	study	0.01279	phase	0.01165
social	0.00909	provide	0.00742	chemistry	0.01804	work	0.01037	chemical	0.00837
study	0.00855	evolution	0.00721	support	0.01735	systems	0.01011	surface	0.00796
model	0.00832	understanding	0.00692	state	0.0148	problem	0.00912	energy	0.00787
models	0.00684	patterns	0.00679	dr	0.01355	mathematical	0.00892	optical	0.00747
economic	0.00597	environmental	0.00638	equipment	0.01101	models	0.00891	magnetic	0.00736
understanding	0.00573	studies	0.00604	project	0.01023	analysis	0.00799	electron	0.0068

TOPIC 6		TOPIC 7		TOPIC 8		TOPIC 9		TOPIC 10	
research	0.04002	molecular	0.01428	data	0.01493	students	0.04399	system	0.01947
scientists	0.01163	cell	0.01152	water	0.01087	science	0.03092	design	0.01944
national	0.01114	protein	0.01101	processes	0.00908	project	0.02327	systems	0.01931
researchers	0.01018	specific	0.01058	study	0.00893	program	0.01754	control	0.01329
workshop	0.00976	function	0.00979	model	0.00772	engineering	0.01521	based	0.01257
scientific	0.00958	cells	0.00942	flow	0.00761	education	0.01385	performance	0.01067
field	0.00911	studies	0.00894	ocean	0.00748	undergraduate	0.0127	data	0.01043
support	0.009	proteins	0.0089	climate	0.0067	laboratory	0.01098	develop	0.00939
areas	0.00894	mechanisms	0.00883	ice	0.00629	faculty	0.01078	network	0.00839
international	0.00886	dna	0.00804	field	0.00622	learning	0.0107	software	0.00814

TOPIC 1		TOPIC 2		TOPIC 3		TOPIC 4		TOPIC 5	
research	0.7831	theory	0.3742	materials	0.465	species	0.392	mathematics	0.4311
project	0.2861	analysis	0.2975	engineering	0.3472	molecular	0.3605	education	0.3067
students	0.228	model	0.2848	chemistry	0.3194	dr	0.3161	teachers	0.2793
university	0.2101	models	0.2846	design	0.309	chemical	0.3081	physics	0.2777
program	0.1897	problems	0.2702	laboratory	0.2879	surface	0.2715	year	0.2727
science	0.1515	understanding	0.2536	computer	0.2436	experiments	0.2677	school	0.2646
data	0.1466	studies	0.2491	state	0.2061	phase	0.2373	faculty	0.2498
study	0.1448	methods	0.2455	technology	0.2044	process	0.2359	undergraduate	0.2371
support	0.1356	important	0.2374	techniques	0.2022	determine	0.2258	college	0.2279
systems	0.1292	information	0.2273	properties	0.1972	effects	0.2014	student	0.2183

TOPIC 6		TOPIC 7		TOPIC 8		TOPIC 9		TOPIC 10	
cell	0.4392	abstract	0.7037	group	0.4609	order	0.4939	equipment	0.5951
protein	0.3731	fellowship	0.51	groups	0.4016	experimental	0.2891	nsf	0.362
cells	0.3254	postdoctoral	0.3304	areas	0.3214	theoretical	0.2804	projects	0.3613
proteins	0.2873	required	0.2857	number	0.2699	dynamics	0.2796	grant	0.2695
plant	0.2709	error	0.113	area	0.2677	scientific	0.2789	network	0.2596
gene	0.2686	mathematical	0.1006	water	0.231	task	0.2612	funds	0.1912
genes	0.2525	sciences	0.0961	focus	0.2276	test	0.243	performance	0.1826
dna	0.2093	worry	0.0846	behavior	0.2159	scientists	0.2212	community	0.1692
function	0.2075	matter	0.0462	environmental	0.2149	level	0.2193	instrumentation	0.1597
biology	0.1953	length	0.044	related	0.2063	national	0.215	magnetic	0.15

Table 3.4: Comparison between LDA (top) and Sparse PCA (bottom) on the NSF data set.

	TOPIC 1	TOPIC 2	TOPIC 3	TOPIC 4	TOPIC 5	TOPIC 6	TOPIC 7	TOPIC 8	TOPIC 9	TOPIC 10
LDA	Project	Species	Research	Theory	Materials	Research	Molecular	Data/Climate	Students	Systems
SPCA	Research	Theory	Materials	Species	Mathematics/Education	Cell	Abstract	Research Group	Order	Equipment

Table 3.5: Manually interpreted topics in the NSF data set experiment.

Data set	Amazon	Reuters	NSF
LDA	1 minute	13 minutes	1 hour
SPCA	**19 seconds**	**3 minutes**	**14 minutes**

Table 3.6: Computational times for LDA and sparse PCA.

provide topics with overlap, and/or without much specificity (topics 1 and 3 for example), while non-overlap is automatically enforced with sparse PCA. As discussed next, sparse PCA runs much faster than LDA.

3.1.4 Running time

To summarize our findings, we note that overall both LDA and sparse PCA behave well and comparably on the data sets we have used. In the larger data set (NSF), LDA delivers better results. A clear advantage, besides performance, of sparse methods lies with their ease of use and readability; our code for sparse power iteration is a few lines long, and is quite amenable to a distributed computing architecture. Another clear advantage lies with the computational effort that is required. To our knowledge, there is no precise computational complexity analysis for both methods. In our experiments we have observed that LDA takes much longer to run than sparse PCA. Table 3.6 illustrates the dramatic difference in run times.

3.2 Case study on ASRS data

In this section, we perform an analysis of a specific data set coming from the Aviation Safety Reporting System database [4]. This database contains reports generated by pilots, air traffic controllers, and others on a voluntary basis, and is a crucial component of the continuing effort to maintain and improve aviation safety. The ASRS data contains several of the crucial and generic challenges involved under the general banner of “large-scale text data understanding”. First, its scale is growing rapidly, making the need for automated analyses of the processed reports more crucial than ever. Another issue is that the reports themselves are far from being syntactically correct, with lots of abbreviations, orthographic, and grammatical errors. Thus, we are not facing a corpus with well-structured language having clearly-defined rules as we would if we were to consider a corpus of laws or bills or any other well-redacted data set. Finally, in many cases we do not know in advance what to look for, because the goal is to discover precursors to aviation safety incidents and accidents. In other words, the task is not about search, and finding a needle in a haystack: in many cases, we cannot simply monitor the emergence or disappearance of a few keywords that would be known in advance. Instead, the task

[4]See `http://asrs.arc.nasa.gov/search/database.html` for more information on the ASRS system. The text reports are available on that website.

resembles more one of trying to visualize the haystack itself, compare various parts of it, or summarize some areas.

Our focus is on reports from the Aviation Safety Reporting System (ASRS). The ASRS is a voluntary program in which pilots, co-pilots, other members of the flight crew, flight controllers, and others file a text report to describe any incident that they may have observed that has a bearing on aviation safety. Because the program is completely voluntary and the data are de-identified, meaning that the author, his or her position, the carrier, and other identifying information is not available in the report. After reports are submitted, analysts from ASRS may contact the author to obtain clarifications. However, the information provided by the reporter is not investigated further. This motivates the use of (semi-) automated methods for the real-time analysis of the ASRS data. In our experiments, we have used the data provided by NASA as part of the SIAM 2007 Text Mining Competition[5]. It consists of more than 20,000 flight reports submitted by pilots after their flights. Each report is a small paragraph describing any incident that was recorded during flight, and is assigned a category (totaling 22), or type of incident.

Our goals here are as follows. We first report on previous work on this particular data set (Section 3.3). Then in Section 3.4, our aim is to experiment our methods based on categorical information. Using our comparative summarization methods, we investigate if we can recover summaries for each category that can clearly distinguish between them, and are consistent with their meaning.

In Section 3.4.2, we illustrate how sparse PCA can be used to visualize the data, specifically visualize the different categories. We also make a comparison with thresholded LDA. In Section 3.5, we focus on the analysis of runway incursions, which are events in which one aircraft moves into the path of another aircraft during landing or takeoff. A key question that arises in the study of runway incursions is to understand whether there are significant distinguishing features of runway incursions for different airports. Although runway incursions are common, the causes may differ with each airport. These are the causal factors that enable the design of the intervention appropriate for that airport, whether it may be runway design, runway lighting, procedures, etc. To do this kind of analysis, we prepare a special second dataset, obtained from ASRS data as detailed below (we shall call it Runway dataset onward; and we will experiment on both ASRS and this special data).

Runway dataset. We present our special data preparation step-by-step as follows:

1. We reverted all characters to lower case and scrubbed all punctuation and special characters. We remove redundant white spaces and remove stop words.

2. We glued each runway or taxiway and their labels as a single word. This is achieved by the following regular expression: `/\b(runway|taxiway)\ [a-z0-9]{1,3}\b/` .

[5]`https://c3.nasa.gov/dashlink/resources/138/`

This, for example, converts `taxiway xy` to `taxiway_xy`, and `runway 9x` to `runway_9x`.

The purpose of this step is to enable us to just focus on the issues raised by the runways, by the taxiways. Additionally, this allows for immediate perception of machine learning results by the reviewer. For example, we would rather expect a list of words associated to an airport as `taxiway_91`, `runway_00`, `runway_y` than simply `91`, `00`, `y` etc.

3. We tokenized each ASRS report by separating the words of the ASRS report using a single space.

4. We vectorized the text by extracting all uni-grams from the sample. In doing so, we first scan through all ASRS documents and build a dictionary of all uni-grams ever used across a total of $21,519$ ASRS reports. We thus find $27,081$ distinct uni-grams. Hence, we obtain a data matrix of dimension $21,519 \times 27,081$.

3.3 Related work on ASRS data

In this section we list some previous work in applying data mining/machine learning methods for analyzing ASRS data, along with pointers for further research.

Text Cube [68] and Topic Cube [107] are multi-dimensional data cube structures which provide a solid foundation for effective and flexible analysis of the multidimensional ASRS text database. The text cube structure is constructed based on the TF/IDF (i.e., vector space) model while the topic cube is based on a probabilistic topic model. Techniques have also been developed for mining repetitive gapped subsequences [28], multi-concept document classification [102, 103], and weakly supervised cause analysis [1]. The work by Cindy et al. [68] has been further extended by Ding et al. in 2010 [29] where the authors have proposed a keyword search technique. Given a keyword query, the algorithm ranks the aggregations of reports, instead of individual reports. For example, given a query "forced landing" an analyst may be interested in finding the external conditions (e.g. weather) that causes this kind of query and also find other anomalies that might co-occur with this one. This kind of analysis can be supported through keyword search, providing an analyst a ranked list of such aggregations for efficient browsing of relevant reports. In order to enrich the semantic information in a multidimensional text database for anomaly detection and causal analysis, Persing and Ng have developed new techniques for text mining and causal analysis from ASRS reports using semi-supervised learning [84] and subspace clustering [4].

Some work has also been done on categorizing ASRS reports into anomalous categories. It poses some specific challenges such as high and sparse dimensionality as well as multiple labels per document. Oza et al. [82] presents an algorithm called Mariana which learns a one-vs-all SVM classifier per anomaly category on the bag-of-words matrix. This provides good accuracy on most of the ASRS anomaly categories.

Topic detection from ASRS datasets have also received some recent attention. Shan et al. have developed the Discriminant Latent Dirichlet Allocation (DLDA) model [90], which is a supervised version of LDA. It incorporates label information into the generative model using logistic regression. Compared to Mariana, it not only has a better accuracy, but it also provides the topics along with the classification.

Gaussian Process Topic Models (GPTMs) by Agovic and Banerjee [3] is a novel family of topic models which define a Gaussian Process Mapping from the document space into the topic space. The advantage of GPTMs is that it can incorporate semi-supervised information in terms of a Kernel over the documents. It also captures correlations among topics, which leads to a more accurate topic model compared to LDA. Experiments on ASRS dataset show better topic detection compared to LDA. The experiments also illustrate that the topic space can be manipulated by changing the Kernel over documents.

3.4 Understanding Categories

3.4.1 Recovering categories

In our first experiment, we sought to understand if sparse learning methods could perform well in a blind test. The categorical data did not contain category *names*, only referring to them with letter capitals. We sought to understand what these categories were about. To this end, we have solved one LASSO problem for each category, corresponding to classifying that category against all the others [57]. As shown in Table 3.7, we did recover a differentiated image of the categories. For example, the categories M, T, U correspond to the ASRS categories *Weather/Turbulence*, *Smoke/Fire/Fumes/Odor,* and *Illness*. These categories names are part of the ASRS Events Categories as defined in `http://asrs.arc.nasa.gov/docs/dbol/ASRS_Database_Fields.pdf`. This blind test indicates that the method reveals the underlying categories using the words in the corpus alone. Some abbreviations in Table 3.7 are explained in Table 3.8; a full list of abbreviations can be found on ASRS website `https://asrs.arc.nasa.gov/docs/dbol/ASRS_Abbreviations.pdf` .

The analysis reveals that there is a singular category, labelled B. This category makes up about 50% of the total number of reports. Its LASSO images points to two terms, which happen to be two categories, A (mechanical issues) and N (airspace issues). The other terms in the list are common to either A or N. Our interpretation and analysis point to the fact that category is a “catch-all” one, and that many reports in it could be re-classified as A or N.

3.4.2 Sparse PCA for understanding categories

In this section, we plot the data set on a pair of axes that contain a lot of the variance, at the same time maintaining some level of interpretability to each of the four directions.

Category	Term 1	Term 2	Term 3	Term 4	Term 5	Term 6	Term 7
A (1441)	MEL	install	maintain	mechanic	defer	logbook	part
B (12876)	CATA	CATN	airspace	install	MEL	AN	
C (393)	abort	reject	ATO	takeoff	advance	TOW	pilot
D (428)	grass	CATJ	brake	mud	veer	damage	touchdown
E (3062)	runway	taxi	taxiway	hold	tower	CATR	ground control
F (6065)	CATH	clearance	cross	hold	feet	runway	taxiway
G (1684)	altitude	descend	feet	CATF	flightlevel	autopilot	cross
H (2213)	turn	head	course	CATF	radial	direct	airway
I (405)	knotindicator	speed	knot	slow	airspeed	overspeed	speedlimit
J (1107)	CATO	CATD	wind	brake	encounter	touchdown	pitch
K (353)	terrain	GPWS	GP	MD	glideslope	lowaltitude	approach
L (3357)	traffic	TACAS	RA	AN	climb	turn	separate
M (2162)	weather	turbulent	cloud	thunderstorm	ice	encounter	wind
N (1261)	airspace	TFR	area	adiz	classb	classdairspace	contact
O (325)	CATJ	glideslope	approach	high	goaraound	fast	stabilize
P (935)	goaround	around	execute	final	approach	tower	miss
Q (394)	gearup	land	towerfrequency	tower	contacttower	gear	GWS
R (1139)	struck	damage	bird	wingtip	truck	vehicle	CATE
S (6767)	maintain	engine	emergency	CATA	MEL	gear	install
T (647)	smoke	smell	odor	fire	fume	flame	evacuate
U (304)	doctor	paramedic	nurse	ME	breath	medic	physician
V (574)	police	passenger	behave	drink	alcohol	seat	firstclass

Table 3.7: LASSO images of the categories: each list of terms correspond to the most predictive list of features in the classification of one category against all the others. The numbers in parentheses denote the number of reports in each category. The meaning of abbreviations is listed in Table 3.8.

Meaning	Abbreviation
aborted take-off	ATO
aircraftnumber	AN
airtrafficcontrol	ATC
gearwarningsystem	GWS
groundproximity	GP
groundproximitywarningsystem	GPWS
groundproximitywarningsystemterrain	GPWS-T
knotsindicatedairspeed	KIAS
medicalemergency	ME
minimumdescent	MD
minimumequipmentlist	MEL
noticestoairspace	NTA
resolutionadvisory	RA
trafficalertandcollisionavoidancesystem	TACAS
takeoffclear	TOC
takeoffwarning	TOW
temporaryflightrestriction	TFR

Table 3.8: Some abbreviations used in the ASRS data.

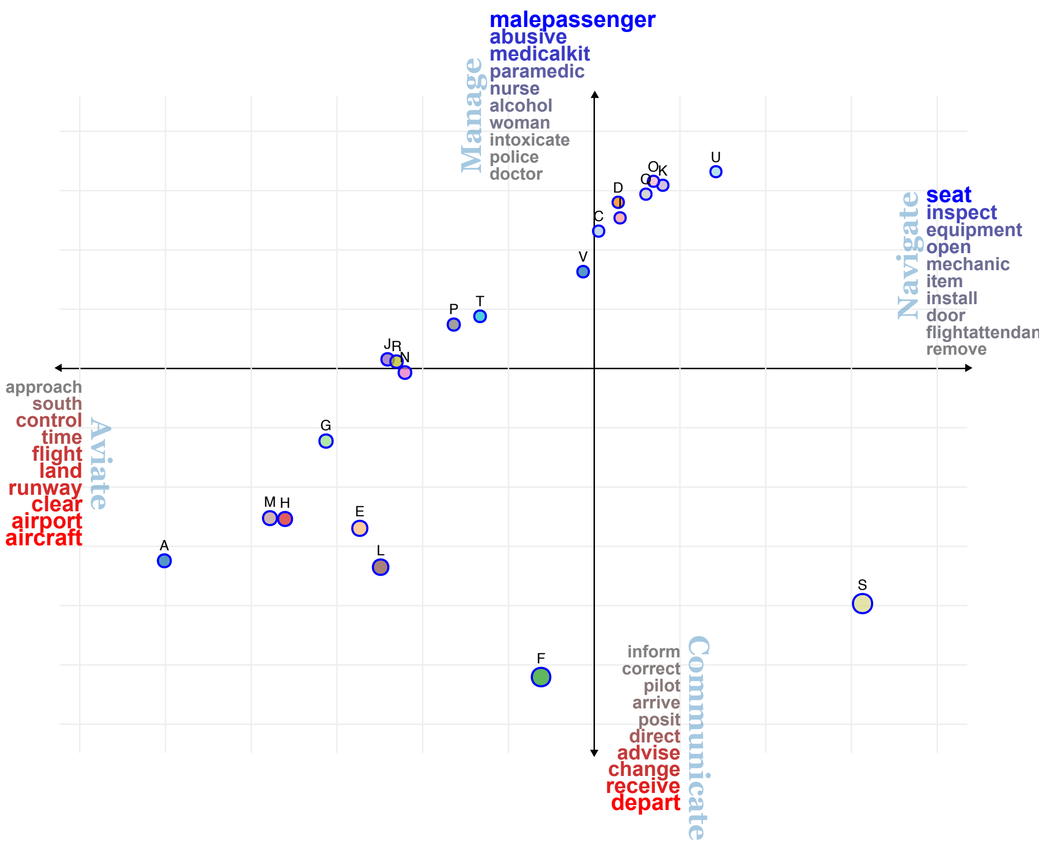


Figure 3.1: A sparse PCA plot of the category ASRS data. Here, each data point is a category, with size of the circles consistent with the number of reports in each category. We have focussed the axes and visually removed category B which appears to be a catch-all category. Each direction of the axes is associated with only a few terms, allowing an easy understanding of what each means. Each direction matches with one of the missions assigned to pilots in FAA documents (in light blue).

Here the purpose is simply to perform an exploratory data analysis step, and evaluate if the results are consistent with domain knowledge. Our choice for setting the number of (sparse) principal components to two is not related to the data set itself. Rather, our choice simply allows us to plot the data on a two-dimensional figure, each component leading to one positive or negative direction.

We have proceeded with this analysis on the category data set. To this end we have applied sparse PCA using modified power iteration with hard thresholding (Algorithm 2) to the category data matrix M (with each column an ASRS report), and obtained Fig. 3.1. We have not thresholded the direction q, only the direction p, which is the vector along which we project the points, so that it has at most 10 positive and 10 negative components. Hence, on our plot the underlying space is that corresponding to vector p.

Sparse PCA plot shows that the data involves four different themes, each corresponding to the positive and negative directions of the first two sparse principal components.

Without any supervision, the sparse PCA algorithm found themes along with the four missions of pilots, widely cited in aviation documents [59]: *Aviate*, *Navigate*, *Communicate*, and *Manage Systems*. These four actions form the basis of flight training for pilots in priority order. The first and foremost activity for a pilot is to aviate, i.e., ensure that the airplane stays aloft and in control. The second priority is to ensure that the airplane is moving in the desired direction with appropriate speed, altitude, and heading. The third priority is to communicate with other members of the flight crew and air traffic control as appropriate. The final priority is to manage the systems (and humans involved) on the airplane to ensure safe flight. These high-level tasks are critical for pilots to follow because of their direct connection with overall flight safety.

The sparse algorithm discovers these four high-level tasks as the key factors in the category data set. The words associated with each direction in Fig. 3.1 (for example, "seat". "inspect", etc, along the East direction) were automatically assigned by the algorithm. On the plot, we manually assigned a higher-level label (such as "Navigate") to the list of words associated with each direction. As claimed, the list of words are very consistent with the high-level labels.

We validated our discovery by applying the Latent Dirichlet Allocation algorithm to the ASRS data and set the desired number of topics equal to 4. Because there is currently no method to discover the "correct" number of topics, we use this high-level task breakdown as for an estimate of the number of topics described in the documents. While the results did not reveal the same words as sparse PCA, it revealed a similar task breakdown structure. More detailed results involving LDA are described in section 3.4.3.

Runway data. In a second illustration we have analyzed the Runway data set described in the introduction of Chapter 3. Fig 3.2 shows that two directions remain associated with the themes found in the category data set, namely "aviate" (negative horizontal direction) and "communicate". The airports near those directions, in the bottom left quadrant of the plot (CLE, DFW, ORD, LAX, MIA, BOS) are high-traffic ones with relatively bigger number of reports, as is indicated by the size of the circles. This is to be expected from airports where large amounts of communication is necessary (due to high traffic volume and complicated layouts). Another cluster (on the NE quadrant) corresponds to the two remaining directions, which we labelled "specifics" as they related to specific runways and taxiways in airports. This other cluster of airports seem to be affected by issues related to specific runway configuration that are local to each airport.

Feature removal on Runway data. In the third plot (Fig. 3.3) we redid the analysis after removal of all the features related to runways and taxiways, in order to discover what is "beyond" runway and taxiway issues. We recover the four themes of *Aviate*, *Navigate*, *Communicate* and *Manage*. As before, high-traffic airports remain affected mostly by aviate and communicate issues. Note that the disappearance of passenger-related issues

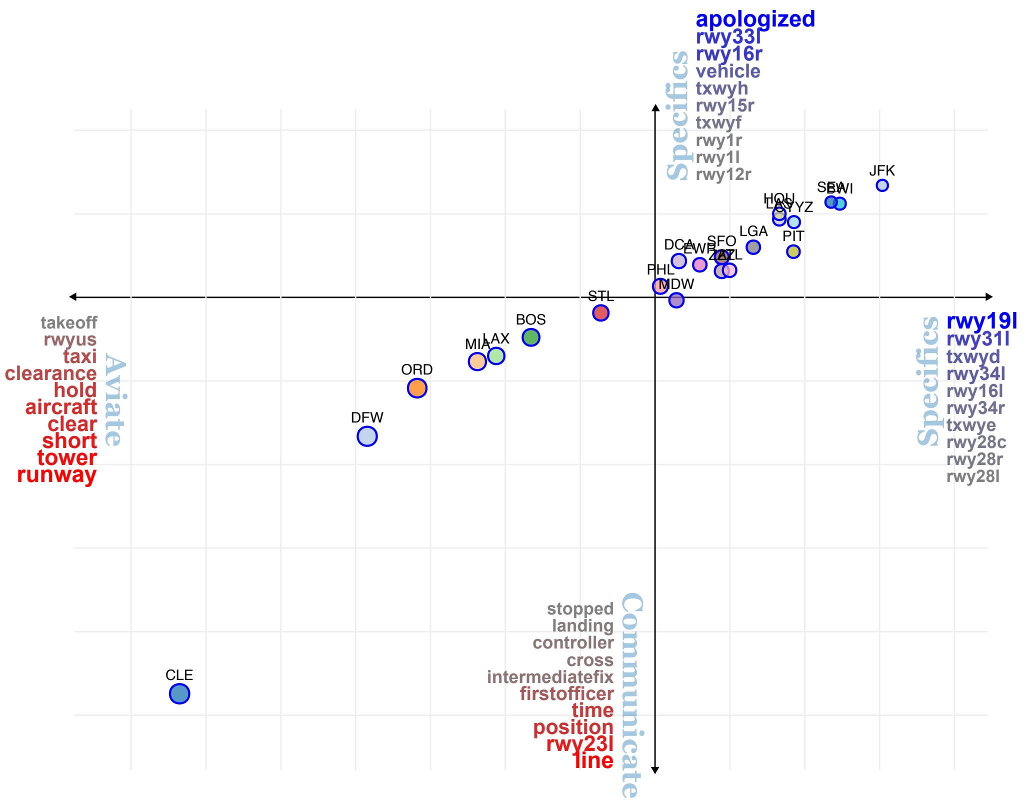


Figure 3.2: A sparse PCA plot of the Runway data. Here, each data point is an airport, with size of the circles consistent with the number of reports for each airport. We note that two out of four directions are more airport-specific (shown in blue).

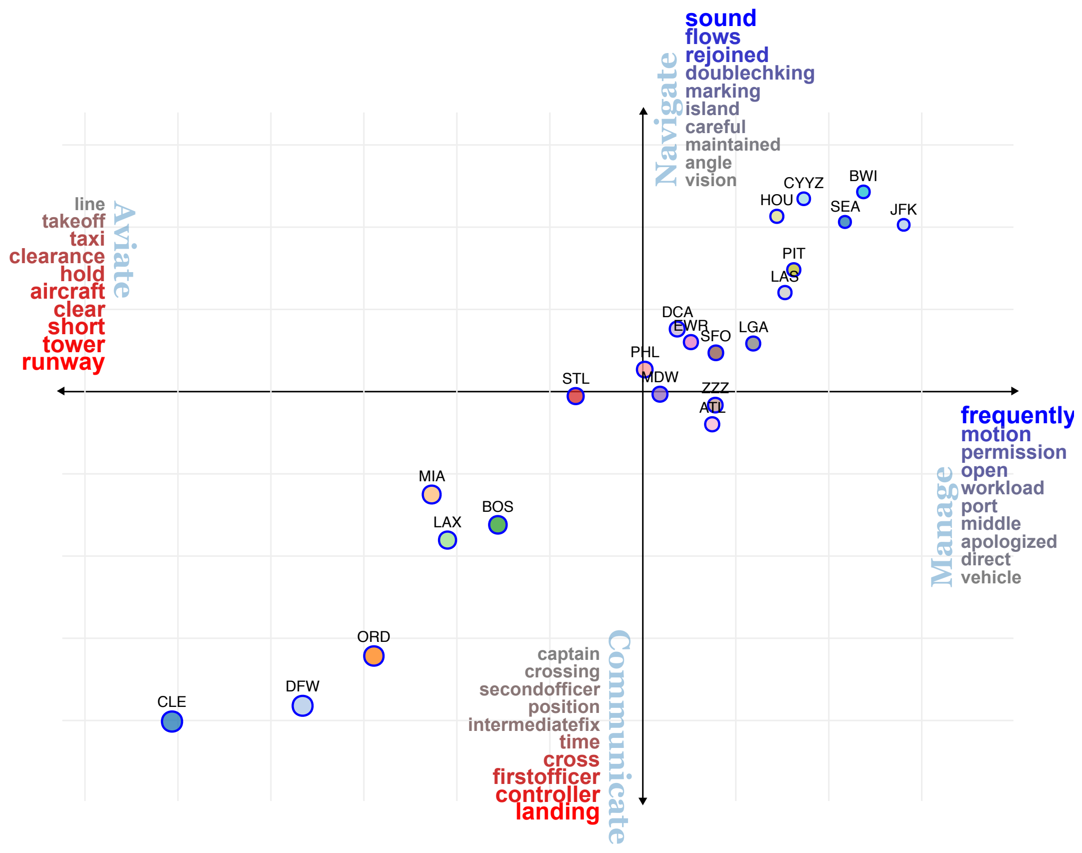


Figure 3.3: A sparse PCA plot of the runway ASRS data, with runway features removed.

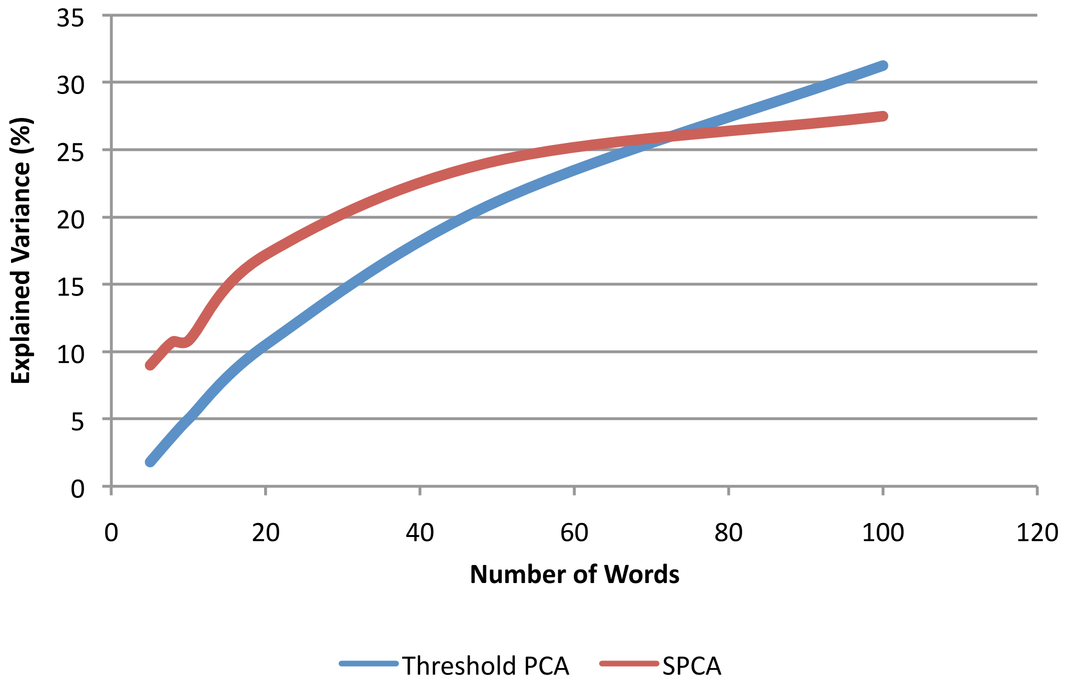


Figure 3.4: Explained variance by SPCA and thresholded PCA.

within the *Manage* theme, which was defining the positive-vertical direction in Fig 3.1. This is to be expected, since the data is now restricted to runway issues: what involved passenger issues in the category data set, now becomes mainly related to the other humans in the loop, pilots ("permission"), drivers ("vehicle") and other actors, and their actions or challenges ("workload, open, apologized").

One look at the sparse PCA plots (Figs. 3.1 and 3.3) reveals a commonality: the themes of *Aviate* and *Communicate* seem to go together in the data, and are opposed to the other sub-group of *Navigate* and *Manage Systems*.

Thresholded PCA. Fig. 3.4 shows the total explained variance by sparse and thresholded PCA discussed in Section 2.2.4 as a function of the number of words allowed for the axes, for the category data set. With the empirical covariance matrix Σ from the data, the total explained variance $\text{var}\,(u)$ given a unit-norm principal component u is defined as:

$$\text{var}\,(u) := u^T \Sigma u.$$

We observe that thresholded PCA does not explain as much variance (in fact, only half as much) as sparse PCA, with the same budget of words allowed for each axis. This ranking is reversed only after 80 words are allowed in the budget. The two methods do reach the maximal variance explained by PCA as we relax our word-budget constraint. Similar observations can be made for the runway data set.

	term 1	term 2	term 3	term 4	term 5	term 6	term 7	term 8	term 9	term 10
Topic 1	runway	taxi	ground	taxiway	turn	control	captain	airport	gate	txwyno
Topic 2	tower	clear	takeoff	aircraft	rwyus	clearance	position	firstofficer	captain	flight
Topic 3	runway	hold	short	line	cross	told	rwyus	nar	aircraft	taxi
Topic 4	runway	aircraft	landing	ctlapproachcontrol	feet	report	lights	pilot	due	crew

Table 3.9: 4 topics extracted from ASRS dataset.

	term 1	term 2	term 3	term 4	term 5	term 6	term 7	term 8	term 9	term 10
Topic 1	captain	firstofficer	time	flight	departure	secondofficer	airtrafficcontrol	chklist	txwyme	crew
Topic 2	runway	hold	short	line	aircraft	stopped	taxi	stop	nar	clear
Topic 3	runway	taxi	ground	control	cross	rwyus	instructions	crossing	told	gate
Topic 4	tower	clear	takeoff	clearance	position	rwyus	aircraft	aircarrier	controller	call
Topic 5	runway	aircraft	landing	ctlapproachcontrol	feet	traffic	tower	clear	landed	approximately
Topic 6	runway	taxiway	taxi	turn	airport	report	txwyno	lights	end	area

Table 3.10: 6 topics extracted from ASRS dataset.

3.4.3 Thresholded Latent Dirichlet Allocation

For the sake of comparison, we have similarly applied the Latent Dirichlet Allocation (LDA) algorithm to the ASRS data. LDA is an unsupervised technique for topic modeling and as such it requires the number of topics to extract from the data. For our ASRS data, we have generated 4, 6, and 10 topics. We have used the code in [95], with default parameter values, as previously detailed in Table 3.1.

In our first attempt, we have not removed any stop words and found the corresponding lists to be quite uninformative, as stop words did show up. We have then removed the stop words using a standard list of stop words from the English dictionary.

Table 3.9 shows the four topics with the top 10 words (according to posterior distribution) thresholded from the entire distribution of words. Unlike the Spase PCA method (Fig. 3.1), the 4 topics of LDA model do not correspond to the four missions of the pilot: *Aviate*, *Navigate*, *Communicate*, and *Manage Systems*. In fact, there are certain words such as ‘aircraft’, ‘runway’ etc. which seem to occur in most of the topics and are therefore not very informative for discrimination purposes. From a high level, the topics roughly seem to correspond to the following: (1) Topic 1 – gate events or ground events, (2) Topic 2 – ATC communication or clearance related, (3) Topic 3 – not clear , and (4) Topic 4 – approach/landing.

Tables 3.10 and 3.11 depicts 6 and 10 topics extracted from the ASRS data. Both of these tables show that there the topics are not very unique since the words in the topics appear to be substantially overlapping and therefore, (1) there is not much discriminative power of the components (words) and, (2) the topics do not discover unique structures in the data. Finally, we report the running time of LDA algorithm for these three experiments: 12 secs for 4 topics, 16 secs for 6 topics and 25 secs for 10 topics. For all these experiments, we have run the Gibbs sampler for 500 iterations.

	term 1	term 2	term 3	term 4	term 5	term 6	term 7	term 8	term 9	term 10
Topic 1	runway	clearance	controller	cross	clear	crossing	back	instructions	airtrafficcontrol	aircraft
Topic 2	runway	taxiway	taxi	end	txwyno	lights	airport	turn	turned	side
Topic 3	aircraft	runway	tower	clear	takeoff	landing	rwyus	nar	stop	speed
Topic 4	taxi	time	airport	ramp	crew	flight	departure	problem	due	factors
Topic 5	runway	ground	taxi	rwyus	control	told	instructions	gate	instructed	crossed
Topic 6	tower	takeoff	position	clear	rwyus	aircarrier	call	frequency	heard	clearance
Topic 7	hold	short	line	runway	report	stopped	past	nar	taxi	holding
Topic 8	runway	intermediatefix	txwyme	txwyno	secondofficer	intersection	asked	txwydo	txwygo	approach
Topic 9	ctlapproachcontrol	feet	landing	aircraft	traffic	final	approximately	land	report	pilot
Topic 10	captain	taxi	firstofficer	runway	turn	chklist	time	looked	rwy4l	txwyb

Table 3.11: 10 topics extracted from ASRS dataset.

3.5 Analysis of runway incursion incidents

In this section, our objective is to understand specific runway-related issues affecting each airport using the Runway data.

We will use three different methods to obtain the image (as given by a short list of terms) for each airport. A first approach is basic and relies on co-occurrence between the airport's name and the other terms appearing in documents mentioning that name. The two other approaches, thresholded naïve Bayes and LASSO, rely on classification. For this, we separate the data into two sets: one set corresponds to the ASRS reports that contain the name of the airport under analysis; the other contains all the remaining ASRS documents in our corpus. We have selected for illustration purposes the top 22 airports, as ordered by the number of reports that mention their names.

3.5.1 Co-occurrence analysis

With no stop words removed or word-stemming, the simplest method is the co-occurrence on term frequency, which expectedly gives commonly-used words with little meaning as term association for the airports. Results are shown in Table 3.12. Among these top words across the airports are simply “the”, “runway”, “and”.

To avoid stop words, we also experiment with the TF-IDF scores for the co-occurrence method, which adds a weight of inverse document frequency to each term. When considering an airport, TF-IDF generally favors terms that occur more exclusively in documents containing the name of that airport. Results are shown in Table 3.13. Among the top 8 terms chosen for each airport in the experimentation are: the airport name (ATL, LGA, LAS, BWI, JFK) and specific runways with taxiways that have reported aviation issues. Some focus on actions are shown in a few airports: MIA (takeoff), PHL (cross), DCA and BWI (turn).

3.5.2 Näıve Bayes classification

To emphasize the differences between two sets of documents, one method is to make use of the Naïve Bayes classifier on the binary term-appearance matrix. This method relies on a strong assumption of term's independence across the whole corpus. To obtain the term

Airport	term 1	term 2	term 3	term 4	term 5	term 6	term 7	term 8
CLE	the	runway	and	i	was	hold	short	a
DFW	the	runway	and	i	was	tower	a	aircraft
ORD	the	runway	and	i	was	a	that	were
MIA	the	runway	and	was	i	a	hold	taxi
BOS	the	runway	and	i	was	a	hold	were
LAX	the	runway	and	i	was	a	hold	short
STL	the	runway	and	i	was	short	a	that
PHL	the	runway	and	was	i	aircraft	taxi	a
MDW	the	runway	and	i	was	a	taxi	hold
DCA	the	runway	and	i	was	a	were	that
SFO	the	runway	and	i	was	a	that	aircraft
ZZZ	the	and	runway	i	was	a	aircraft	were
EWR	the	runway	and	i	was	a	tower	that
ATL	the	runway	and	was	i	a	aircraft	tower
LGA	the	runway	and	was	i	aircraft	hold	a
LAS	the	runway	and	i	was	a	for	were
PIT	the	runway	and	was	i	a	taxi	that
HOU	the	runway	and	i	was	for	a	rwy12r
BWI	the	runway	and	was	i	taxi	a	that
CYYZ	the	runway	and	hold	short	was	i	line
SEA	the	runway	and	i	was	hold	tower	a
JFK	the	runway	and	was	i	a	that	clear

Table 3.12: Images of airports via the co-occurrence method on the binary term by document matrix, without stop word removal.

Airport	term 1	term 2	term 3	term 4	term 5	term 6	term 7	term 8
CLE	rwy23l	rwy24l	rwy24c	cle	rwy23r	rwy5r	rwy6r	rwy5l
DFW	rwy18l	dfw	rwy17r	rwy35l	rwy35c	rwy17c	rwy18r	rwy36r
ORD	ord	rwy22r	rwy27r	rwy32r	rwy27l	rwy9l	rwy4l	rwy22l
MIA	rwy9l	mia	txwyq	rwy9r	line	rwy8r	txwym	takeoff
BOS	rwy4l	bos	rwy33l	rwy22r	rwy22l	rwy4r	captain	frequency
LAX	rwy25r	lax	rwy25l	rwy24l	rwy24r	i	captain	firstofficer
STL	rwy30l	rwy12l	rwy12r	stl	rwy30r	cross	aircarrier	short
PHL	rwy9l	rwy27r	phl	rwy27l	txwyk	x	e	cross
MDW	rwy31c	rwy31r	mdw	rwy22l	rwy4r	txwyp	midway	rwy13c
DCA	dca	txwyj	airplane	turn	ground	traffic	i	pad
SFO	rwy28l	rwy28r	sfo	rwy1l	rwy1r	rwy10r	rwy10l	captain
ZZZ	xxr	zzz	radio	hangar	tow	i	speed	rwyxa
EWR	rwy4l	rwy22r	ewr	rwy22l	txwyp	txwyz	rwy4r	txwypb
ATL	atl	rwy26l	rwy8r	rwy9l	rwy27r	rwy26r	dixie	atlanta
LGA	lga	txwyb	instrumentlandingsystem	txwyb4	vehicle	line	lights	txwyp
LAS	las	rwy25r	rwy19l	rwy7l	rwy1r	rwy19r	rwy25l	rwy1l
PIT	rwy28c	rwy10c	pit	rwy28l	txwye	txwyw	txwyv	txwyn1
HOU	rwy12r	hou	rwy12l	heading	takeoff	i	rwy30r	txwyme
BWI	bwi	rwy15r	txwyp	rwy33l	turn	intersection	txwyp1	taxiway
CYYZ	txwyq	txwyh	yyz	line	rwy6l	rwy33r	short	length
SEA	rwy34r	rwy16l	rwy34l	sea	rwy16r	position	firstofficer	y
JFK	jfk	rwy31l	vehicle	rwy13r	rwy4l	rwy22r	rwy13l	rwy31r

Table 3.13: Images of airports via the co-occurrence method, using TF-IDF scores.

association for each airport, we compute the estimated log-odds ratio of term appearance in "positive" documents to that in "negative" ones, normalized by the variance of this estimation, in order to cope with noise in the data. Hard thresholding these log-odds ratios allows to retain a fixed number of terms associated to each airport.

Results from the Naïve Bayes classification are shown in Table 3.14. It seems that the method, applied to the runway ASRS dataset, is effective in pointing out generic actions relevant to the aviation system. Term associations mostly reveal "cross", "landed", "tower" as strong discriminating features. Nevertheless, this generic result provides little help in understanding specific runway-related issues that affect each airport.

3.5.3 LASSO

We turn to a LASSO regression to analyze the image of each airport. Our results, shown in Table 3.15, are based on the TF-IDF representation of the text data. They indicate that the LASSO images for each airport reveal runways that are specific to that airport, as well as some specific taxiways. We elaborate on this next.

3.5.4 Tree images via two-stage LASSO

To further illustrate the LASSO-based approach, we focus on a single airport (say DFW). We propose a two-stage LASSO analysis allowing to discover a tree structure of

Airport	term 1	term 2	term 3	term 4	term 5	term 6	term 7	term 8
CLE	line	short	this	are	for	following	first	didn
DFW	cross	crossing	tower	landed	aircraft	across	short	holding
ORD	turn	but	when	l	speed	get	than	txwyb
MIA	rwy9l	txwyp	taxiway	txwym	signage	chart	line	via
BOS	frequency	told	s	contact	controller	txwyk	rwy4l	rwy22r
LAX	tower	cross	short	call	high	landing	speed	firstofficer
STL	cross	line	short	call	hold	aircraft	trying	supervisor
PHL	cross	e	rwy9l	crossed	x	txwye	spot	txwyk
MDW	clearance	i	taxi	hold	gate	captain	crossed	short
DCA	his	turn	just	captain	but	airplane	txwyj	through
SFO	control	crossing	short	crossed	some	txwyb	landing	cross
ZZZ	radio	time	proceeded	while	way	any	i	approximately
EWR	tower	landing	aircraft	txwyp	rwy22r	between	high	rwy4l
ATL	cross	crossing	roll	speed	high	hold	txwyd	knot
LGA	txwyb	aircarrier	instrumentlandingsystem	off	lights	txwyp	behind	error
LAS	after	saw	rwy25r	procedure	lights	approximately	never	signs
PIT	via	txwye	intersection	firstofficer	conversation	night	looking	down
HOU	txwyme	hold	rwy12r	takeoff	via	around	trying	little
BWI	turn	intersection	taxi	mistake	ground	gate	made	crossed
CYYZ	line	short	stopped	hold	past	taxi	full	end
SEA	feet	firstofficer	cross	tower	read	after	called	back
JFK	prior	instructed	departure	report	his	being	out	txwya

Table 3.14: Images of airports via Naïve Bayes classification, using the binary term by document data.

Airport	term 1	term 2	term 3	term 4	term 5	term 6	term 7	term 8
CLE	Rwy23L	Rwy24L	Rwy24C	Rwy23R	Rwy5R	Line	Rwy6R	Rwy5L
DFW	Rwy35C	Rwy35L	Rwy18L	Rwy17R	Rwy18R	Rwy17C	cross	Tower
ORD	Rwy22R	Rwy27R	Rwy32R	Rwy27L	Rwy32L	Rwy22L	Rwy9L	Rwy4L
MIA	Rwy9L	TxwyQ	Rwy8R	Line	Rwy9R	PilotInCommand	TxwyM	Takeoff
BOS	Rwy4L	Rwy33L	Rwy22R	Rwy4R	Rwy22L	TxwyK	Frequency	Captain
LAX	Rwy25R	Rwy25L	Rwy24L	Rwy24R	Speed	cross	Line	Tower
STL	Rwy12L	Rwy12R	Rwy30L	Rwy30R	Line	cross	short	TxwyP
PHL	Rwy27R	Rwy9L	Rwy27L	TxwyE	amass	TxwyK	AirCarrier	TxwyY
MDW	Rwy31C	Rwy31R	Rwy22L	TxwyP	Rwy4R	midway	Rwy22R	TxwyY
DCA	TxwyJ	Airplane	turn	Captain	Line	Traffic	Landing	short
SFO	Rwy28L	Rwy28R	Rwy1L	Rwy1R	Rwy10R	Rwy10L	b747	Captain
ZZZ	hangar	radio	Rwy36R	gate	Aircraft	Line	Ground	Tower
EWR	Rwy22R	Rwy4L	Rwy22L	TxwyP	TxwyZ	Rwy4R	papa	TxwyPB
ATL	Rwy26L	Rwy26R	Rwy27R	Rwy9L	Rwy8R	atlanta	dixie	cross
LGA	TxwyB4	ILS	Line	notes	TxwyP	hold	vehicle	Taxiway
LAS	Rwy25R	Rwy7L	Rwy19L	Rwy1R	Rwy1L	Rwy25L	TxwyA7	Rwy19R
PIT	Rwy28C	Rwy10C	Rwy28L	TxwyN1	TxwyE	TxwyW	Rwy28R	TxwyV
HOU	Rwy12R	Rwy12L	citation	Takeoff	Heading	Rwy30L	Line	Tower
BWI	TxwyP	Rwy15R	Rwy33L	turn	TxwyP1	Intersection	TxwyE	Taxiway
CYYZ	TxwyQ	TxwyH	Rwy33R	Line	YYZ	Rwy24R	short	toronto
SEA	Rwy34R	Rwy16L	Rwy34L	Rwy16R	AirCarrier	FirstOfficer	TxwyJ	SMA
JFK	Rwy31L	Rwy13R	Rwy22R	Rwy13L	vehicle	Rwy4L	amass	Rwy31R

Table 3.15: Images of airports via LASSO regression, using TF-IDF data.

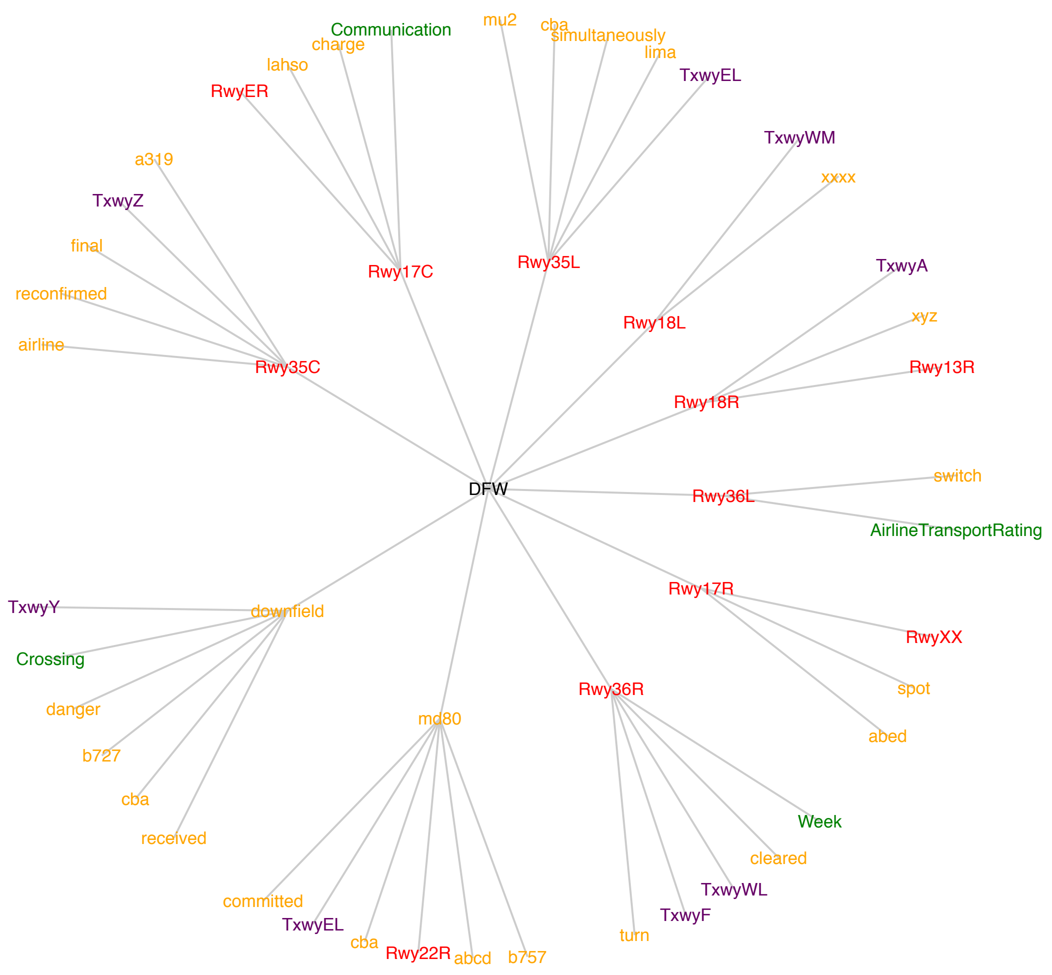


Figure 3.5: A tree LASSO analysis of the DFW airport, showing the LASSO image (inner circle) and for each term in that image, a further image.

terms. We first run a LASSO algorithm to discover a short list of terms that correspond to the image of the term "DFW" in the data set. For each term in that image, we re-run a LASSO analysis, comparing all the documents in the DFW-related corpus containing the term, against all the other documents in the DFW-related corpus. Hence the second step in this analysis only involves the ASRS reports that contain the term "DFW". The approach produces a tree-like structure that can be visualized as two concentric circles of terms, as in Figs. 3.5 and 3.6.

The tree analysis, which is visualized in Figs. 3.5 and 3.6, highlights which issues are pertaining to specific runways, and where *attention* could be focussed. In the airport diagram in Figure 3.7, we have highlighted some locations discussed next.

As highlighted in red in the airport diagram 3.7, the major runway 35L crosses the taxiway EL; likewise for runway 36R and its siblings taxiway WL and F. Runway/taxiway intersections are generally known to contain a risk of collision. At those particular

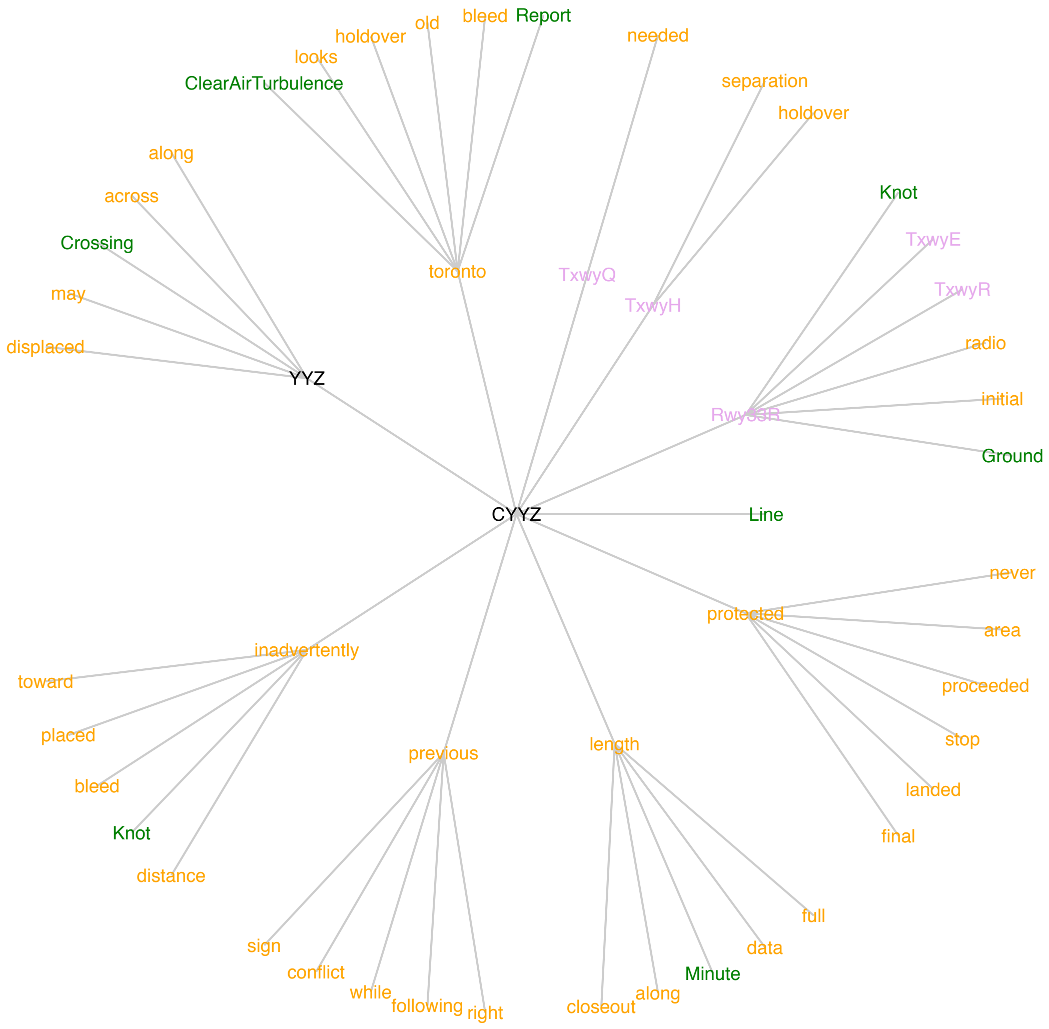


Figure 3.6: A tree LASSO analysis of the CYYZ airport, showing the LASSO image (inner circle) and for each term in that image, a further image.

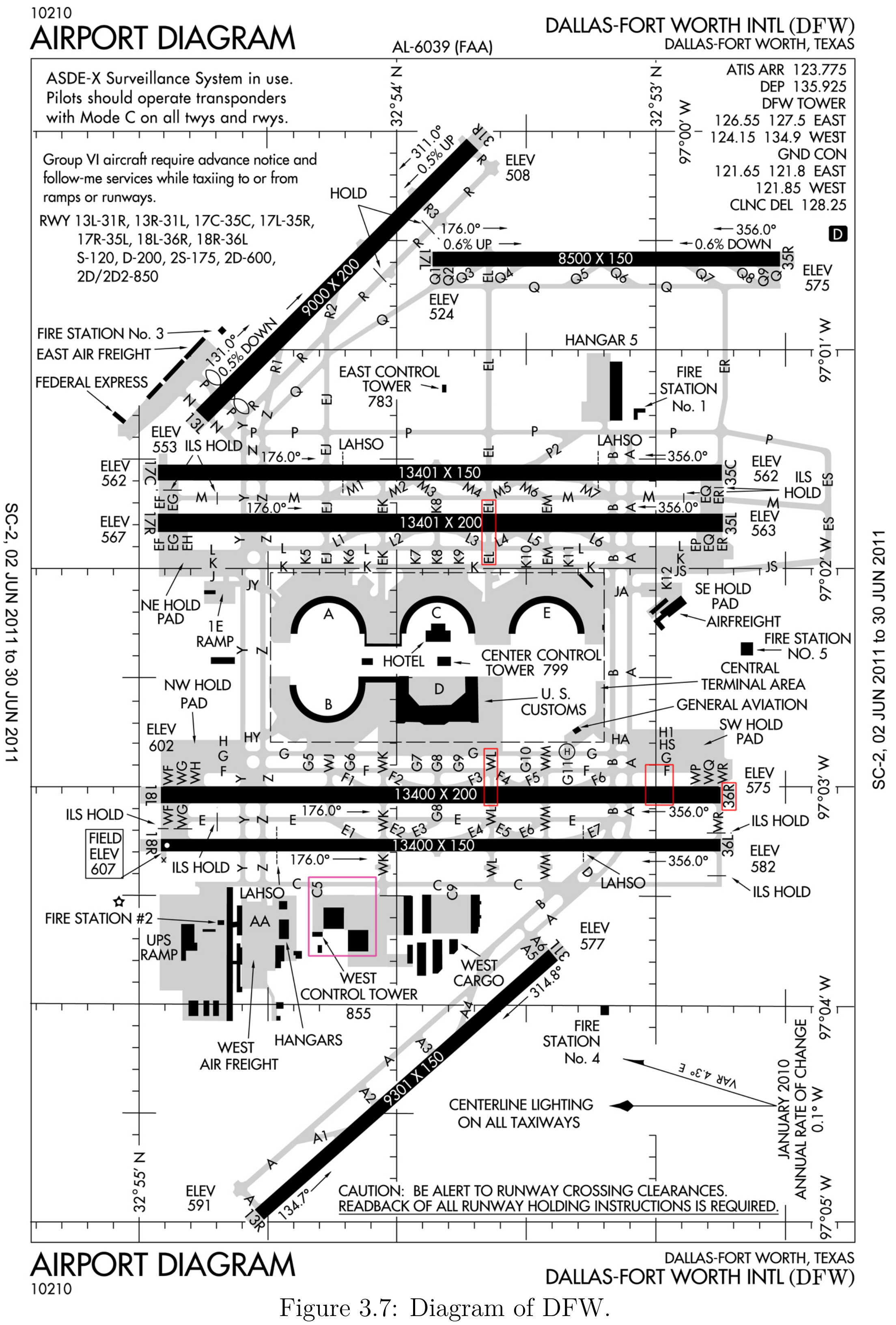


Figure 3.7: Diagram of DFW.

intersections, the issues seem to be about obtaining "clearance" to "turn" from the tower, which might be due to the absence of line of sight from the tower (here we are guessing that the presence of the west cargo area could be a line-of-sight hindrance). The corresponding tree image in Fig. 3.5 is consistent with the location of DFW in the sparse PCA plot (Fig. 3.3), close to the themes of *Aviate* and *Communicate*.

3.6 Summary

Sparse learning problems are formulated as optimization problem with explicit encoding of sparsity requirements, either in the form of constraint or via a penalty on the model variables. This encoding leads to a higher degree of interpretability of the model without penalizing, in fact *improving*, the computational complexity of the algorithm. As such, the results offer an explicit trade-off between accuracy and sparsity, based on the value of the sparsity-controlling parameter that is chosen. In comparison to thresholded PCA, LDA or similar methods, which provide "after-the-fact" sparsity, sparse learning methods offer a principled way to explicitly encode the trade-off in the optimization problem. Thus, the enhanced interpretability of the results is a direct result of the optimization process. We demonstrated the sparse learning techniques on a real-world data set from the Aviation Safety Reporting System and showed that they can reveal contributing factors to aviation safety incidents such as runway incursions. We also show that the sparse PCA and LASSO algorithms can discover the underlying task hierarchy that pilots perform. We have compared the LDA and sparse PCA approaches on other commonly used data sets. Our experiments indicate that the sparse PCA and LASSO methods are very competitive with respect to thresholded methods (involving say LDA and naïve Bayes), at very moderate computational cost. One question is, can we use sparse optimization in other practical areas beyond data mining or text analytics as well? To answer this question, in the next chapter, we will explore the realm of dynamical systems and evaluate the use of sparse methods in complex engineering applications.

Chapter 4

Sparse optimization in energy systems

In the previous chapter, we discuss how text analytics can benefit from sparse machine learning. This chapter moves beyond to explore another domain which, despite often being very technical in nature, still can inherit sparse model ideas. We will study, analyze and then uncover structures in engineering models with real energy applications.

Many engineering applications involve data simulation, system modeling, and parameter optimization. Optimal control for linear dynamical system, or sum-of-square optimization models for complex nonlinear control systems are classical examples in the literature. Recent advances in energy modeling show an important role in energy management applications, where reliable modeling and consumption forecasting are indispensable in engineers' toolbox. Examples include HVAC system optimization, energy prediction, and building's climate control.

A typical engineering optimization commences with an initial sampling of the design space, using an expensive simulation. Then a surrogate model is constructed to model the changes in an objective function in the magnitude of the design variables. Comprehensive reviews for the state-of-the-art surrogate modeling can be found in [87, 101].

In recent years, there has been an extensive body of literature in data-driven approach to engineering modeling. Recent advances are applications of machine learning and statistical learning in this domain, including the use of support vector regression [56], Gaussian process regression [64], Kriging [42], and artificial neural network [66].

Our objective in this chapter is to present sparse optimization-friendly models to predict building energy consumption. When the models achieve high accuracy for the modeling task, one can use these surrogate model to optimize the system parameters in order to minimize quantities of energy interest. We study in particular the posynomial and signomial models, which have in the literature many applications in various engineering design problems, such as aerospace engineering [55], antennas design [8], and circuit design [2, 17, 25].

4.1 Sparse surrogate model

4.1.1 Posynomial model

Posynomials, suggested by Richard Duffin, Elmor Peterson, and Clarence Zener [34] is a combination of polynomial and positive, hence the term. One distinction is posynomials may contain not only integer exponents but also real ones. Mathematically, posynomial model is a function $\psi_{c,\alpha}(p)$ of parameter $p \in \mathbf{R}_+^{n_p}$ that has the form:

$$\psi_{c,\alpha}(p) := \sum_{i=1}^{n_c} \left(c_i \prod_{j=1}^{n_p} p_j^{\alpha_{ij}} \right)$$

where $\alpha_{ij} \in \mathbf{R}$ and all coefficients c_i's are nonnegative.

Much of existing efforts focuses on identifying posynomial models for engineering applications [25, 41, 86]. Recently Calafiore et al. [19] suggest a sparse identification approach, given prior information on the exponents $\alpha_{ij} \in Q_j$, where Q_j is a set of exponents considered for the variable p_j. The set of all possible exponent vectors is then defined as:

$$S := \{\alpha_i\}_{i=1}^n = Q_1 \times Q_2 \times \cdots \times Q_{n_p}.$$

The authors propose the nonnegative square-root Lasso for identifying the sparse posynomial model, given m data or simulation points $\{p(i), y(i)\}_{i=1}^m$:

$$\min_{c \geq 0} \quad \left\| \begin{bmatrix} \Phi c - y \\ \sigma c \end{bmatrix} \right\|_2 + \lambda \|c\|_1 \tag{4.1}$$

with Φ is the matrix constructed from all simulation parameters and possible exponent set:

$$\Phi := \begin{bmatrix} p(1)^{\alpha_1} & \cdots & p(1)^{\alpha_n} \\ \vdots & \ddots & \vdots \\ p(m)^{\alpha_1} & \cdots & p(m)^{\alpha_n} \end{bmatrix} \in \mathbf{R}^{m \times n}.$$

We observe that problem (4.1) is a convex program, and can be solved using a convex solver (e.g. CVX, SDPT3, or MOSEK). When the constructed matrix Φ is large, especially when there are many possible exponent vectors $\{\alpha_i\}_{i=1}^n$, we can employ the safe feature elimination (SAFE) procedure to reduce the size of the optimization problem [39].

4.1.2 Signomial model

Signomial model is an extension of posynomials, also proposed by Duffin and Peterson [35]. We can represent in the form of difference of posynomials, i.e.:

$$\begin{aligned} \zeta(p) &:= \psi_{a,\alpha}(p) - \psi_{b,\beta}(p) \\ &= \textstyle\sum_{i=1}^{n_c} \left(a_i \prod_{j=1}^{n_p} p_j^{\alpha_{ij}} \right) - \sum_{i=1}^{n_d} \left(b_i \prod_{j=1}^{n_p} p_j^{\beta_{ij}} \right) \end{aligned}$$

where a_i and b_i are all non-negative. Similar to the posynomial identification method, we note that, given prior information Q_j's, the sparse identification problem for $\zeta(p)$ can be found by solving a standard square-root Lasso [12]:

$$\min_{c} \quad \left\| \begin{bmatrix} \Phi c - y \\ \sigma c \end{bmatrix} \right\|_2 + \lambda \|c\|_1 \tag{4.2}$$

This unconstrained second-order cone problem can be solved using the interior point method [74], a convex solver, or using alternating direction method of multipliers algorithm.

4.2 Parameter optimization

In many engineering design applications, sparse models are useful in understanding system structure. Examples include discovering variables or groups of variables that contribute to the interested quantities. In system design, however, it's often desirable to discover the optimal system parameters to minimize an objective function, such as optimizing energy consumption. With surrogate optimization-friendly models, we can employ various optimization schemes to find the optimal parameters. Furthermore, having identified an interested local region, we can re-sample our data in a refined region to build better surrogate local models. We discuss in this section techniques for parameter optimization with sparse models presented in Section 4.1.

4.2.1 Posynomial model

The posynomial model has the form of a sum of monomials. Given a feasible set of parameters $\mathcal{P}$, the parameter optimization problem writes:

$$\begin{array}{ll} \min\limits_{p} & \sum_{i=1}^{n_c} \left(c_i \prod_{j=1}^{n_p} p_j^{\alpha_{ij}} \right) \\ \text{s.t.} & p \in \mathcal{P} \end{array} \tag{4.3}$$

If the original parameters are in positive domain, problem (4.3) can be convexified by taking a logarithm transformation on the original parameters:

$$\begin{array}{ll} \min\limits_{z} & \sum_{i=1}^{n_c} c_i \exp\left(\alpha_i^T z \right) \\ \text{s.t.} & z \in P' \end{array} \tag{4.4}$$

where $z := \log(p)$ is the transformed variable and P' is the transformed set of P. Thus problem (4.4) is convex if this transformed set is convex.

4.2.2 Signomial model

The parameter optimization problem for signomial models has a similar form to problem (4.3), except that signomial models have a concave term in the tranformed problem. We write this optimization in logarithm scale as:

$$\begin{array}{ll} \min\limits_{z} & \sum_{i=1}^{n_c} a_i \exp\left(\alpha_i^T z\right) - \sum_{i=1}^{n_d} b_i \exp\left(\beta_i^T z\right) \\ \text{s.t.} & z \in P' \end{array} \tag{4.5}$$

Problem (4.5) is non-convex even when the set P' is convex. Global optimization techniques, approximation algorithms, non-linear optimization methods have been proposed for non-convex problems [2, 80, 100]. In this chapter, we consider a simple heuristic to find a local solution to the difference of convex functions program, known as the convex-concave procedure [65, 69, 106].

The procedure involves linearizing the concave part $\sum_{i=1}^{n_d} b_i \exp\left(\beta_i^T z\right)$ around the current solution z_0, and perform the solution update with convex optimization techniques assuming the feasible set P' is convex. The optimal parameter z^* is selected after this local procedure is repeated T times for different random initializations. We outline the pseudo-code of this algorithm below:

```
for t := $1, 2, 3, \dots,$ T do
    Randomly initialize $z_0 \in P'$.
    for $k := 1, 2, 3, \dots,$ K do
        $z^* := \underset{z \in P'}{\text{argmin}} \; \sum_i a_i \exp\left(\alpha_i^T z\right) - \sum_j b_j \left(\exp\left(\beta_j^T z_0\right) + \exp\left(\beta_j^T z_0\right) \beta_j^T (z - z_0)\right)$
        $z_0 := z^*$
    end
    Compare $z^*$ with the current best parameter.
end
```

Algorithm 3: The convex-concave procedure for signomial models.

Remark. The optimization problems (4.4) and (4.5) aim to find the optimal parameter with respect to the surrogate models. In practice, the engineering design task with a set of parameters often incurs construction cost, for instance building wall thickness for heat insulation. Therefore, one may be interested in optimizing a weighted combination of the surrogate loss and chosen construction cost. We will revisit this idea with a real-life example in Section 4.3.

4.2.3 Iterative sampling and optimization

Even with a broad class of surrogate models, modeling a complex energy system may not always give sufficiently high accuracy for optimizing real-life design of complex structure. We consider in this chapter an iterative approach where data sampling, surrogate modeling, and parameter optimization are carried out in iterations. The key idea is to repeatedly narrow down the search space for the design variables, build a better-fit local

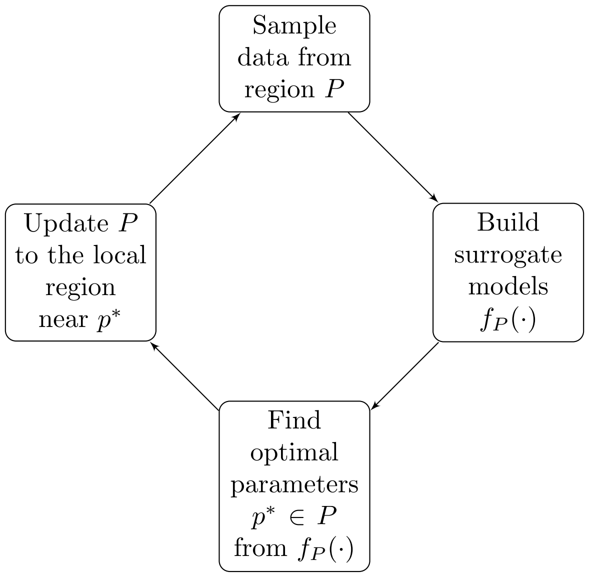

Sample data from region P
Build surrogate models $f_P(\cdot)$
Find optimal parameters $p^* \in P$ from $f_P(\cdot)$
Update P to the local region near p^*

Figure 4.1: Iterative sampling and optimization.

surrogate model, and find the optimal parameters in the local region. The diagram in Figure 4.1 summarizes the key steps of this approach.

4.3 Real-life examples

4.3.1 NACA 4412 Airfoil

We first evaluate the sparse models using the NACA 4412 Airfoil dataset [19], of which the main goal is to model the drag force from 4 parameters: air flow density, the wing chord, the incident angle and the flow velocity. This data set is obtained via costly simulations based on computational fluid dynamics by integrations of the Navier-Stokes equations. In this model fitting example, we use the leave-one-out cross validation procedure over 50 data points as described in [19], and we also use the same metric of accumulated relative error AE $:= \sqrt{\sum_j \nu_j^2}$, where:

$$\nu_j := \frac{|y\,(j) - \hat{y}\,(j)|}{\|y_{\mathrm{LOO}}\|}$$

for every j in the validation set. As Table 4.1 shows, the posynomial model and standard support vector regression ν-SVR [22] have similar error, while the signomial model has better performance, reducing the error by a factor of 2. This is because the signomial function class is much broader than the posynomial function class.

NACA 4412 Airfoil	SVR	Posynomial model	Signomial model
Accumulated error	27.05%	24.72%	**13.43%**

Table 4.1: Drag force model fitting on NACA 4412 airfoil data set.

	Meaning
p_1	Insulation thickness
p_2	Cement coating thickness
p_3	Percent of thermal bridges taken into account
p_4	Solar gain for walls
p_5	Solar gain for glazing

Table 4.2: 5 parameters of interest at EDF.

4.3.2 EDF 22 buildings

In this example, we experiment on a real-life engineering example of modeling complex energy system. The data is obtained from Dymola simulations of 22 buildings at Électricité de France (EDF). Our goal is to understand the relationship between variables and energy quantities like power consumption and temperature deviation from a nominal set point. From the surrogate models, we also propose an optimal parameter to minimize the power consumption of the electric system.

4.3.2.1 Data simulation

We first perform Dymola simulation on 22 buildings at EDF, each of which was simulated in 672 time steps with $\Delta t = 3,600$ seconds. The final time for each simulation was 28 days. The initial set point temperature for all 22 buildings was set to $T_{\text{sp}} := 293.15$ (room temperature). We collected $3,174$ simulations over a Latin hypercube sampling of 5 normalized parameters in Table 4.2; all normalized parameters are between 0 and 1. For each simulation, we recorded the power consumption $P_i(t)$ and temperature $T_i(t)$ of each building i over 28 days.

4.3.2.2 Model fitting

The quantities of interest in EDF data are the average power consumption and the average temperature deviation from the nominal set-point temperature of all buildings. We split half of the simulation data into training and the other half into validation sets. Our criterion to evaluate models is the relative error, defined as:

$$\text{RE} := \frac{\left\|E - \hat{E}\right\|_2}{\|E\|_2}$$

where $\hat{E} \in \mathbf{R}^n$ is the model prediction and $E \in \mathbf{R}^n$ is the true response vector of an energy quantity of interest.

Power	SVR - linear kernel	SVR - RBF kernel	Posynomial model	Signomial
Train RE	42.72%	43.63%	35.94%	**4.53%**
Validation RE	43.96%	44.89%	36.49%	**5.01%**

Table 4.3: Model fitting results with relative error on power consumption.

Temperature	SVR - linear kernel	SVR - RBF kernel	Posynomial model	Signomial
Train RE	16.40%	10.06%	20.81%	**3.96%**
Validation RE	16.96%	10.86%	21.25%	**4.22%**

Table 4.4: Model fitting results with relative error on temperature deviation.

We perform experiments on the signomial model, posynomial model, and support vector regression. All model hyper-parameters are selected via cross-validation based on the scree plot of training and validation data. For the posynomial and signomial models, we set the prior information Q_j for possible exponents of variable j to $Q_j := \{-1, -0.9, \ldots, 0.9, 1\}$.

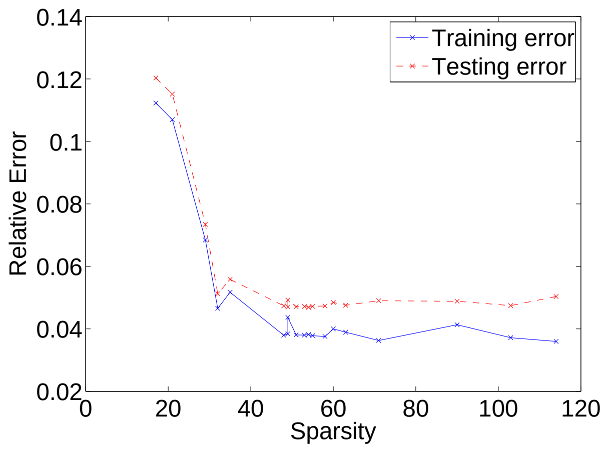
Table 4.3 shows the performance of all models on modeling power consumption. and Table 4.4 shows the results for temperature deviation modeling. We observe that the signomial model has much higher accuracy compared to the posynomial model and the regression model, both of which have similar order of relative error. The model is also very sparse; low relative errors above can be achieved using less than 40 monomials. Figure 4.2 shows the Pareto optimal curve between the model sparsity and relative error of the signomial model for both power and temperature modeling.

4.3.2.3 Parameter optimization

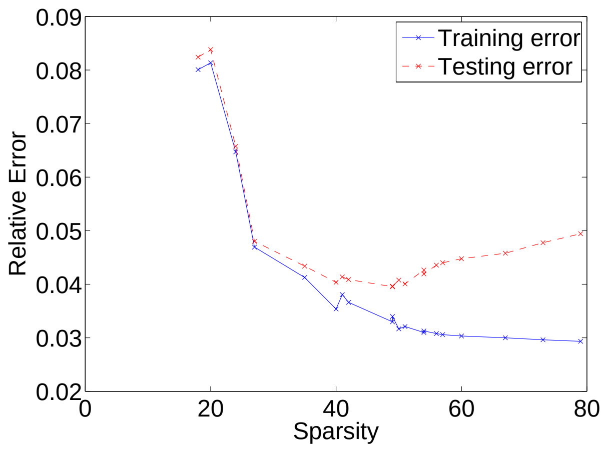
With the signomial model, we aim to find the optimal parameters to minimize the power consumption of 22 EDF buildings. The feasible set for all normalized parameters is $\mathcal{P} := [0,1]^5$, thus the transformed set $P' := \{z : \ z \leq 0\}$ is convex. We also use the construction-cost/regularization term in the form of a log-barrier function with a regularization constant $\gamma \geq 0$ to obtain the optimal solution path $z^*(\gamma)$. Here we assume the upper bounds correspond to more costly construction:

$$z^*(\gamma) := \underset{z \leq 0}{\operatorname{argmin}} \sum_{i=1}^{n_c} c_i \exp\left(\alpha_i^T z\right) - \sum_{i=1}^{n_d} d_i \exp\left(\beta_i^T z\right) - \gamma \sum_k \log(-z_k) \tag{4.6}$$

Figure 4.3 shows the solution path of the original parameters $p^*(\gamma)$ computed from the convex-concave procedure (4.6). We observe that when there's no construction cost ($\gamma = 0$), most variables are near the upper bound. When the construction cost carries more weight ($\gamma > 0$), the variables change at different rates, with variable p_4 (solar gain for walls) decreasing the fastest, while both variable p_1 (insulation thickness) and p_5 (solar gain for glazing) have a much slower rate.



(a) Pareto optimal curve for power consumption



(b) Pareto optimal curve for temperature deviation

Figure 4.2: Signomial model sparsity versus relative error in power and temperature modeling.

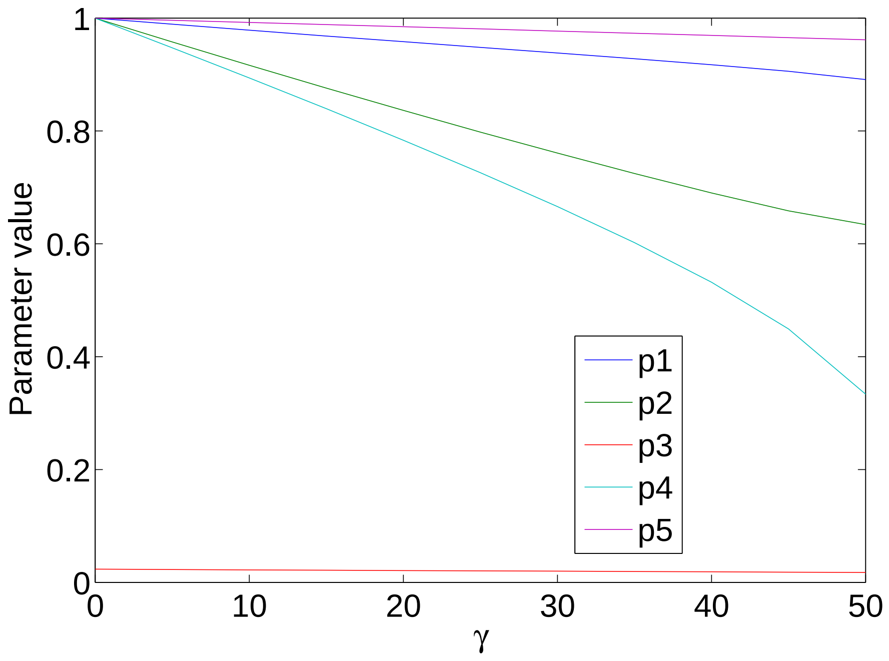
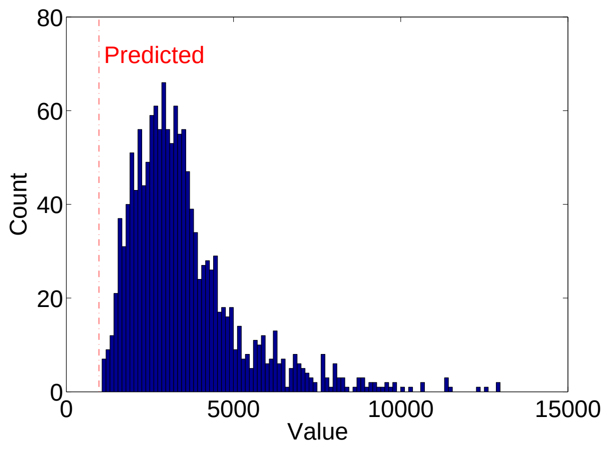
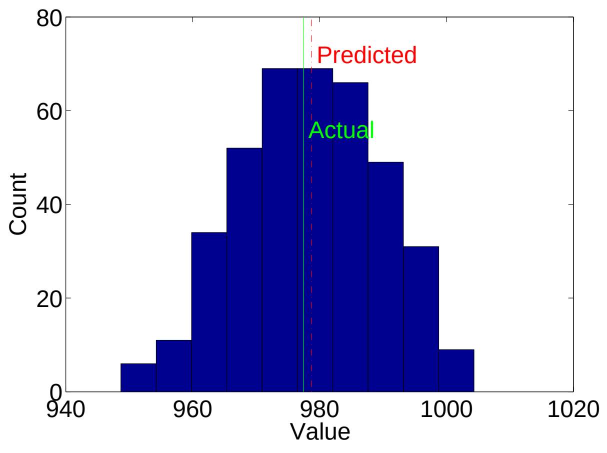


Figure 4.3: Solution path $p^*(\gamma)$ as the regularization parameter γ increases.



(a) Power consumption on the original data



(b) Power consumption on the local region $\hat{P}$

Figure 4.4: Histogram of power consumption and the predicted and actual value at $p^* (\gamma_0)$.

4.3.3 Refined fitting

We further experiment on refined sampling in the vicinity of an optimal parameter $p^* (\gamma_0)$ using Dymola simulation. The power consumption predicted from our signomial model at $p^* (\gamma_0)$ is shown in Figure 4.4a. We perform Latin Hypercube sampling in the local region $\hat{P}$ around $p^* (\gamma_0)$, defined as $\hat{P} := [p^* (\gamma_0) - \sigma\mathbf{1}, p^* (\gamma_0) + \sigma\mathbf{1}]$ with $\sigma = 1\%$. Figure 4.4b shows the histogram of power consumption of refined sampling data in $\hat{P}$ and the actual power consumption simulated at $p^* (\gamma_0)$; we observe that the actual value is very close (less than 1% error) to the signomial model's prediction.

We subsequently perform signomial model fitting on $\hat{P}$. As Table 4.5 shows for all 22 buildings at EDF, the relative error of the signomial model in the local region is an order of magnitude lower than all other methods. All errors are less than 1%, which suggests that this local model can be useful in analyzing the behavior of the complex energy system in the region of interest.

4.4 Summary

This chapter studies an approach to model a complex energy system with sparse optimization. Our approach involves model fitting with sparse models and parameter optimization with the convex-concave procedure, taking into account the construction cost. We show that the sparse model achieves high accuracy in terms of modeling quantities of energy interest, compared to other state-of-the-art models. The parameter optimization procedure suggests an optimal parameter, of which model prediction and actual value from the system are very close to each other. We experiment refined fitting in the vicinity of the optimal parameter, and the high accuracy suggests the sparse model can help understanding the complex energy system in local regions.

Power	SVR	SVR	Posynomial	Signomial	
Building	(linear)	(RBF)	Original fitting	Original fitting	Refined fitting
1	44.67%	45.67%	37.24%	5.16%	**0.59%**
2	42.09%	43.21%	35.83%	4.15%	**0.29%**
3	44.31%	44.76%	37.04%	4.28%	**0.34%**
4	45.96%	47.00%	38.35%	5.47%	**0.76%**
5	42.80%	44.11%	35.76%	5.12%	**0.51%**
6	40.67%	41.49%	32.52%	5.25%	**0.54%**
7	45.35%	46.33%	37.81%	5.32%	**0.65%**
8	45.32%	46.33%	37.78%	5.33%	**0.63%**
9	43.65%	44.59%	36.20%	5.03%	**0.50%**
10	41.28%	42.09%	33.05%	5.41%	**0.61%**
11	43.03%	43.56%	35.17%	4.74%	**0.54%**
12	44.59%	45.62%	37.15%	5.20%	**0.55%**
13	44.12%	45.10%	36.77%	5.03%	**0.50%**
14	44.76%	45.76%	37.15%	5.34%	**0.61%**
15	44.47%	45.49%	37.06%	5.16%	**0.55%**
16	44.65%	45.67%	37.21%	5.18%	**0.58%**
17	44.18%	45.17%	36.81%	5.07%	**0.52%**
18	44.28%	45.21%	36.70%	5.19%	**0.57%**
19	44.76%	45.78%	37.28%	5.23%	**0.56%**
20	46.12%	47.17%	38.49%	5.55%	**0.77%**
21	44.87%	45.90%	37.25%	5.37%	**0.65%**
22	46.53%	48.00%	40.87%	4.74%	**0.15%**

Table 4.5: Relative error on validation set of refined fitting and original fitting

Chapter 5

Robust sketching for sparse models

We have presented applications of sparse models in domains of text analytics and energy modeling in previous chapters. This chapter on the other hand approaches a standard sparse model from a theoretical standpoint. From previous examples, we observe that in many applications, learning tasks arise not in isolation, but in multiple instances that share most, if not all of the data. A simple classical example is in computing the full regularization path or in parameter search [46, 53, 83]. In this problem, the main task is to compute the full solutions under different tuning parameters while the "design matrix" stays unchanged.

Consider another classical example: leave-one-out cross-validation. At each step a single instance is being held-out and the learning problem is to be solved with the remaining "design matrix". This procedure is repeated m times, where m is the number of observations, which suggests that a computationally efficient method is essential to make cross-validation feasible.

Many practical applications are also in the same setting: examples include sparse covariance estimation with the LASSO [47], or in robust subspace clustering [92]. Both require solving m instances of the LASSO; each instance differs by one example or one feature being left out. Most of the data is shared among all instances. Hence it makes sense to spend preprocessing time on the common part of the problems to compress it in certain sense, and in the big picture speed up the overall computation.

5.1 Robust sketching overview

In this section we propose an approach to multi-instance square root LASSO based on "robust sketching", where the data matrix of the optimization problem is approximated by a sketch — a simpler matrix that preserves some property of interest — and on which computation can be performed much faster than the original. Our focus is the square-root LASSO problem:

$$\min_{w \in \mathbf{R}^n} \|X^T w - y\|_2 + \lambda \|w\|_1$$

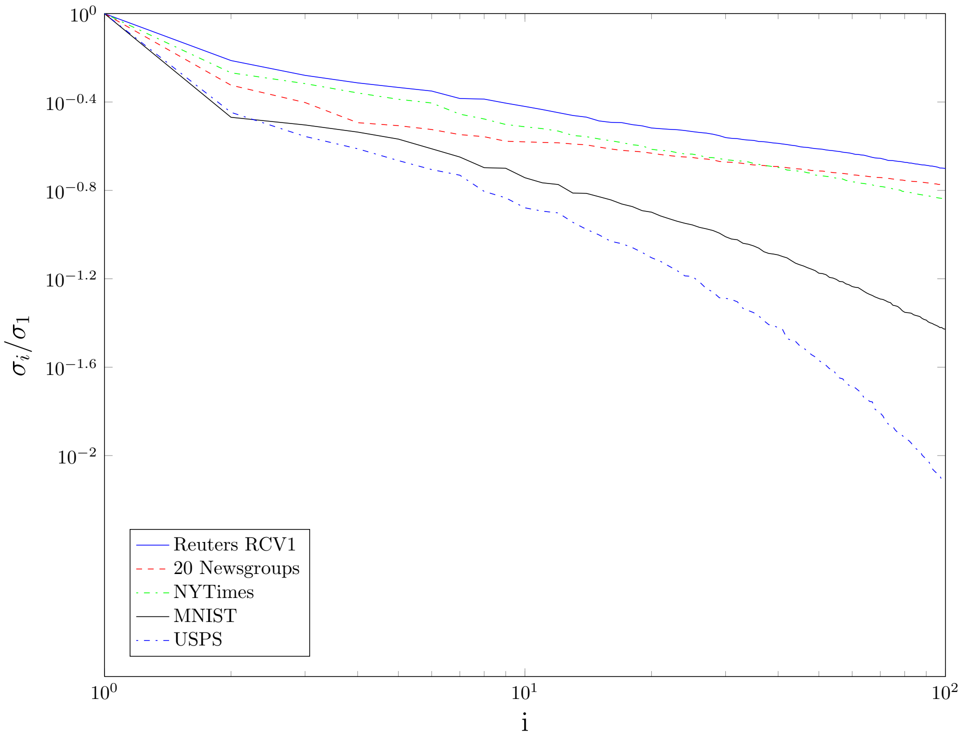


Figure 5.1: Graphs of the top 100 singular values from real-life text and digit recognition data sets.

with $X \in \mathbf{R}^{n\times m}$ and $y \in \mathbf{R}^m$. The square-root LASSO problem has pivotal recovery properties; also solving a square-root LASSO problem is as fast as solving an equivalent LASSO problem with both first-order and second-order methods [11]. We chose the square-root version to make the derivation simpler; these derivations can also be adapted to the original LASSO, in which the loss function is squared.

In practice, real-life data sets often have very large number of observations m and number of features n. Nevertheless, a key observation is that high-dimensional data often have special structure we can exploit in many learning tasks. Figure 5.1 shows large-scale data sets are often close to a low-rank sketch. The linear decay in log-log plot of singular values suggests these data can be well-approximated by low-rank structures. Furthermore, in many applications, including signal processing, speech encoding, or image processing, low rank approximation is also known as a common tool for noise removal (see [24] and references therein).

Objective. Our goal is to solve multiple instances of a learning model fast, each instance being a small modification to the same design matrix. Our approach is to first spend computational effort in finding a low-rank sketch of the full data. With this sketch, we propose a robust model that takes into account the approximation error, and explain how to solve that approximate problem faster, in effect reducing m, the number of observations, to k, the number of singular values retained in our low-rank model. This approach leads to a dramatic reduction in computational effort: with the low-rank model, we can perform cross-validation, for example, an order of magnitude faster, with the low-rank sketching included in our approach.

There has been an extensive body of literature on sketching, including power method, random projection, random sampling, and Nyström method [32, 33, 52, 67, 72, 76]. Our framework works with any approximation algorithms, thus provides flexibility when working with different sketching methods, deterministic or random, and remains highly-scalable in learning tasks.

Computing the full regularization path is an example of multiple-instance setting that has been widely studied in the literature [9, 46, 83]. The most popular approach is the warm-start technique, which was first proposed in a specific optimization algorithm for linear program [104], then applied in many statistical learning models [61, 62]. Recent works [99] show strong interest in incremental and decremental training, employing the same warm-start strategy. These techniques are very specific to the learning task at hand, and often require special handling of the underlying algorithm for each case.

Our framework, on the other hand, does not go into specific algorithms. In our approach, we propose a generic, robust, algorithm-independent model for solving multiple LASSO problems fast. Our model can be implemented with any generic convex solver, providing guarantees for computational savings while not sacrificing the statistical learning performance.

5.2 Robust sketching with square-root LASSO

In this section, we study the square-root LASSO problem [11] given m observations, each having n features:

$$\min_{w\in\mathbf{R}^n} \|X^Tw-y\|_2+\lambda\|w\|_1 \tag{5.1}$$

where $X\in\mathbf{R}^{n\times m}$ and $y\in\mathbf{R}^m$. We first consider the square-root LASSO problem when a sketch of the original data matrix is available, then solve our robust sketching problem fast, and analyze a non-robust approach to sketching.

5.2.1 Robust square-root LASSO

Assume we are given $\hat{X}$ as a sketch to the data matrix X, the robust square-root LASSO is defined as follows:

$$\min_{w\in\mathbf{R}^n}\ \max_{X:\|X-\hat{X}\|_2\le\epsilon}\|X^Tw-y\|_2+\lambda\|w\|_1 \tag{5.2}$$

The authors of [37] have shown that

$$\begin{aligned}\max_{X:\|X-\hat{X}\|_2\le\epsilon}\|X^Tw-y\|_2 &= \max_{\|\Delta\|_2\le\epsilon}\|\left(\hat{X}+\Delta\right)^T w-y\|_2\\ &\le \|\hat{X}^Tw-y\|_2+\epsilon\|w\|_2.\end{aligned}$$

This bound is tight with the choice of Δ as

$$\begin{aligned}\Delta &:= \epsilon uv^T\\ u &:= \begin{cases}\frac{\hat{X}^Tw-y}{\|\hat{X}^Tw-y\|_2} & \text{if } \hat{X}^Tw\neq y\\ \text{any unit-norm vector} & \text{otherwise}\end{cases}\\ v &:= \begin{cases}\frac{w}{\|w\|_2} & \text{if } w\neq 0\\ \text{any unit-norm vector} & \text{otherwise}\end{cases}\end{aligned}$$

We also have $\mathbf{rank}(\Delta)=1$ and $\|\Delta\|_F=\|\Delta\|_2$ which implies Δ is the worse-case pertubation for both the Frobenius norm and the operator norm. Problem (5.2) can be rewritten as:

$$\phi_{\lambda,\epsilon}=\min_{w\in\mathbf{R}^n}\|\hat{X}^Tw-y\|_2+\epsilon\|w\|_2+\lambda\|w\|_1 \tag{5.3}$$

Note the presence of an "elastic net" term proposed by [112], directly imputable to the error on the original design matrix X.

In our framework, we employ low-rank approximation from any sketching algorithm: $X\approx\hat{X}=PQ^T$, where $P\in\mathbf{R}^{n\times k}, Q\in\mathbf{R}^{m\times k}$, P and Q full rank with rank $k\ll\min\{m,n\}$. With an efficient method to solve the robust low-rank problem, for leave-one-out analysis we can quickly compute the "design matrix" $X_{\setminus i}\approx PQ^T_{\setminus i}$ and efficiently solve a new instance for every observation i being held out. We now turn to a fast solution to the low-rank elastic net problem (5.3).

Theorem 1. *Given a sketching matrix $\hat{X}=PQ^T$, we first compute $c:=(Q^TQ)^{-1/2}Q^Ty\in\mathbf{R}^k$, $s:=\sqrt{y^Ty-c^Tc}\in\mathbf{R}$, $R:=P(Q^TQ)^{1/2}\in\mathbf{R}^{n\times k}$, the dual problem of (5.3) is:*

$$\begin{aligned}\phi_{\lambda,\epsilon} = \max_{\substack{u\in\mathbf{R}^k\\ v,r\in\mathbf{R}^n\\ t\in\mathbf{R}}} \quad & u^Tc+st\\ \text{s.t.}\quad & \left\|\begin{bmatrix}u\\ t\end{bmatrix}\right\|_2\le 1,\ \|v\|_\infty\le\lambda,\ \|r\|_2\le\epsilon,\\ & Ru=v+r\end{aligned} \tag{5.4}$$

Proof. The robust low-rank square-root LASSO is:

$$\phi_{\lambda,\epsilon}=\min_{w\in\mathbf{R}^n}\|\hat{X}^Tw-y\|_2+\epsilon\|w\|_2+\lambda\|w\|_1\ . \tag{5.5}$$

With $\hat{X} = PQ^T$, $P \in \mathbf{R}^{n\times k}, Q \in \mathbf{R}^{m\times k}$, the dual of the convex problem (5.5) is:

$$\begin{aligned}
\phi_{\lambda,\epsilon} &= \min_{w,z\in\mathbf{R}^n} \|Qz-y\|_2 + \epsilon\|w\|_2 + \lambda\|w\|_1 \ : \ z = P^Tw \\
&= \min_{w,z\in\mathbf{R}^n}\max_{u\in\mathbf{R}^k} \|Qz-y\|_2 + \epsilon\|w\|_2 + \lambda\|w\|_1 + u^T(z-P^Tw) \\
&= \max_{u\in\mathbf{R}^k}\min_{w,z\in\mathbf{R}^n} \|Qz-y\|_2 + \epsilon\|w\|_2 + \lambda\|w\|_1 + u^T(z-P^Tw) \\
&= \max_{u\in\mathbf{R}^k} f_1(u) + f_2(u)
\end{aligned}$$

where $f_1(u) := \min_{z\in\mathbf{R}^n}\|Qz-y\|_2 + u^Tz$ and $f_2(u) := \min_{w\in\mathbf{R}^n}\epsilon\|w\|_2 + \lambda\|w\|_1 - (Pu)^Tw$.

First subproblem. Consider the first term in $f_1(u)$:

$$\begin{aligned}
\|Qz-y\|_2^2 &= z^TQ^TQz - 2y^TQz + y^Ty \\
&= \bar{z}^T\bar{z} - 2c^T\bar{z} + y^Ty \\
&\quad \text{where } \bar{z} := (Q^TQ)^{1/2}z \in \mathbf{R}^n \text{ and } c := (Q^TQ)^{-1/2}Q^Ty \in \mathbf{R}^n \\
&= \|\bar{z}-c\|_2^2 + y^Ty - c^Tc.
\end{aligned}$$

Note that $c^Tc = y^TQ(Q^TQ)^{-1}Q^Ty \le y^Ty$ since $Q(Q^TQ)^{-1}Q^T \preceq I$ is the projection matrix onto $\mathbf{range}(Q)$. Letting $s := \sqrt{y^Ty - c^Tc} \ge 0$ gives

$$\begin{aligned}
f_1(u) &= \min_z \|Qz-y\|_2 + u^Tz \\
&= \min_z \sqrt{\|\bar{z}-c\|_2^2 + s^2} + \bar{u}^T\bar{z} && : \text{ by letting } \bar{u} := (Q^TQ)^{-1/2}u \\
&= \bar{u}^Tc + \min_x \sqrt{\|x\|_2^2 + s^2} - \bar{u}^Tx && : \text{ by letting } x := c - \bar{z}.
\end{aligned}$$

Now consider the second term $\min_x \sqrt{\|x\|_2^2 + s^2} - \bar{u}^Tx$. The optimal x^* must be in the direction of $\bar{u}$. Letting $x := \alpha\bar{u}$, $\alpha \in \mathbf{R}$, we have the expression

$$\min_\alpha \sqrt{\alpha^2\|\bar{u}\|_2^2 + s^2} - \alpha\|\bar{u}\|_2^2.$$

When $\|\bar{u}\|_2 \ge 1$, the problem is unbounded below. When $\|\bar{u}\|_2 < 1$, the optimal solution is $\alpha^* = \frac{s}{\sqrt{1-\|\bar{u}\|_2^2}}$ and the optimal value is thus $s\sqrt{1-\|\bar{u}\|_2^2}$. The closed-form expression for $f_1(u)$ is therefore:

$$\begin{aligned}
f_1(u) &= \bar{u}^Tc + \min_x \sqrt{\|x\|_2^2 + s^2} - \bar{u}^Tx \\
&= \bar{u}^Tc + s\sqrt{1-\|\bar{u}\|_2^2} \\
&= u^T(Q^TQ)^{-1/2}c + s\sqrt{1 - u^T(Q^TQ)^{-1}u} \\
&= u^TK^{-1/2}c + s\sqrt{1-u^TK^{-1}u} && : \text{ by letting } K := Q^TQ.
\end{aligned} \tag{5.6}$$

Second subproblem. Consider the function $f_2(u) := \min_{w\in\mathbf{R}^n}\epsilon\|w\|_2 + \lambda\|w\|_1 - (Pu)^Tw.$

We observe that the objective function is homogeneous. Strong duality gives:

$$\begin{aligned}
f_2(u) &= \min_x \max_{v,r} r^T w + v^T w - (Pu)^T w \\
&\quad \text{s.t.} \quad \|r\|_2 \le \epsilon, \|v\|_\infty \le \lambda \\
&= \max_{v,r} \min_x (r + v - Pu)^T w \\
&\quad \text{s.t.} \quad \|r\|_2 \le \epsilon, \|v\|_\infty \le \lambda \\
&= \max_{v,r} 0 \\
&\quad \text{s.t.} \quad \|r\|_2 \le \epsilon, \|v\|_\infty \le \lambda, Pu = v + r
\end{aligned} \tag{5.7}$$

Hence $f_2(u) = 0$ if there exists $v, r \in \mathbf{R}^n$ satisfying the constraints. Otherwise $f_2(u)$ is unbounded below.

Dual problem. From (5.6) and (5.7), the dual problem to (5.5) can be derived as:

$$\begin{aligned}
\phi_{\lambda,\epsilon} = \max_{\substack{u \in \mathbf{R}^k \\ v, r \in \mathbf{R}^n}} \quad & u^T K^{-1/2} c + s\sqrt{1 - u^T K^{-1} u} \\
\text{s.t.} \quad & \|v\|_\infty \le \lambda,\ \|r\|_2 \le \epsilon,\ Pu = v + r
\end{aligned}$$

Letting $R := PK^{1/2} \in \mathbf{R}^{n \times k}$ and replacing u by $K^{-1/2}u$, we have:

$$\begin{aligned}
\phi_{\lambda,\epsilon} &= \max_{\substack{u \in \mathbf{R}^k \\ v, r \in \mathbf{R}^n}} \quad u^T c + s\sqrt{1 - u^T u} \\
&\qquad \text{s.t.} \quad \|v\|_\infty \le \lambda,\ \|r\|_2 \le \epsilon,\ Ru = v + r \\
&= \max_{\substack{u \in \mathbf{R}^k \\ v, r \in \mathbf{R}^n \\ t \in \mathbf{R}}} \quad u^T c + st \\
&\qquad \text{s.t.} \quad \left\| \begin{bmatrix} u \\ t \end{bmatrix} \right\|_2 \le 1,\ \|v\|_\infty \le \lambda,\ \|r\|_2 \le \epsilon, \\
&\qquad\qquad Ru = v + r
\end{aligned} \tag{5.8}$$

□

An important observation is that problem (5.4) is a standard conic problem with linear objective, norm constraints of size n and k, but now the size of the data matrix R is only n-by-k, instead of n-by-m as the original problem. We can thus solve it with any conic solver much more efficiently than the original (5.1). Analyzing this problem gives a relationship between w^* in (5.3) and u^*, v^*, r^*, t^* in (5.4), as the following theorem shows.

Theorem 2. *The optimal variable w^* can be recovered from the optimal dual variables u^*, v^*, r^*, t^* by equation:*

$$w^* = \alpha r^*,$$

where $\alpha, \beta \in \mathbf{R}$ solve

$$c = \alpha \left(R^T r^* \right) + \beta u^*.$$

Also, with strong duality, the dual of the conic problem (5.4) is

$$\phi_{\lambda,\epsilon} = \min_{w \in \mathbf{R}^n} \left\| \begin{bmatrix} c - R^T w \\ s \end{bmatrix} \right\|_2 + \epsilon \|w\|_2 + \lambda \|w\|_1 \ . \tag{5.9}$$

Proof. The dual of problem (5.4) can be derived as:

$$\begin{array}{rcccl} \phi_{\lambda,\epsilon} & = & \underset{u,v,r,t}{\max} & \underset{w \in \mathbf{R}^n}{\min} & u^T c + st + w^T(v + r - Ru) \\ & & \text{s.t.} & & \left\| \begin{bmatrix} u \\ t \end{bmatrix} \right\|_2 \leq 1, \ \|v\|_\infty \leq \lambda, \ \|r\|_2 \leq \epsilon, \\ & = & \underset{w \in \mathbf{R}^n}{\min} & \underset{u,v,r,t}{\max} & \begin{bmatrix} c - R^T w \\ s \end{bmatrix}^T \begin{bmatrix} u \\ t \end{bmatrix} + w^T v + w^T r \\ & & \text{s.t.} & & \left\| \begin{bmatrix} u \\ t \end{bmatrix} \right\|_2 \leq 1, \ \|v\|_\infty \leq \lambda, \ \|r\|_2 \leq \epsilon, \\ & = & \underset{w \in \mathbf{R}^n}{\min} & & \left\| \begin{bmatrix} c - R^T w \\ s \end{bmatrix} \right\|_2 + \epsilon \|w\|_2 + \lambda \|w\|_1 \end{array}$$

From the optimality condition, we have

$$\begin{array}{rcl} w^{*T} r^* & = & \epsilon \|w^*\|_2 \\ \begin{bmatrix} c - R^T w^* \\ s \end{bmatrix}^T \begin{bmatrix} u^* \\ t^* \end{bmatrix} & = & \left\| \begin{bmatrix} c - R^T w^* \\ s \end{bmatrix} \right\|_2 \end{array} .$$

Since $\|r\|_2 \leq \epsilon$ and $\left\| \begin{bmatrix} u \\ t \end{bmatrix} \right\|_2 \leq 1$, we conclude

$$\begin{array}{rcl} w^* & = & \alpha r^* \\ \begin{bmatrix} c - R^T w^* \\ s \end{bmatrix} & = & \beta \begin{bmatrix} u^* \\ t^* \end{bmatrix} \end{array}$$

for some $\alpha, \beta \in \mathbf{R}$, or equivalently, $c = \alpha \left(R^T r^* \right) + \beta u^*$. □

5.2.2 Computational time complexity

In this section, we give a worst-case time complexity analysis on solving robust sketching problem. Our proof technique adapts from that of Andersen, Roos, and Terlaky [7], which analyzes a second-order method via log-barrier functions for conic problems.

Theorem 3. *The worst-case complexity of solving problem (5.4) and (5.9) grows as $O(kn^2 + k^3)$. This is in contrast with the original problem (5.1) when no structure is exploited, in which case the complexity grows as $O(mn^2 + m^3)$.*

Proof. We analyze the standard second-order method via log-barrier functions for the robust square-root LASSO:

$$\min_{x \in \mathbf{R}^n} \|Ax - b\|_2 + \epsilon\|x\|_2 + \lambda\|x\|_1 \tag{5.10}$$

where $A \in \mathbf{R}^{k\times n}$ and $b \in \mathbf{R}^k$.

Square-root LASSO. In second-order methods, the main cost at each iteration is from solving a linear system of equations involving the Hessian of the barrier function [7]. Consider the original square-root LASSO problem:

$$\min_{x \in \mathbf{R}^n} \|Ax - b\|_2 = \min_{x \in \mathbf{R}^n, s \in \mathbf{R}} s \ : \ \|Ax - b\|_2 \le s$$

The log-barrier function is $\varphi_\gamma(x, s) = \gamma s - \log(s^2 - \|Ax - b\|_2^2)$. The cost is from evaluating the inverse Hessian of

$$f := -\log\left(s^2 - (Ax - b)^T(Ax - b)\right) \tag{5.11}$$

Let $g := -\log\left(s^2 - w^Tw\right)$, we have $\nabla g = \frac{2}{-g}\begin{bmatrix} -w \\ s \end{bmatrix}$ and $\nabla^2 g = \frac{2}{-g}\begin{bmatrix} -I & 0 \\ 0 & 1 \end{bmatrix} + \nabla g(\nabla g)^T$. The Hessian $\nabla^2 g$ is therefore a diagonal plus a dyad.

For (5.11), rearranging the variables as $\tilde{x} = \begin{bmatrix} x \\ s \end{bmatrix}$ gives $\begin{bmatrix} Ax - b \\ s \end{bmatrix} = \begin{bmatrix} A & 0 \\ 0 & 1 \end{bmatrix}\begin{bmatrix} x \\ s \end{bmatrix} - \begin{bmatrix} b \\ 0 \end{bmatrix} = \tilde{A}\tilde{x} - \tilde{b}$ where $\tilde{A} \in \mathbf{R}^{(k+1)\times n}$ and:

$$\begin{aligned} \nabla^2 f &= \tilde{A}^T \tfrac{2}{-f}\begin{bmatrix} -(Ax - b) \\ s \end{bmatrix} \nabla^2 f \\ &= \tilde{A}^T\left(\tfrac{2}{-f}\begin{bmatrix} -I & 0 \\ 0 & 1 \end{bmatrix} + \tfrac{4}{f^2}\begin{bmatrix} -Ax + b \\ s \end{bmatrix}\begin{bmatrix} -Ax + b \\ s \end{bmatrix}^T\right)\tilde{A} \\ &= \tfrac{2}{-f}\tilde{A}^T\begin{bmatrix} -I & 0 \\ 0 & 1 \end{bmatrix}\tilde{A} + \tfrac{4}{f^2}\left(\tilde{A}^T\begin{bmatrix} -(Ax - b) \\ s \end{bmatrix}\right)\left(\tilde{A}^T\begin{bmatrix} -(Ax - b) \\ s \end{bmatrix}\right)^T \end{aligned} \tag{5.12}$$

The Hessian $\nabla^2 f$ is therefore simply a $(k + 2)$-dyad.

Regularized square-root LASSO. For (5.10), decomposing $x = p - q$ with $p \ge 0, q \ge 0$ gives:

$$\begin{aligned} \phi &= \min_{x \in \mathbf{R}^n} && \|Ax - b\|_2 + \epsilon\|x\|_2 + \lambda\|x\|_1 \\ &= \min_{\substack{p, q \in \mathbf{R}^n \\ s, t \in \mathbf{R}}} && s + \epsilon t + \lambda\left(\mathbf{1}^T p + \mathbf{1}^T q\right) \\ & \quad \text{s.t.} && \|p - q\|_2 \le t, \ \|A(p - q) - b\|_2 \le s, \\ &&& p \ge 0, q \ge 0 \end{aligned}$$

The log-barrier function is thus:

$$\begin{aligned}\varphi_\gamma(p,q,s,t) &= \gamma\left(s+\epsilon t+\lambda\left(\mathbf{1}^Tp+\mathbf{1}^Tq\right)\right)\\ &\quad -\log\left(t^2-\|p-q\|_2^2\right)\\ &\quad -\log\left(s^2-\|A(p-q)-b\|_2^2\right)\\ &\quad -\textstyle\sum_{i=1}^n\log(p_i)-\sum_{i=1}^n\log(q_i)-\log(s)-\log(t).\end{aligned}$$

First log term. Let $l_1 := -\log\left(t^2-\|p-q\|_2^2\right)$. Rearranging our variables as $\tilde{x} = \left[p_1, q_1, \cdots p_n, q_n, t\right]^T$, we have:

$$\begin{aligned}\nabla l_1 &= \tfrac{2}{-l_1}\left[p_1-q_1, q_1-p_1, \cdots p_n-q_n, q_n-p_n, t\right]^T\\ \nabla^2 l_1 &= \tfrac{2}{-l_1}\begin{bmatrix}B & & & \\ & \ddots & & \\ & & B & \\ & & & 1\end{bmatrix} + \nabla l_1(\nabla l_1)^T\end{aligned}$$

where there are n blocks of $B := \begin{bmatrix}-1 & 1\\ 1 & -1\end{bmatrix}$ in the Hessian $\nabla^2 l_1$.

Second log term. Let $l_2 := -\log\left(s^2-\|A(p-q)-b\|_2^2\right)$. Keeping the same order of variables $\tilde{x} = \left[p_1, q_1, \cdots p_n, q_n, s\right]^T$, we have

$$\begin{bmatrix}A(p-q)-b\\ s\end{bmatrix} = \tilde{A}\tilde{x}$$

where $\tilde{A} \in \mathbf{R}^{(k+1)\times(2n+1)}$. Following (5.12), we have the Hessian is a $(k+2)$-dyad.

Third log term. Let $l_3 := -\sum_{i=1}^n\log(p_i)-\sum_{i=1}^n\log(q_i)-\log(s)-\log(t)$. Every variable is decoupled; therefore the Hessian is simply diagonal.

Summary. The Hessian of the log barrier function $\varphi_\gamma(p,q,s,t)$ is a block diagonal plus a $(k+2)$-dyad. At each iteration of second-order method, inverting the Hessian following the matrix inversion lemma costs $O(kn^2)$. For the original square-root LASSO problem (5.1), using similar methods will cost $O(mn^2)$ at each iteration [7]. □

We note from Theorem (3) that if k is fixed as a constant, solving m instances of the robust problem with structure has the same complexity of solving *one* single instance of the original. This is the key observation for dramatic computational speed-up in multi-instance applications in the sequel.

5.2.3 Safe feature elimination

In this section we present a screening rule to reduce the dimension of problem (5.9) without changing the optimal value. We first propose a simple rule similar to those in

[39], then we will provide an improved safe feature elimination with provided bounds. Let us define $A := \begin{bmatrix} R^T \\ 0 \end{bmatrix} \in \mathbf{R}^{(k+1)\times n}$ and $b := \begin{bmatrix} c \\ s \end{bmatrix} \in \mathbf{R}^{k+1}$ and problem (5.9) can be written as:

$$\min_{w\in\mathbf{R}^n} \|Aw-b\|_2 + \epsilon\|w\|_2 + \lambda\|w\|_1 \tag{5.13}$$

Using the technique in [39], we have the first theorem on safe feature elimination for problem (5.13):

Theorem 4. *Let a_i's be the columns of data matrix A in problem (5.13). If $\|a_i\|_2 \le \lambda - \epsilon$, then we can discard the i-th feature without affecting the optimal value.*

Proof. Problem (5.13) is equal to:

$$\begin{aligned}
& \min_{w\in\mathbf{R}^n} \|Aw-b\|_2 + \epsilon\|w\|_2 + \lambda\|w\|_1 \\
= & \min_{w\in\mathbf{R}^n} \max_{\substack{\|\alpha\|_2\le 1\\ \|\beta\|_2\le\epsilon \\ \|\gamma\|_\infty\le\lambda}} \alpha^T(Aw-b) + \beta^T w + \gamma^T w \\
= & \max_{\substack{\|\alpha\|_2\le 1\\ \|\beta\|_2\le\epsilon \\ \|\gamma\|_\infty\le\lambda}} \min_{w\in\mathbf{R}^n} \alpha^T(Aw-b) + \beta^T w + \gamma^T w \\
= & \max_{\alpha,\beta,\gamma} \ -\alpha^T b \\
& \text{s.t} \quad \|\alpha\|_2 \le 1,\ \|\beta\|_2 \le \epsilon,\ \|\gamma\|_\infty \le \lambda, \\
& \qquad A^T\alpha + \beta + \gamma = 0 \\
= & \max_{\alpha,\beta} \ -\alpha^T b \\
& \text{s.t} \quad \|\alpha\|_2 \le 1,\ \|\beta\|_2 \le \epsilon, \\
& \qquad |a_i^T\alpha + \beta_i| \le \lambda, \forall i = 1\dots n
\end{aligned} \tag{5.14}$$

where a_i's are columns of A. If $\|a_i\|_2 \le \lambda-\epsilon$, we always have $|a_i^T\alpha+\beta_i| \le |a_i^T\alpha|+|\beta_i| \le \lambda$. In other words, we can then safely discard the i-th feature without affecting our optimal value. □

We can further improve this screen rule if we have a lower bound l and an upper bound u on the optimal value. Note that a lower bound l is always non-negative from properties of norms, and an upper bound $u = \|b\|_2$ is from plugging in $w = 0$ in the primal (5.13). In fact, an upper bound can be obtained by any primal variable in (5.13) and a lower bound can be obtained by any feasible dual variable in (5.14).

With these bounds, if the optimization problem:

$$\max_{\alpha} a_i^T\alpha \quad \text{s.t.} \quad \|\alpha\|_2 \le 1,\ l \le \alpha^T(-b) \le u \tag{5.15}$$

has the optimal value at most $\lambda-\epsilon$, we can safely discard the i-th feature. This intuition gives a geometric approach in the following theorem.

Theorem 5. *The i-th feature a_i can be decomposed into two orthogonal components by the fundamental theorem of linear algebra:*

$$a_i = q_i\left(-b\right) + r_i, \quad r_i \perp b.$$

where $q_i := -\frac{a_i^T b}{\|b\|_2^2}$ and $r_i := a_i + q_i b$. Given the bounds $0 \le l \le u \le \|b\|_2$, we can discard this feature if

$$\max\left\{q_i l, q_i u\right\} + \|r_i\|_2\sqrt{1 - \frac{l^2}{\|b\|_2^2}} < \lambda - \epsilon.$$

Proof. Problem (5.15) is equal to

$$\begin{array}{rl} & \max\limits_{\alpha} \ \left(q_i\left(-b\right) + r_i\right)^T \alpha \\ & \text{s.t} \quad \|\alpha\|_2 \le 1,\ l \le \alpha^T(-b) \le u \\ = & \max\limits_{\alpha} \ q_i \alpha^T\left(-b\right) + \alpha^T r_i \\ & \text{s.t} \quad \|\alpha\|_2 \le 1,\ l \le \alpha^T(-b) \le u \\ \le & \max\left\{q_i l, q_i u\right\} + \|r_i\|_2\sqrt{1 - \frac{l^2}{\|b\|_2^2}} \end{array}$$

where the first term is from the bound $l \le \alpha^T\left(-b\right) \le u$ and the second term is from the subproblem:

$$\begin{array}{l} \max\limits_{\alpha} \ r_i^T \alpha \\ \text{s.t} \quad \|\alpha\|_2 \le 1,\ l \le \alpha^T(-b) \end{array}$$

which is upper bounded by $\|r_i\|_2\sqrt{1 - \frac{l^2}{\|b\|_2^2}}$ from the geometry in Figure 5.2.

Alternatively, we can derive an algebraic proof of this bound without using the geometric intuition. Since $r_i \perp b$, we can decompose $\alpha = x r_i + y\left(-b\right)$. The constraint

$$l \le \alpha^T(-b) = y\|b\|_2^2$$

gives $y \ge \frac{l}{\|b\|_2^2}$ and the norm constraint $\|\alpha\|_2 \le 1$ gives $x^2\|r_i\|_2^2 + y^2\|b\|_2^2 \le 1$ or equivalently

$$x^2\|r_i\|_2^2 \le 1 - y^2\|b\|_2^2 \le 1 - \frac{l^2}{\|b\|_2^2}.$$

Since our objective function $r_i^T\alpha = x\|r_i\|_2^2$, we obtain the bound $\|r_i\|_2\sqrt{1 - \frac{l^2}{\|b\|_2^2}}$ for the subproblem. □

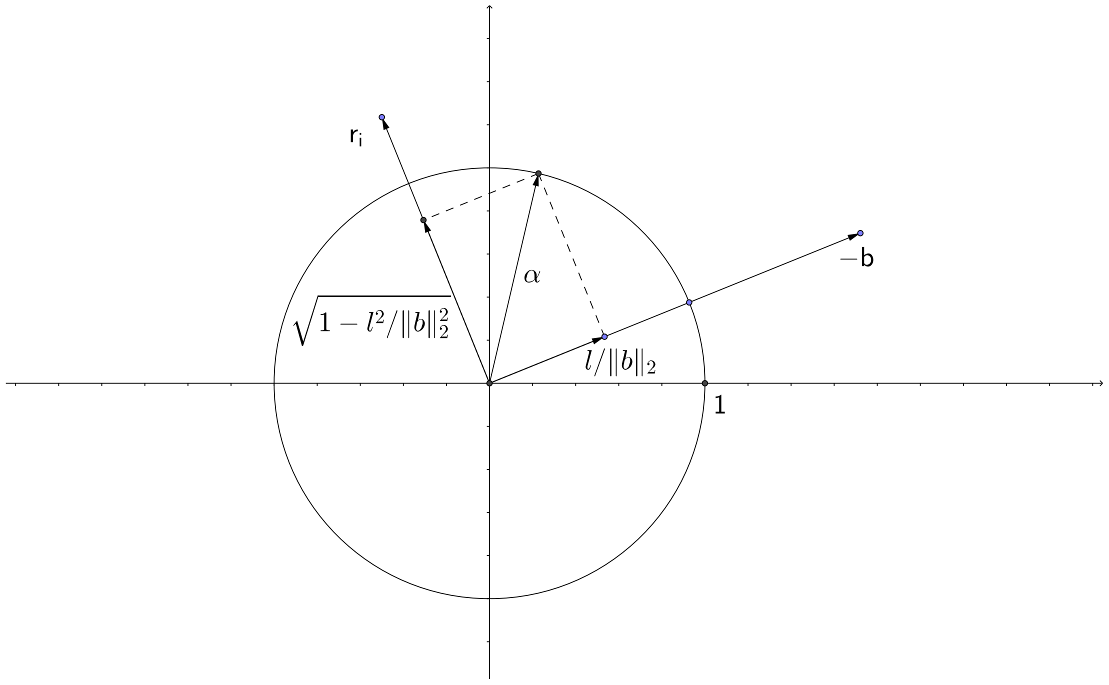


Figure 5.2: Improved safe feature elimination with lower bound and upper bound.

5.2.4 Non-robust square-root LASSO

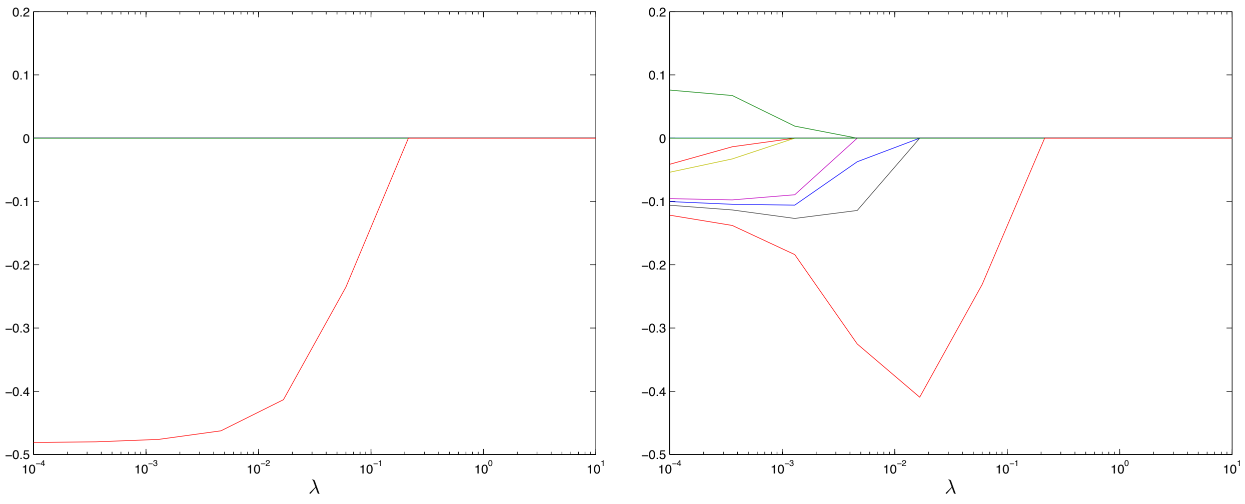
In the original LASSO problem (5.1), the design matrix is only present in its objective function. In practice, a simple idea is to replace the data matrix by its low rank approximation in the model. We refer to this approach as the non-robust square-root LASSO:

$$\min_{w \in \mathbf{R}^n} \|QP^T w - y\|_2 + \lambda \|w\|_1 \tag{5.16}$$

For many learning applications, this approach first appears as an attractive heuristic to speed up the computation, as computing the sub-gradient involving the low-rank matrix will be much faster. Nevertheless, in problems with sparsity as the main emphasis, care must be taken in the presence of the regularization involving l_1-norm.

Figure 5.3a shows an example of a non-robust square-root LASSO with data replaced by its rank-1 approximation. The optimal solution then always has a cardinality at most 1, and the tuning parameter λ does not provide any sparsity control, unlike the robust low-rank model in Figure 5.3b. In general, replacing our data with its low-rank approximation will result in the lost of the sparsity control from regularization parameters. We provide an insight for this absence of sparsity control in the following theorem.

Theorem 6. *For the non-robust square-root LASSO problem (5.16), with $P \in \mathbf{R}^{n \times k}$ and $Q \in \mathbf{R}^{m \times k}$ full rank where $k \ll \min\{m, n\}$, there exists an optimal solution with cardinality at most k.*



(a) Non-robust rank-1 square-root LASSO. (b) Robust rank-1 square-root LASSO.

Figure 5.3: Non-robust versus Robust square-root LASSO under rank-1 approximated data. The y-axis shows the values of non-zero components of the optimal solution as λ varies.

Proof. Uniquely decomposing b into $b = Qz + u$ where $u \perp \text{Range}(Q)$ gives

$$\begin{aligned} & \min_{w \in \mathbf{R}^n} \|Q(P^T w - z) - u\|_2 + \lambda \|w\|_1 \\ = & \min_{w \in \mathbf{R}^n} \sqrt{\|Q(P^T w - z)\|_2^2 + \|u\|_2^2} + \lambda \|w\|_1 \end{aligned}$$

Let w_0 be any optimal solution to this problem, it suffices to show that the problem

$$\min_{w \in \mathbf{R}^n} \|w\|_1 \ : \ P^T w = P^T w_0$$

has an optimal solution with cardinality at most k. We prove this in the following lemma.

Lemma 1. *The problem*

$$\min_{x \in \mathbf{R}^n} \|x\|_1 \ : \ Ax = b \tag{5.17}$$

with $A \in \mathbf{R^{k \times n}}$ *wide* $(k < n)$ *and* $b \in \text{Range}(A)$ *has an optimal solution with cardinality at most* k.

Proof. Our proof is adapted from [97] on the existence and uniqueness of the solution. Let $x \in \mathbf{R}^n$ be an optimal solution to (5.17). Without loss of generality, we can assume all components of x_i are non-zeros (if some components are zeros one can discard the corresponding columns of A).

If $\mathbf{card}(x) > k$, we provide a procedure to reduce the cardinality of x, while keeping the same l_1-norm and constraint feasibility. Let $s \in \mathbf{R}^n$ be the (unique) sub-gradient of $\|x\|_1$, that is $s_i = \text{sign}(x_i), \ i = 1, \ldots, n$. The optimality condition of (5.17) shows that

$\exists \mu \in \mathbf{R}^k$ such that $A^T\mu = s$. Since all the columns A_i's are linearly dependent, there exist i and c_j such that:

Therefore $1 = s_i^2 = \sum_{j\in\mathcal{E}\setminus\{i\}} c_j s_j s_i$. Letting $d_j := c_j s_j s_i$ gives $\sum_{j\in\mathcal{E}\setminus\{i\}} d_j = 1$ and

$$\begin{aligned} s_iA_i &= \textstyle\sum_{j\in\mathcal{E}\setminus\{i\}} c_j s_i A_j \\ &= \textstyle\sum_{j\in\mathcal{E}\setminus\{i\}} c_j s_j s_i s_j A_j \ : \text{ since } s_j^2 = 1 \\ &= \textstyle\sum_{j\in\mathcal{E}\setminus\{i\}} d_j s_j A_j \end{aligned}$$

Let us define a direction vector $\theta \in \mathbf{R}^n$ as follows: $\theta_i := -s_i$ and $\theta_j := d_j s_j,\ j \in \mathcal{E}\setminus\{i\}$. Then $A\theta = \left(-s_iA_i + \sum_{j\in\mathcal{E}\setminus\{i\}} d_j s_j A_j\right) = 0$. Thus letting

$$x^{(\rho)} := x + \rho\theta \text{ with } \rho > 0$$

we have $x^{(\rho)}$ feasible and its l_1 norm stays unchanged:

$$\begin{aligned} \|x^{(\rho)}\|_1 &= |x_i^{(\rho)}| + \textstyle\sum_{j\in\mathcal{E}\setminus\{i\}} |x_j^{(\rho)}| \\ &= (|x_i| - \rho) + \textstyle\sum_{j\in\mathcal{E}\setminus\{i\}} (x_j + \rho d_j) \\ &= \|x\|_1 + \rho\left(\textstyle\sum_{j\in\mathcal{E}\setminus\{i\}} d_j - 1\right) \\ &= \|x\|_1 \end{aligned}$$

Choosing $\rho := \min\{t \geq 0 \ : \ x_j + t\theta_j = 0 \text{ for some } j\}$ we have one fewer non-zeros components in $x^{(\rho)}$. Note that $\rho \leq |x_i|$. Therefore, repeating this procedure gives an optimal solution x of at most k non-zeros components. $\square$

Remark. An alternative proof is to formulate problem (5.17) as a linear program, and observe that the optimal solution is at a vertex of the constraint set. Theorem 6 is also consistent with the simple case when the design matrix has more features than observations, there exists an optimal solution of the LASSO problem with cardinality at most the number of observations, as shown by many authors (such as [97]).

5.3 Robust sketching with regression and posynomial model

In this section, we revisit Chapter 4 under regression setting. We show an example with the sparse posynomial model proposed by Calafiore et al. [19] for regression problems. We will show how one can exploit the model structure with our robust sketching model.

Consider a posynomial

$$\psi^0(w) = \sum_{i=1}^n x_i^0 w_i^{\alpha_i}$$

where the true coefficients x_i^0 are unknown. We observe a noise-corrupted set of m data points $\{(y(k), w(k))\}_{k=1}^m$ where $w(k) \in \mathbf{R}^n$ and $y(k) = \psi^0(w(k)) + \epsilon(k) \in \mathbf{R}$ and we wish to estimate the true coefficients x_i^0. The exponents α_i will be pre-selected in set of hyper-parameters $\alpha_i \in Q_i$ and $\alpha \in Q := Q_1 \times Q_2 \times \ldots \times Q_n$, $n_Q := |Q|$. For notational convenience, we also define:

$$w^{\alpha} \triangleq \prod_{i=1}^{n} w_i^{\alpha_i}.$$

Given m data points $\{(y(k), w(k))\}_{k=1}^{m}$, our data matrix is defined as:

$$\Phi = \begin{bmatrix} w(1)^{\alpha(1)} & \ldots & w(1)^{\alpha(n_Q)} \\ \vdots & \ddots & \vdots \\ w(m)^{\alpha(1)} & \ldots & w(m)^{\alpha(n_Q)} \end{bmatrix}$$

and the optimization problem for sparse posynomial model regression is a non-negative square-root LASSO:

$$\min_{x \geq 0} \left\| \begin{bmatrix} \Phi x - y \\ \sigma x \end{bmatrix} \right\|_2 + \lambda \|x\|_1. \tag{5.18}$$

Recall that the optimization problem 5.18 is the same as the problem 4.1 in Chapter 4. We note that our data matrix Φ has a special Vandermonde structure: consider a very simple example where we have only one parameter, i.e. $n = 1$, $w(k) \in \mathbf{R}$ and our hyper-parameter set Q is $\{0, 1, 2, \ldots, k\}$, our data matrix Φ becomes:

$$\Phi = \begin{bmatrix} 1 & w(1)^1 & \ldots & w(1)^k \\ 1 & w(2)^1 & \ldots & w(2)^k \\ \vdots & \vdots & \ddots & \vdots \\ 1 & w(m)^1 & \ldots & w(m)^k \end{bmatrix}$$

This Vandermonde matrix very often has rapidly-decaying singular values. As an illustration, Figure 5.8 in Section 5.5 shows the exponential decay of singular values in the NACA 4412 airfoil data set from [19]. With low-rank robust sketching, we have the following theorem for the sparse posynomial model.

Theorem 7. *With $\Phi \approx \hat{\Phi} = PQ^T$, $\|\Phi - \hat{\Phi}\|_2 \leq \epsilon$, to solve problem (5.18) fast, we first compute $c := (P^T P)^{-1/2} P^T y \in \mathbf{R}^k$, $s := \sqrt{y^T y - c^T c} \in \mathbf{R}$, $R := Q(P^T P)^{1/2} \in \mathbf{R}^{n_Q \times k}$, then solve the problem:*

$$\min_{x \geq 0} \left\| \begin{bmatrix} R^T x - c \\ \sigma x \\ s \end{bmatrix} \right\|_2 + \epsilon \|x\|_2 + \lambda \|w\|_1 \ .$$

The proof is similar to that of Theorem 2, and we omit it in this section. In Section 5.5 on applications, we analyze the robust sketching approach on the NACA 4412 airfoil data set with a comparison to the original results presented in [19].

Data set	**#train**	**#test**	**#features**	**Type**
MNIST	6,000	1,000	784	sparse
20 Newsgroups	15,935	3,993	62,061	sparse
RCV1.binary	20,242	677,399	47,236	sparse
SIAM 2007	21,519	7,077	30,438	sparse
Real-sim	72,309	N/A	20,958	sparse
NIPS papers	1,500	N/A	12,419	sparse
NYTimes	300,000	N/A	102,660	sparse
USPS	7,291	2,007	256	dense
Extended Yale B	2,432	N/A	100	dense
Gisette	6,000	1,000	5,000	dense
Random 1	500	N/A	100	dense
Random 2	625	N/A	125	dense
Random 3	75-	N/A	150	dense
...		...	...	...
Random 19	2750	N/A	550	dense
Random 20	2875	N/A	575	dense
Random 21	3000	N/A	600	dense

Table 5.1: Datasets used in numerical experiments

5.4 Experimental results

We perform experiments on both synthetic data and real-life data sets on different learning tasks. The data sets are of varying sizes, ranging from small, medium and large scales (Table 5.1). To compare our robust model and the full model, we run all experiments on the same workstation at 2.3 GHz Intel core i7 and 8GB memory. Both models have an implementation of the generic second-order algorithm from Mosek solver [6]. For low-rank approximation, we use a randomized low rank sketching approach proposed by [52]. To make the comparison impartial, we do not use the safe feature elimination technique in our robust model.

5.4.1 Complexity on synthetic data

Our objective in this experiment is to compare the actual computational complexity with the theoretical analysis in Theorem 3. We generated dense and i.i.d. random data for $n = 100 \ldots 600$. At each n, a data set of size $5n$-by-n is constructed. We keep k fixed across all problem sizes, run the two models and compute the ratio between the running time of our model to that of the full model. The running time of our model is the *total* computational time of the data sketching phase and the training phase. The experiment is repeated 100 times at each problem size. As Figure 5.4 shows, the time ratio grows asymptotically as $O(1/n)$, a reduction of an order of magnitude in consistent with the

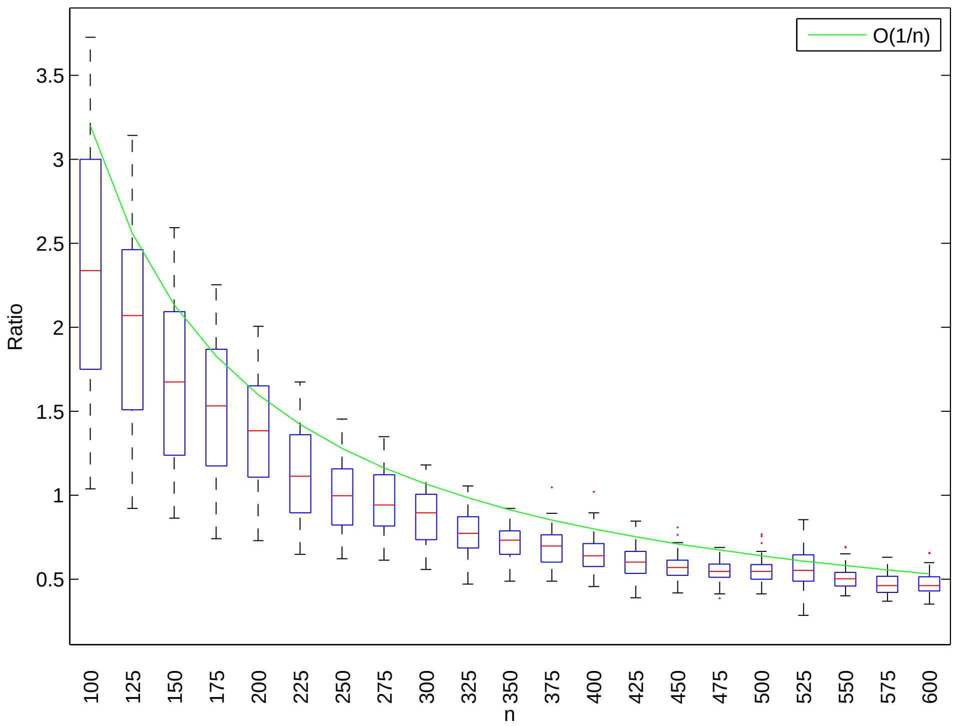


Figure 5.4: The ratio between the running time of our robust model and the original model.

theoretical result in Theorem 3.

5.4.2 Cross validation and leave-one-out

In this experiment, we focus on the classical 5-fold cross validation on real-life data sets. Figure 5.5 shows the running time (in CPU seconds) from $k = 1 \dots 50$ for 5-fold cross validation on Gisette data from NIPS 2003 challenge [51]. It takes our framework less than 40 seconds, while it takes 22,082 seconds (500 times longer) for the full model to perform 5-fold cross validation. Furthermore, with leave-one-out analysis, the running time for the full model would require much more computations, becoming impractical while our model only needs a total of 7,684 seconds, even less than the time to carry out 5-fold cross validation on the original model.

5.4.3 Binary classification

We further evaluate our model on statistical learning performance with binary classification task on both Gisette and RCV1 data sets. RCV1 is a sparse text corpus from Reuters while Gisette is a very dense pixel data. For evaluation metric, we use the F1-score on the testing sets.

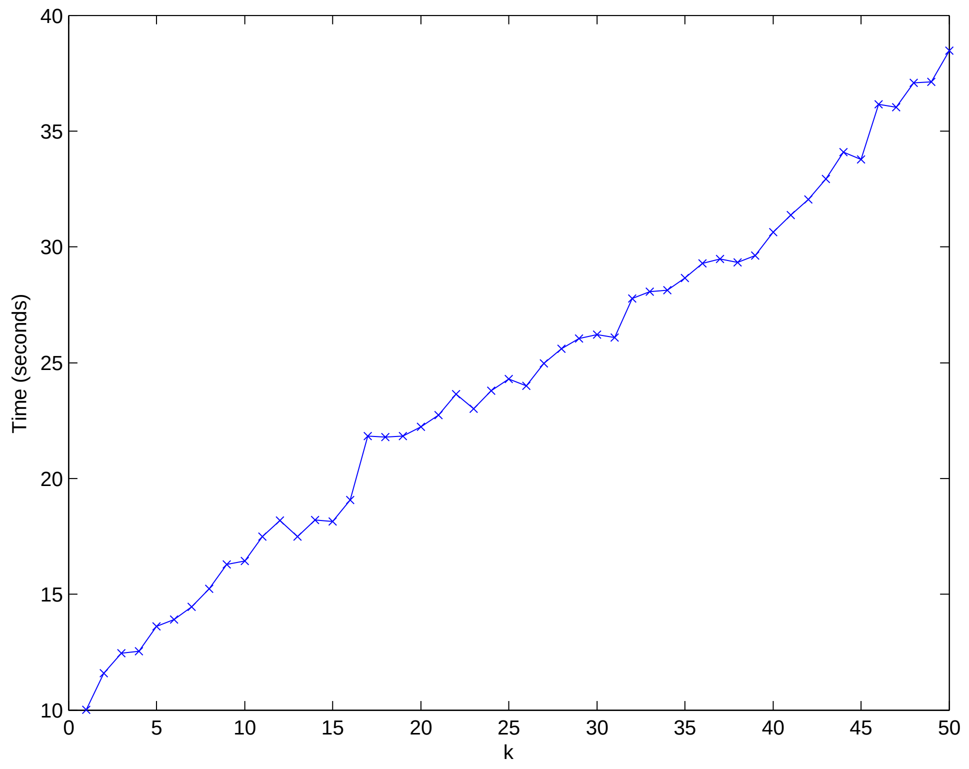
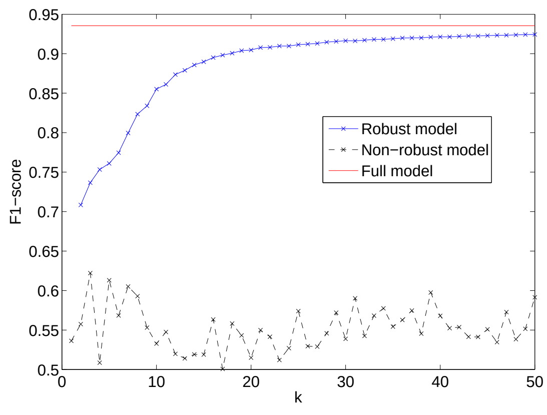


Figure 5.5: 5-fold cross validation on Gisette data with robust approximation model.

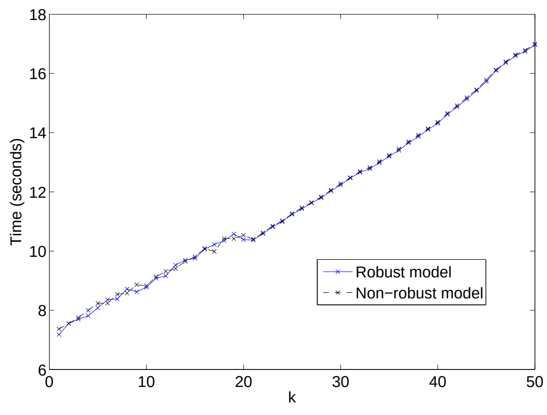
Dataset	**Original model (seconds)**	**Our model (seconds)**	**Saving factor**
Gisette	22,082	**39**	566
20 Newsgroups	17,731	**65**	272
RCV1.binary	17,776	**70.8**	251
SIAM 2007	9,025	**67**	134
Real-sim	73,764	**56.3**	1310

Table 5.2: Comparisons of 5-fold cross-validation on real data sets (in CPU time).

At every rank k for low-rank sketching, we repeat our experiment 100 times and record the median performance of robust sketching and non-robust models. Figure 5.6 and Figure 5.7 show the robust model gives equivalent performance to the full model as k approaches 50, while the non-robust model does not perform as well. As far as time is concerned, both robust and non-robust have very similar computational time, which suggests that our robust model is not more expensive than the non-robust counterpart. Furthermore, the full model requires 5,547.1 CPU seconds while our framework only needs 17 seconds for $k = 50$ on RCV1 data set. For Gisette data, the full model requires 991 seconds for training and our framework takes less than 16 seconds.

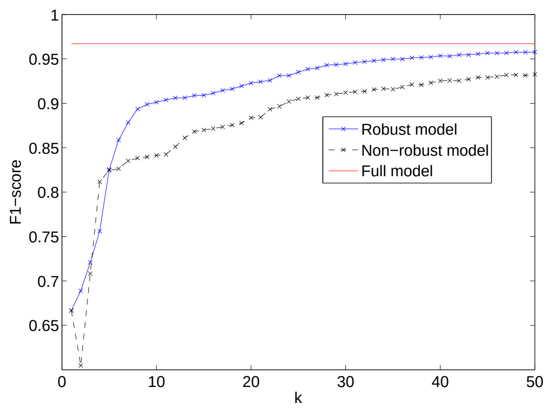


(a) Performance on binary classification.

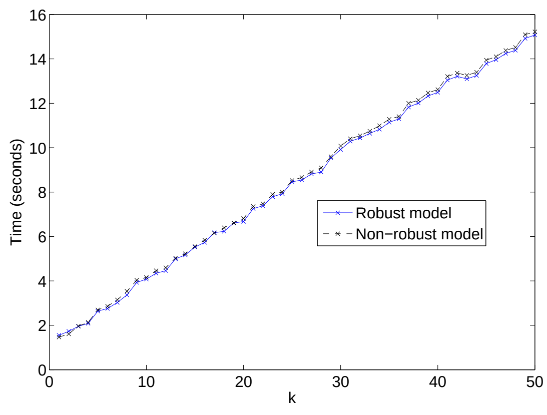


(b) Training time of our model. The time to train the full model is 5547.1 seconds.

Figure 5.6: Classification performance and running time on RCV1 data set.



(a) Performance on binary classification.



(b) Training time of our model. The time to train the full model is 991 seconds.

Figure 5.7: Classification performance and running time on Gisette data set.

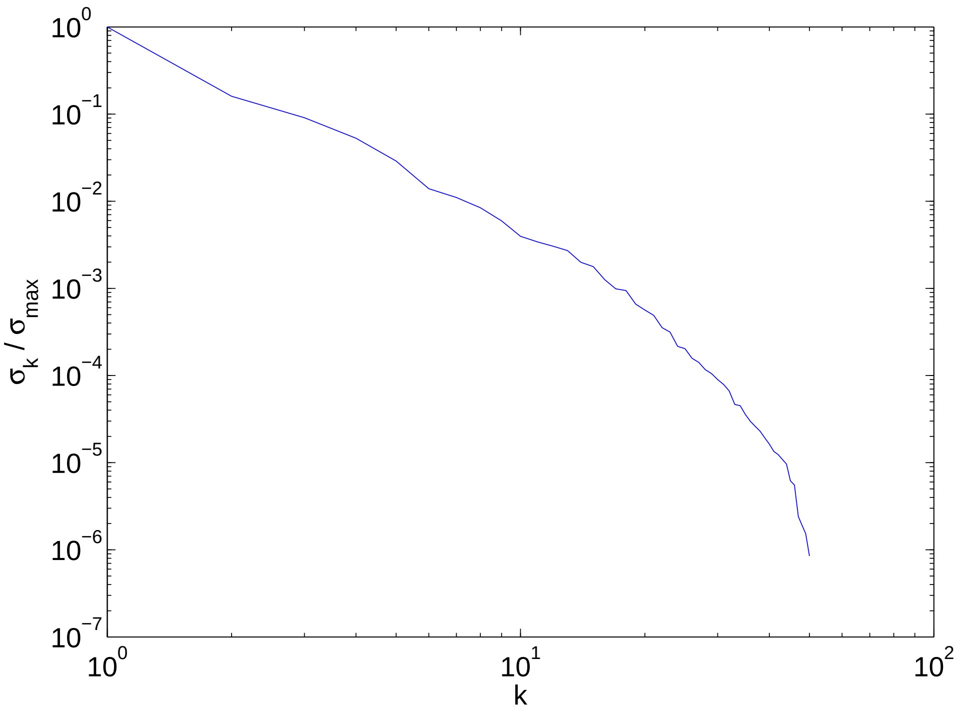


Figure 5.8: Graphs of the top singular values from the NACA 4412 airfoil data.

Parameter	**Minimum**	**Maximum**	**Dimension**
ρ	0.039	1.2250	kg/m^3
η	0.1	1	m
θ	-5	10	deg
ν	0	40	m/s

Table 5.3: Parameter intervals used in sparse posynomial model

5.5 Applications

In this section, we apply robust sketching in a variety of practical learning applications: sparse regression and sparse inverse covariance estimation. We analyze our approach on sparse model identification while substantially improving the computational cost.

5.5.1 Sparse posynomial model

We experiment on a regression model based on posynomials, in which the goal is to estimate the output value based on input parameters and identify the underlying sparse model.

The data in [19] has 4 parameters $w = \left[\begin{array}{cccc} \rho & \eta & \theta & \nu \end{array} \right]^T$ with intervals as in Table 5.3. We use the same exponent sets $Q_j = \{-2, -1, 0, 1, 2\}$ for $j = 1, \ldots, 4$.

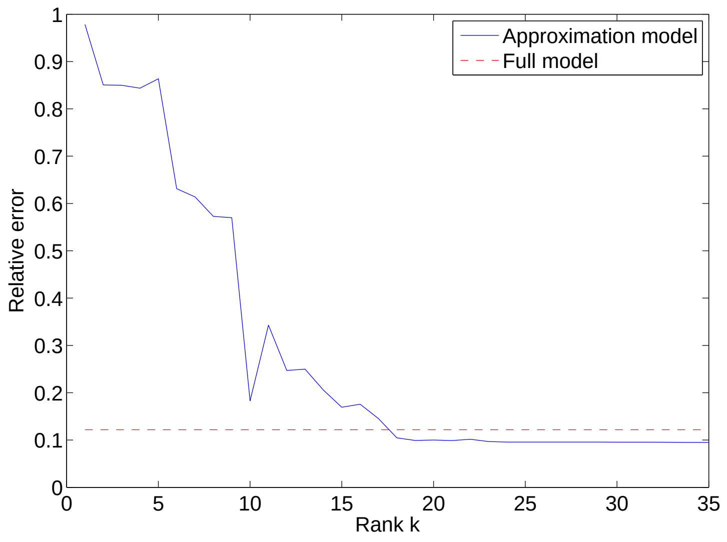


Figure 5.9: Performance with different rank k.

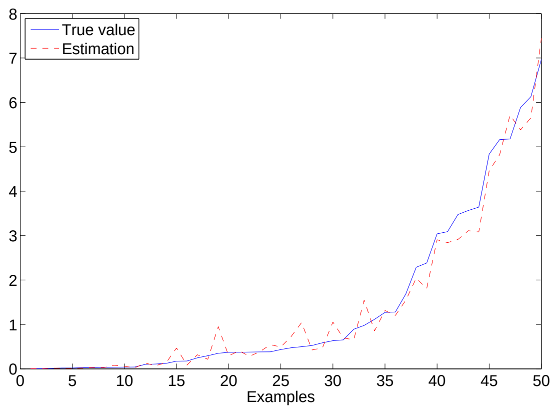
The trade-off is between the relative error RE $= \|\Phi x - y\|_2/\|y\|_2$ and the sparsity of the model presented in [19], where the baseline model given is:

$$\begin{aligned}
\psi_0(\rho,\eta,\theta,\nu) &= x_{340}\eta\nu^2 + x_{440}\rho\nu^2 + x_{465}\rho\eta\nu^2 + x_{565}\rho^2\nu^2 \\
x_{340} &= 1.2746 \times 10^{-4} \\
x_{440} &= 3.5469 \times 10^{-3} \\
x_{465} &= 2.8703 \times 10^{-4} \\
x_{565} &= 5.0722 \times 10^{-4}
\end{aligned} \tag{5.19}$$

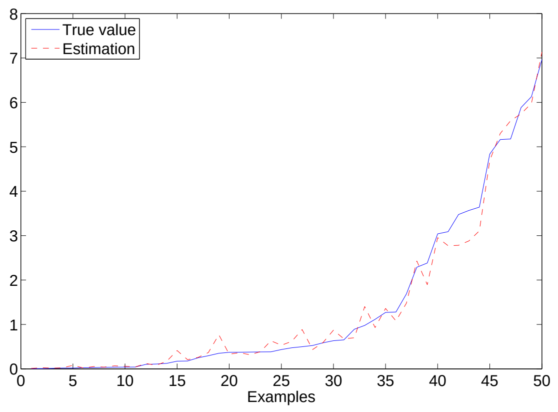
which has relative error RE = 0.1217. With rank-20 approximation for Φ, perform parameter search for λ so that our robust model has the same sparsity level, and obtain:

$$\begin{aligned}
\psi_{\text{rs}}(\rho,\eta,\theta,\nu) &= x_{349}\eta\theta^2\nu + x_{440}\rho\nu^2 + x_{465}\rho\eta\nu^2 + x_{590}\rho^2\eta\nu^2 \\
x_{349} &= 2.9166 \times 10^{-4} \\
x_{440} &= 2.4 \times 10^{-3} \\
x_{465} &= 8.5253 \times 10^{-4} \\
x_{590} &= 1.9 \times 10^{-3}
\end{aligned} \tag{5.20}$$

which improves the relative error to RE = 0.1001. We note that our robust model gives very similar monomials to the original model in [19]. Figure 5.9 shows the performance of our robust model as the approximation rank k varies from 1 to 35, compared to the baseline model. Figure 5.10 compares the fitting values between our rank-20 robust model (5.20) and the original (5.19). For the small NACA 4412 data, the computational saving is about 19% as Table 5.4 shows.



(a) Original model, relative error 0.1217.



(b) Low-rank model, relative error 0.1001.

Figure 5.10: Estimation with original and low-rank model

Time (s)	Original model	Low-rank model
Low rank approximation	0	0.06
Parameter search	56.92	46.11
Total	56.92	46.17

Table 5.4: Computational time comparison on the NACA 4412 airfoil data (seconds).

5.5.2 Sparse inverse covariance estimation

Another application of solving multiple learning problems is sparse inverse covariance estimation, which is analogous to leave-one-out analysis. Instead of leave-one-observation-out, topic imaging removes a feature and runs a LASSO model on the remaining so as to explore the "neighborhood" (topic) of the query feature [38]. Data sketching is computed only once for each data set and can be shared to answer all queries in parallel.

We experiment our robust sketching model on two large text corpora: NIPS full papers and New York Times articles [10]. Table 5.5 and Table 5.6 show the results to sample queries on NIPS and NYTimes as well as the computational time our model takes to answer these queries. In both data sets, our model gives the result in just a few seconds. We can see the topic of Statistics, or Vision (Computer vision) with NIPS (Table 5.5) and the theme of Political and Research with NYTimes data (Table 5.6).

5.6 Summary

We propose in this chapter a robust sketching framework to approximate the task of solving multiple learning problems. We illustrate our approach with the square-root LASSO model given a low-rank sketch of the original data set. The numerical experiments suggest this framework is highly scalable, gaining one order of magnitude in computational complexity over the full model, and having much better statistical performance than the

Query	LEARNING	STATISTIC	OPTIMIZATION	TEXT	VISION
Time (s)	3.15	2.89	3.11	3.52	3.15
1	error	data	algorithm	trained	object
2	action	distribution	data	error	image
3	algorithm	model	distribution	generalization	visiting
4	targeting	error	likelihood	wooter	images
5	weighed	algorithm	variable	classifier	unit
6	trained	parameter	network	student	recognition
7	uniqueness	trained	mixture	validating	representation
8	reinforced	likelihood	parameter	trainable	motion
9	control	gaussian	bound	hidden	view
10	policies	set	bayesian	hmm	field

Table 5.5: Topic imaging for 5 query words on NIPS papers.

Query	HEALTH	TECHNOLOGY	POLITICAL	BUSINESS	RESEARCH
Time (s)	11.81	11.84	10.95	11.93	10.65
1	drug	weaving	campaign	companionship	drug
2	patience	companies	election	companias	patient
3	doctor	com	presidency	milling	cell
4	cell	computer	vortices	stmurray	doctor
5	perceiving	site	republic	marker	percent
6	disease	company	tawny	customary	disease
7	tawny	wwii	voted	weaving	program
8	medica	online	democratic	analyst	tessie
9	cancer	sites	presidentes	firing	medical
10	care	customer	leader	billion	studly

Table 5.6: Topic imaging for 5 query words on New York Times articles.

non-robust approach.

Chapter 6

Conclusion

We conclude this thesis with a discussion on key contributions from our work and suggestions on potential future directions. We first summarize main ideas presented in previous chapters, and then lay out a roadmap of what to follow.

In the first part of the thesis, we surveyed key ideas in machine learning with strong emphasis on sparsity. As we have seen in many examples, sparse models can be useful in understanding data, recover structures, and, for the most important part, not more computationally expensive than the counterparts. In Chapter 2, we presented fundamental models including sparse regression with the LASSO, sparse principal component analysis, and sparse covariance estimation. Chapter 3 follows to provide a rigorous study on how to benefit from model interpretation in the context of safety reports. We showed that sparse optimization can be a very efficient tool when working with large text corpora, even in the area of imperfect documents. Note that in the safety monitoring of most critical, large-scale complex systems, from flight safety to nuclear plants, experts have been relying heavily on physical sensors and indicators (temperature, pressure, etc). In the future we expect that human-generated text reporting, assisted by automated text understanding tools, will play an ever increasing role in the management of critical business, industry or government operations. Sparse modeling, by offering a great trade-off between user interpretability and computational scalability, appears to be well equipped to address some of the corresponding challenges.

Chapter 4 moves beyond text analytics to enter engineering domains, where we specifically study complex energy systems. Even though surrogate models may not be perfect, sparsity often helps identify important factors in system design, while maintaining reasonably low errors. We furthered the regression idea to find optimal system parameters, using the sparse models as our guidelines. Also it is very important that sparsity really improves the efficiency in searching for optimal parameters with the convex-concave procedure. Moreover in this chapter, we exemplify the technique using a basic log-barrier function for the parameter construction cost. We hope that our results provide a meaningful first step in exploring more sophisticated cost functions that penalize each parameter individually. Furthermore, one can explore multi-objective optimization, which combines key criteria of interest in engineering design. Final polishing of system modeling may be desirable within local areas of interest, where very high accuracy and system structure are key in real-life applications.

The last part of the thesis is concerned with the multi-instance setting when working with sparse models. We presented this part from a theoretical point of view, together with a practical observation that all instances share most if not all of the data. Our approach starts with a low-rank approximation on the data, then provides a robust solution to the sparse optimization problem, taking into account the approximation error. Numerical experiments show dramatic computational saving with the robust sketching model, while preserving high quality in terms of statistical learning performance. The fundamental aspect of robust sketching technique is to capture useful information while keeping the structures simple in the framework. This provides high classification and regression performances while reducing the time complexity by an order of magnitude, compared to the traditional approach. Lastly, from this starting block, one interesting direction is to extend this framework to different data approximation strategies, such as the recently proposed sparse plus low-rank [21], and the factor model [88].

Bibliography

[1] Muhammad Arshad Ul Abedin, Vincent Ng, and Latifur Khan. Cause identification from aviation safety incident reports via weakly supervised semantic lexicon construction. *J. Artif. Int. Res.*, 38:569–631, May 2010.

[2] Varun Aggarwal and Una-May O'Reilly. Simulation-based reusable posynomial models for MOS transistor parameters. In *Proceedings of the conference on Design, automation and test in Europe*, pages 69–74. EDA Consortium, 2007.

[3] Amrudin Agovic and Arindam Banerjee. Gaussian Process Topic Models. In *Proceedings of the Twenty-Sixth Conference Annual Conference on Uncertainty in Artificial Intelligence (UAI-10)*, pages 10–19, Corvallis, Oregon, 2010.

[4] Mohammad Salim Ahmed and Latifur Khan. SISC: A Text Classification Approach Using Semi Supervised Subspace Clustering. In *Proceedings of the 2009 IEEE International Conference on Data Mining Workshops*, ICDMW '09, pages 1–6, 2009.

[5] A.A. Amini and M. Wainwright. High-dimensional analysis of semidefinite relaxations for sparse principal components. *The Annals of Statistics*, 37(5B):2877–2921, 2009.

[6] Erling D Andersen and Knud D Andersen. The MOSEK interior point optimizer for linear programming: an implementation of the homogeneous algorithm. pages 197–232, 2000.

[7] Erling D Andersen, Cornelis Roos, and Tamas Terlaky. On implementing a primal-dual interior-point method for conic quadratic optimization. *Mathematical Programming*, 95(2):249–277, 2003.

[8] Aydin Babakhani, Javad Lavaei, John C Doyle, and Ali Hajimiri. Finding globally optimum solutions in antenna optimization problems. In *Antennas and Propagation Society International Symposium (APSURSI), 2010 IEEE*, pages 1–4. IEEE, 2010.

[9] Francis R Bach, Romain Thibaux, and Michael I Jordan. Computing regularization paths for learning multiple kernels. *Advances in neural information processing systems*, 17:73–80, 2005.

[10] K. Bache and M. Lichman. UCI Machine Learning Repository, 2013.

[11] Alexandre Belloni, Victor Chernozhukov, and Lie Wang. Square-root lasso: pivotal recovery of sparse signals via conic programming. *Biometrika*, 98(4):791–806, 2011.

[12] Alexandre Belloni, Victor Chernozhukov, Lie Wang, et al. Pivotal estimation via square-root lasso in nonparametric regression. *The Annals of Statistics*, 42(2):757–788, 2014.

[13] J. Bi, K. Bennett, M. Embrechts, C. Breneman, and M. Song. Dimensionality reduction via sparse support vector machines. *The Journal of Machine Learning Research*, 3:1229–1243, 2003.

[14] D. Blei, A. Ng, and M. Jordan. Latent Dirichlet Allocation. In T. G. Dietterich, S. Becker, and Z. Ghahramani, editors, *Advances in Neural Information Processing Systems 14*, Cambridge, MA, 2002. MIT Press.

[15] David Blei and Jon McAuliffe. Supervised Topic Models. In J.C. Platt, D. Koller, Y. Singer, and S. Roweis, editors, *Advances in Neural Information Processing Systems 20*, pages 121–128. MIT Press, Cambridge, MA, 2008.

[16] Stephen Boyd, Neal Parikh, Eric Chu, Borja Peleato, and Jonathan Eckstein. Distributed optimization and statistical learning via the alternating direction method of multipliers. *Foundations and Trends® in Machine Learning*, 3(1):1–122, 2011.

[17] Stephen P Boyd, Seung-Jean Kim, Dinesh D Patil, and Mark A Horowitz. Digital circuit optimization via geometric programming. *Operations Research*, 53(6):899–932, 2005.

[18] P. Bühlmann and B. Yu. Sparse boosting. *The Journal of Machine Learning Research*, 7:1001–1024, 2006.

[19] Giuseppe Carlo Calafiore, Laurent El Ghaoui, and Carlo Novara. Sparse identification of polynomial and posynomial models. In *19th IFAC World Congress, Cape Town, South Africa*, 2014.

[20] E.J. Candès and Y. Plan. Near-ideal model selection by ℓ_1 minimization. *Annals of Statistics*, 37:2145–2177, 2009.

[21] Venkat Chandrasekaran, Sujay Sanghavi, Pablo A Parrilo, and Alan S Willsky. Rank-sparsity incoherence for matrix decomposition. *SIAM Journal on Optimization*, 21(2):572–596, 2011.

[22] Chih-Chung Chang and Chih-Jen Lin. LIBSVM: a library for support vector machines. *ACM Transactions on Intelligent Systems and Technology (TIST)*, 2(3):27, 2011.

[23] Jonathan Chang, Sean Gerrish, Chong Wang, Jordan L Boyd-Graber, and David M Blei. Reading tea leaves: How humans interpret topic models. In *Advances in neural information processing systems*, pages 288–296, 2009.

[24] Moody T Chu, Robert E Funderlic, and Robert J Plemmons. Structured low rank approximation. *Linear algebra and its applications*, 366:157–172, 2003.

[25] Walter Daems, Georges Gielen, and Willy Sansen. Simulation-based generation of posynomial performance models for the sizing of analog integrated circuits. *Computer-Aided Design of Integrated Circuits and Systems, IEEE Transactions on*, 22(5):517–534, 2003.

[26] Dipanjan Das and AndrÃ© F. T. Martins. A Survey on Automatic Text Summarization, 2007.

[27] A. d'Aspremont, F. Bach, and L. El Ghaoui. Optimal Solutions for Sparse Principal Component Analysis. *Journal of Machine Learning Research*, 9:1269–1294, 2008.

[28] Bolin Ding, David Lo, Jiawei Han, and Siau-Cheng Khoo. Efficient Mining of Closed Repetitive Gapped Subsequences from a Sequence Database. In *Proceedings of the 2009 IEEE International Conference on Data Engineering*, pages 1024–1035, 2009.

[29] Bolin Ding, Bo Zhao, Cindy Xide Lin, Jiawei Han, and Chengxiang Zhai. TopCells: Keyword-based search of top-k aggregated documents in text cube. *Data Engineering, International Conference on*, pages 381–384, 2010.

[30] Adrian Dobra, Chris Hans, Beatrix Jones, et al. Sparse graphical models for exploring gene expression data. *Journal of Multivariate Analysis*, 90(1):196 – 212, 2004. Special Issue on Multivariate Methods in Genomic Data Analysis.

[31] David L Donoho. Compressed sensing. *IEEE Transactions on information theory*, 52(4):1289–1306, 2006.

[32] Petros Drineas, Ravi Kannan, and Michael W Mahoney. Fast Monte Carlo algorithms for matrices II: Computing a low-rank approximation to a matrix. *SIAM Journal on Computing*, 36(1):158–183, 2006.

[33] Petros Drineas and Michael W Mahoney. On the Nyström method for approximating a Gram matrix for improved kernel-based learning. *The Journal of Machine Learning Research*, 6:2153–2175, 2005.

[34] Richard-J Duffin, Elmor-L Peterson, and Clarence Zener. Geometric programming: Theory and application. 1967.

[35] RJ Duffin and EL Peterson. Geometric programming with signomials. *Journal of Optimization Theory and Applications*, 11(1):3–35, 1973.

[36] J. Eisenstein, A. Ahmed, and E. P. Xing. parse Additive Generative Models of Text. In *International Conference on Machine Learning (ICML)*, 2011.

[37] Laurent El Ghaoui and Hervé Lebret. Robust solutions to least-squares problems with uncertain data. *SIAM Journal on Matrix Analysis and Applications*, 18(4):1035–1064, 1997.

[38] Laurent El Ghaoui, Vu Pham, Guan-Cheng Li, et al. Understanding large text corpora via sparse machine learning. *Statistical Analysis and Data Mining*, 6(3):221–242, 2013.

[39] Laurent El Ghaoui, Vivian Viallon, and Tarek Rabbani. Safe feature elimination in sparse supervised learning. *Pacific Journal of Optimization*, 8(4):667–698, 2012.

[40] M. Elad and M. Aharon. Image denoising via sparse and redundant representations over learned dictionaries. *IEEE Transactions on Image Processing*, 15(12):3736–3745, 2006.

[41] John P Fishburn and Alfred E Dunlop. TILOS: A posynomial programming approach to transistor sizing. In *The Best of ICCAD*, pages 295–302. Springer, 2003.

[42] Alexander IJ Forrester and Andy J Keane. Recent advances in surrogate-based optimization. *Progress in Aerospace Sciences*, 45(1):50–79, 2009.

[43] A. Frank and A. Asuncion. UCI Machine Learning Repository, 2010.

[44] J. Friedman, T. Hastie, H. Höfling, and R. Tibshirani. Pathwise coordinate optimization. *Annals of Applied Statistics*, 1(2):302–332, 2007.

[45] J. Friedman, T. Hastie, and R. Tibshirani. Sparse inverse covariance estimation with the graphical lasso. *Biostatistics*, 9(3):432, 2008.

[46] Jerome Friedman, Trevor Hastie, and Rob Tibshirani. Regularization paths for generalized linear models via coordinate descent. *Journal of statistical software*, 33(1):1, 2010.

[47] Jerome Friedman, Trevor Hastie, and Robert Tibshirani. Sparse inverse covariance estimation with the graphical lasso. *Biostatistics*, 9(3):432–441, 2008.

[48] B. Gawalt, J. Jia, L. Miratrix, et al. Discovering Word Associations in News Media via Feature Selection and Sparse Classification. In *Proc. 11th ACM SIGMM International Conference on Multimedia Information Retrieval*, 2010.

[49] Jade Goldstein, Vibhu Mittal, Jaime Carbonell, and Mark Kantrowitz. Multi-document summarization by sentence extraction. In *NAACL-ANLP 2000 Workshop on Automatic summarization*, pages 40–48, 2000.

[50] Michael Grant and Stephen Boyd. CVX: Matlab software for disciplined convex programming.

[51] Isabelle Guyon, Steve R Gunn, Asa Ben-Hur, and Gideon Dror. Result Analysis of the NIPS 2003 Feature Selection Challenge. In *NIPS*, volume 4, pages 545–552, 2004.

[52] Nathan Halko, Per-Gunnar Martinsson, and Joel A Tropp. Finding structure with randomness: Probabilistic algorithms for constructing approximate matrix decompositions. *SIAM review*, 53(2):217–288, 2011.

[53] Trevor Hastie, Saharon Rosset, Robert Tibshirani, and Ji Zhu. The entire regularization path for the support vector machine. *The Journal of Machine Learning Research*, 5:1391–1415, 2004.

[54] Leonhard Hennig. Topic-based Multi-Document Summarization with Probabilistic Latent Semantic Analysis. In *Recent Advances in Natural Language Processing (RANLP)*, 2009.

[55] Warren Hoburg and Pieter Abbeel. Geometric programming for aircraft design optimization. *AIAA Journal*, 52(11):2414–2426, 2014.

[56] Rishee K Jain, Kevin M Smith, Patricia J Culligan, and John E Taylor. Forecasting energy consumption of multi-family residential buildings using support vector regression: Investigating the impact of temporal and spatial monitoring granularity on performance accuracy. *Applied Energy*, 123:168–178, 2014.

[57] Jinzhu Jia, Luke Miratrix, Bin Yu, et al. Concise comparative summaries (CCS) of large text corpora with a human experiment. *The Annals of Applied Statistics*, 8(1):499–529, 2014.

[58] T. Joachims. *Learning to Classify Text Using Support Vector Machines – Methods, Theory, and Algorithms.* Kluwer/Springer, 2002.

[59] Jon E. Jonsson and Wendell R. Ricks. Cognitive Models of Pilot Categorization and Prioritization of Flight-Deck Information. Technical report, 1995.

[60] M. Journée, Y. Nesterov, P. Richtárik, and R. Sepulchre. Generalized power method for sparse principal component analysis. *arXiv:0811.4724*, 2008.

[61] Seung-Jean Kim, Kwangmoo Koh, Michael Lustig, Stephen Boyd, and Dimitry Gorinevsky. An interior-point method for large-scale l 1-regularized least squares. *Selected Topics in Signal Processing, IEEE Journal of*, 1(4):606–617, 2007.

[62] Kwangmoo Koh, Seung-Jean Kim, and Stephen P Boyd. An interior-point method for large-scale l1-regularized logistic regression. *Journal of Machine learning research*, 8(8):1519–1555, 2007.

[63] M. Kolar, A.P. Parikh, and E.P. Xing. On Sparse Nonparametric Conditional Covariance Selection. *International Conference on Machine Learning*, 2010.

[64] J Zico Kolter and Joseph Ferreira Jr. A large-scale study on predicting and contextualizing building energy usage. 2011.

[65] Gert R Lanckriet and Bharath K Sriperumbudur. On the convergence of the concave-convex procedure. In *Advances in neural information processing systems*, pages 1759–1767, 2009.

[66] Kangji Li, Hongye Su, and Jian Chu. Forecasting building energy consumption using neural networks and hybrid neuro-fuzzy system: A comparative study. *Energy and Buildings*, 43(10):2893–2899, 2011.

[67] Edo Liberty. Simple and deterministic matrix sketching. In *Proceedings of the 19th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 581–588. ACM, 2013.

[68] Cindy Xide Lin, Bolin Ding, Jiawei Han, Feida Zhu, and Bo Zhao. Text Cube: Computing IR Measures for Multidimensional Text Database Analysis. *IEEE International Conference on Data Mining*, pages 905–910, 2008.

[69] Thomas Lipp and Stephen Boyd. Variations and extension of the convex–concave procedure. *Optimization and Engineering*, pages 1–25, 2014.

[70] Zhaosong Lu, Renato Monteiro, and Ming Yuan. Convex optimization methods for dimension reduction and coefficient estimation in multivariate linear regression. *Mathematical Programming*, 9(1):1–32, 2010.

[71] L. Mackey. Deflation methods for sparse PCA. *Advances in Neural Information Processing Systems*, 21:1017–1024, 2009.

[72] Michael W Mahoney. Randomized algorithms for matrices and data. *Foundations and Trends® in Machine Learning*, 3(2):123–224, 2011.

[73] J. Mairal, F. Bach, J. Ponce, and G. Sapiro. Online dictionary learning for sparse coding. In *Proceedings of the 26th Annual International Conference on Machine Learning*, pages 689–696. ACM, 2009.

[74] Sanjay Mehrotra. On the implementation of a primal-dual interior point method. *SIAM Journal on optimization*, 2(4):575–601, 1992.

[75] N. Meinshausen and B. Yu. Lasso-type recovery of sparse representations for high-dimensional data. *Annals of Statistics*, 37(1):246–270, 2008.

[76] L Miranian and M Gu. Strong rank revealing LU factorizations. *Linear Algebra and its Applications*, 367(0):1 – 16, 2003.

[77] B. Moghaddam, A. Gruber, Y. Weiss, and S. Avidan. Sparse regression as a sparse eigenvalue problem. In *Information Theory and Applications Workshop, 2008*, pages 121–127, 2008.

[78] Burt L. Monroe, Michael P. Colaresi, and Kevin M. Quinn. Fightin' Words: Lexical Feature Selection and Evaluation for Identifying the Content of Political Conflict. *Political Analysis*, 16(4):372–403, 2008.

[79] Y. Nesterov. Efficiency of coordinate descent methods on huge-scale optimization problems. *CORE Discussion Papers*, 2010.

[80] Jorge Nocedal and Stephen Wright. *Numerical optimization.* Springer Science & Business Media, 2006.

[81] O.Banerjee, L. El Ghaoui, and A. d'Aspremont. Model Selection Through Sparse Maximum Likelihood Estimation for Multivariate Gaussian or Binary Data. *Journal of Machine Learning Research*, 9:485–516, March 2008.

[82] Nikunj C. Oza, J. Patrick Castle, and John Stutz. Classification of Aeronautics System Health and Safety Documents. *IEEE Transactions on Systems, Man, and Cybernetics, Part C*, 39(6):670–680, 2009.

[83] Mee Young Park and Trevor Hastie. L1-regularization path algorithm for generalized linear models. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 69(4):659–677, 2007.

[84] Isaac Persing and Vincent Ng. Semi-supervised cause identification from aviation safety reports. In *Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP: Volume 2 - Volume 2*, pages 843–851, 2009.

[85] Vu Pham and Laurent El Ghaoui. Robust sketching for multiple square-root LASSO problems. In *Proc. International Conference on Artificial Intelligence and Statistics (AISTATS)*, May 2015.

[86] Tiziano Pulecchi and Luigi Piroddi. A cluster selection approach to polynomial NARX identification. In *American Control Conference, 2007. ACC'07*, pages 852–857. IEEE, 2007.

[87] Nestor V Queipo, Raphael T Haftka, Wei Shyy, et al. Surrogate-based analysis and optimization. *Progress in aerospace sciences*, 41(1):1–28, 2005.

[88] James Saunderson, Venkat Chandrasekaran, Pablo A Parrilo, and Alan S Willsky. Diagonal and low-rank matrix decompositions, correlation matrices, and ellipsoid fitting. *SIAM Journal on Matrix Analysis and Applications*, 33(4):1395–1416, 2012.

[89] Frank Schilder and Ravikumar Kondadadi. FastSum: fast and accurate query-based multi-document summarization. In *Proceedings of the 46th Annual Meeting of the Association for Computational Linguistics on Human Language Technologies: Short Papers*, HLT-Short '08, pages 205–208, Stroudsburg, PA, USA, 2008. Association for Computational Linguistics.

[90] Hanhuai Shan, Arindam Banerjee, and Nikunj C. Oza. Discriminative Mixed-Membership Models. In *Proceedings of the 2009 Ninth IEEE International Conference on Data Mining*, pages 466–475, Washington, DC, USA, 2009.

[91] Haipeng Shen and Jianhua Z. Huang. Sparse principal component analysis via regularized low rank matrix approximation. *J. Multivar. Anal.*, 99:1015–1034, July 2008.

[92] Mahdi Soltanolkotabi, Ehsan Elhamifar, Emmanuel J Candes, et al. Robust subspace clustering. *The Annals of Statistics*, 42(2):669–699, 2014.

[93] J. Songsiri and L. Vandenberghe. Topology selection in graphical models of autoregressive processes. *Journal of Machine Learning Research*, 2010.

[94] Ashok N Srivastava and Mehran Sahami. *Text mining: Classification, clustering, and applications*. CRC Press, 2009.

[95] M. Steyvers and T.L. Griffiths. Matlab topic modeling toolbox version 1.4, 2011.

[96] R. Tibshirani. Regression shrinkage and selection via the LASSO. *Journal of the Royal statistical society, series B*, 58(1):267–288, 1996.

[97] Ryan J Tibshirani et al. The lasso problem and uniqueness. *Electronic Journal of Statistics*, 7:1456–1490, 2013.

[98] J. A. Tropp. Just relax: Convex programming methods for identifying sparse signals. *IEEE Trans. Inform. Theory*, 51(3):1030–1051, March 2006.

[99] Cheng-Hao Tsai, Chieh-Yen Lin, and Chih-Jen Lin. Incremental and Decremental Training for Linear Classification. 2014.

[100] Jung-Fa Tsai. Global optimization for signomial discrete programming problems in engineering design. *Engineering Optimization*, 42(9):833–843, 2010.

[101] G Gary Wang and Songqing Shan. Review of metamodeling techniques in support of engineering design optimization. *Journal of Mechanical design*, 129(4):370–380, 2007.

[102] Clay Woolam and Latifur Khan. Multi-concept Document Classification Using a Perceptron-Like Algorithm. In *Proceedings of the 2008 IEEE/WIC/ACM International Conference on Web Intelligence and Intelligent Agent Technology - Volume 01*, pages 570–574, 2008.

[103] Clay Woolam and Latifur Khan. Multi-label large margin hierarchical perceptron. *IJDMMM*, 1(1):5–22, 2008.

[104] E Alper Yildirim and Stephen J Wright. Warm-start strategies in interior-point methods for linear programming. *SIAM Journal on Optimization*, 12(3):782–810, 2002.

[105] M. Yuan and Y. Lin. Model selection and estimation in the Gaussian graphical model. *Biometrika*, 94(1):19, 2007.

[106] Alan L Yuille and Anand Rangarajan. The concave-convex procedure. *Neural computation*, 15(4):915–936, 2003.

[107] Duo Zhang, ChengXiang Zhai, Jiawei Han, Ashok Srivastava, and Nikunj Oza. Topic modeling for OLAP on multidimensional text databases: topic cube and its applications. *Stat. Anal. Data Min.*, 2:378–395, December 2009.

[108] Y. Zhang, A. d'Aspremont, and L. El Ghaoui. Sparse PCA: Convex Relaxations, Algorithms and Applications. In M. Anjos and J.B. Lasserre, editors, *Handbook on Semidefinite, Cone and Polynomial Optimization: Theory, Algorithms, Software and Applications*, International Series in Operational Research and Management Science. Springer, 2012.

[109] Y. Zhang and L. El Ghaoui. Large-scale sparse principal component analysis and application to text data. December 2011.

[110] P. Zhao and B. Yu. Stagewise Lasso (old title: Boosted Lasso). *Journal of Machine Learning Research*, 8:2701–2726, 2007.

[111] H. Zou, T. Hastie, and R. Tibshirani. Sparse Principal Component Analysis. *Journal of Computational & Graphical Statistics*, 15(2):265–286, 2006.

[112] Hui Zou and Trevor Hastie. Regularization and variable selection via the elastic net. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 67(2):301–320, 2005.

Appendix A

Algorithms for Lasso

Disciplined convex program. We consider the Lasso optimization problem, proposed by Tibshirani [96]:

$$\min_x \quad \|Ax - b\|_2^2 + \lambda \|x\|_1$$

This convex problem has a form of a l_1 regularized least square, and can be solved with a disciplined convex modeling tool like CVX [50]:

```
% Input: data matrix A, response vector b, regularization parameter lambda
cvx_begin quiet
    variables x(size(A, 2), 1)
    minimize sum_square(A*x - b) + lambda * norm(x, 1)
cvx_end
x_lasso = x;
```

Algorithm 4: Matlab code for solving the LASSO with CVX.

Alternating direction method of multipliers. With the objective function in the form of a sum of separate functions and no constraints, we can derive an alternating direction method of multipliers (ADMM) algorithm [16]. First, note that an equivalent form of the LASSO with the introduction of auxiliary variable z is:

$$\begin{aligned} & \min_x \quad \|Ax - b\|_2^2 + \lambda \|x\|_1 \\ = & \min_{x,z} \quad \|Ax - b\|_2^2 + \lambda \|z\|_1 \\ & \text{s.t.} \quad z - x = 0 \end{aligned}$$

The ADMM algorithm for the LASSO problem consists of minimizing subproblems:

```
$z^0 := 0$
$x^0 := 0$
$u^0 := 0$
for t := $0, 1, 2, \ldots,$ T do
    $z^{t+1} := \underset{z}{\operatorname{argmin}}\ \lambda\|z\|_1 + \frac{\rho}{2}\|z - x^t + u^t\|_2^2$
    $\phantom{z^{t+1}} = \underset{z}{\operatorname{argmin}}\ \frac{\lambda}{\rho}\|z\|_1 + \frac{1}{2}\|z - x^t + u^t\|_2^2$
    $x^{t+1} := \underset{x}{\operatorname{argmin}}\ \|Ax - b\|_2^2 + \frac{\rho}{2}\|z^{t+1} - x + u^t\|_2^2$
    $u^{t+1} := u^t + z^{t+1} - x^{t+1}$
end
```

Algorithm 5: ADMM algorithm for the LASSO in pseudocode.

We note that the two subproblems of minimization can be derived in closed forms. The subproblem of x is simply a convex quadratic function, and can be solved by setting the gradient to zero. We now consider the first subproblem:

$$\alpha^*(\gamma) := \underset{\alpha}{\operatorname{argmin}}\ \gamma\|\alpha\|_1 + \frac{1}{2}\|v - \alpha\|_2^2 \tag{A.1}$$

Theorem 8. *The closed-form solution to problem A.1 is*

$$\begin{aligned} \alpha^*(\gamma) &:= \mathcal{S}_\gamma(v) \\ &:= \left[\operatorname{sign}(v_i)\max\{|v_i| - \gamma, 0\}\right]_{i=1}^d \end{aligned}$$

where $\mathcal{S}_\gamma(v)$ is the soft-thresholding operator.

Therefore, an ADMM algorithm for the LASSO problem is as follows:

```
rho = 1;
maxiter = 10000;
x = zeros(size(A, 2), 1);
z = x;
u = 0;
cache = (A'*A + rho * eye(size(A, 2)));
cache_v = A' * b;
inv_cache = inv(cache);
for iter = 1 : maxiter
    z = soft_thresholding(x - u, lambda / rho);
    x = inv_cache * (cache_v + rho * (z + u));
    u = u + z - x;
end
x_algo = x;

function [r] = soft_thresholding(v, gamma)
    r = zeros(size(v));
    for i = 1 : length(v)
        if (v(i) > gamma)
            r(i) = v(i) - gamma;
        end
        if (v(i) < -gamma)
            r(i) = v(i) + gamma;
        end
    end
end
```

Algorithm 6: Matlab code for solving the LASSO with ADDM algorithm.

Appendix B

Algorithms for Square-root Lasso

Alternating direction method of multipliers. We consider an equivalent form of the square-root Lasso optimization problem:

$$\begin{aligned} & \min_x \quad \|Ax-b\|_2 + \lambda \|x\|_1 \\ = & \min_{x,y,z} \quad \|y\|_2 + \lambda \|z\|_1 \\ & \text{s.t.} \quad y-(Ax-b)=0 \\ & \qquad\;\; z-x=0 \end{aligned}$$

The ADMM algorithm for the square-root LASSO problem is thus:

```
for t := 0, 1, 2, ..., T do
    $y^{t+1} := \underset{y}{\operatorname{argmin}}\ \|y\|_2 + \frac{\rho}{2}\|y-(Ax^t-b)+u^t\|_2^2$
    $\quad = \underset{y}{\operatorname{argmin}}\ \frac{1}{\rho}\|y\|_2 + \frac{1}{2}\|y-(Ax^t-b)+u^t\|_2^2$
    $z^{t+1} := \underset{z}{\operatorname{argmin}}\ \lambda\|z\|_1 + \frac{\rho}{2}\|z-x^t+v^t\|_2^2$
    $\quad = \underset{z}{\operatorname{argmin}}\ \frac{\lambda}{\rho}\|z\|_1 + \frac{1}{2}\|z-x^t+v^t\|_2^2$
    $x^{t+1} := \underset{x}{\operatorname{argmin}}\ \frac{\rho}{2}\|y^{t+1}-(Ax^t-b)+u^t\|_2^2 + \frac{\rho}{2}\|z^{t+1}-x+v^t\|_2^2$
    $\quad = \underset{x}{\operatorname{argmin}}\ \frac{1}{2}\|y^{t+1}-(Ax^t-b)+u^t\|_2^2 + \frac{1}{2}\|z^{t+1}-x+v^t\|_2^2$
    $u^{t+1} := u^t + y^{t+1} - (Ax^{t+1}-b)$
    $v^{t+1} := v^t + z^{t+1} - x^{t+1}$
end
```

Algorithm 7: ADMM algorithm for square-root LASSO.

We consider the following subproblem:

$$\alpha^*(\gamma) := \underset{\alpha}{\operatorname{argmin}}\ \gamma\|\alpha\|_2 + \frac{1}{2}\|v-\alpha\|_2^2 \tag{B.1}$$

Theorem 9. *The closed-form solution to problem B.1 is*

$$\begin{array}{rcl} \alpha^*(\gamma) & := & \mathcal{G}_\gamma(v) \\ & := & \begin{cases} \left(\frac{\|v\|_2-\gamma}{\|v\|_2}\right) v & \text{if } \|v\|_2 \geq \gamma \\ 0 & \text{otherwise} \end{cases} \end{array}$$

where $\mathcal{G}_\gamma(v)$ is the group soft-thresholding operator.

The ADMM algorithm for the square-root LASSO thus becomes:

```
for t := 0, 1, 2, ..., T do
    $y^{t+1} := \mathcal{G}_{1/\rho}(Ax - b - u)$
    $z^{t+1} := \mathcal{S}_{\lambda/\rho}(x - v)$
    $x^{t+1} := \left(A^TA + I\right)^{-1}\left(A^T(b + y + u) + z + v\right)$
    $u^{t+1} := u^t + y^{t+1} - (Ax^{t+1} - b)$
    $v^{t+1} := v^t + z^{t+1} - x^{t+1}$
end
```

Algorithm 8: ADMM algorithm for square-root LASSO.

Square-root LASSO elastic net. We combine the results in Theorem 8 and Theorem 9 for a variant of the elastic net problem:

$$\begin{array}{rll} & \min\limits_x & \|Ax-b\|_2 + \lambda\|x\|_1 + \epsilon\|x\|_2 \\ = & \min\limits_x & \|y\|_2 + \lambda\|z\|_1 + \epsilon\|w\|_2 \\ & \text{s.t.} & y - (Ax - b) = 0 \\ & & z - x = 0 \\ & & w - x = 0 \end{array}$$

The algorithm can therefore be derived similarly as follows:

```
for t := 0, 1, 2, ..., T do
    $y^{t+1} := \mathcal{G}_{1/\rho}(Ax - b - u)$
    $z^{t+1} := \mathcal{S}_{\lambda/\rho}(x - v)$
    $w^{t+1} := \mathcal{G}_{\epsilon/\rho}(x - r)$
    $x^{t+1} := \left(A^TA + 2I\right)^{-1}\left(A^T(b + y + u) + z + v + w + r\right)$
    $u^{t+1} := u^t + y^{t+1} - (Ax^{t+1} - b)$
    $v^{t+1} := v^t + z^{t+1} - x^{t+1}$
    $r^{t+1} := r + w^{t+1} - x^{t+1}$
end
```

Algorithm 9: ADMM algorithm for square-root elastic net.

We note that when the matrix A is wide, we can use the Woodbury matrix identity to solve the system of linear equations:

$$\begin{array}{rcl} \left(A^TA + \kappa I\right)^{-1} & = & \frac{1}{\kappa}I - \frac{1}{\kappa^2}A^T\left(I + \frac{1}{\kappa}AA^T\right)^{-1}A \\ \left(A^TA + \kappa I\right)^{-1}v & = & \frac{1}{\kappa}v - \frac{1}{\kappa^2}A^T\left(I + \frac{1}{\kappa}AA^T\right)^{-1}Av \end{array}$$

where we can cache the (small) inverse matrix $W := \left(I + \frac{1}{\kappa} AA^T\right)^{-1}$ to compute the decision variable x^{t+1}.

Appendix C

Algorithms for SPCA

Matlab implementation. In this section, we present our implementation in Matlab of the modified power iteration method for SPCA (Algorithm 2). This implementation uses two main sub-procedures, one is for the power iteration step and another for the hard thresholding function.

```
original_Matrix = M;
original_word_list = word_list;
fileID = fopen('output.txt','w');
for num_principal_components = 1 : 4
    fprintf(fileID, '***** TOPIC %d *****\n', num_principal_components);

    [p, ind_p, q, ind_q] = SPCA_power_iteration(M, threshold_column_vector, ...
        threshold_row_vector, epsilon_objective_function);

    for i = 1 : size(ind_q, 1)
        k = ind_q(i);
        fprintf('%d-th %.4f: %s\n', k, full(q(k, 1)), char(word_list(k)));
        fprintf(fileID, '%.4f\t%s\n', full(q(k, 1)), char(word_list(k)));
    end
    fprintf(fileID, '-----\n');

    for i = 1 : size(ind_p, 1)
        k = ind_p(i);
        fprintf('CATEGORY (%s), %d-th %.4f: %s\n', ...
            char(category_of_document_list(k)), k, full(p(k, 1)), ...
            char(document_list(k)));
        fprintf(fileID, '%.4f\t%s\t%s\n', full(p(k, 1)), ...
            char(document_list(k)), char(category_of_document_list(k)));
    end

    M(:, ind_q) = [];
    word_list(ind_q) = [];
    M(ind_p, :) = [];
    document_list(ind_p) = [];
    category_of_document_list(ind_p) = [];
end
word_list = original_word_list;
M = original_Matrix;
fclose(fileID);

function [p, ind_p, q, ind_q] = SPCA_power_iteration(M, ...
    threshold_column_vector, threshold_row_vector, epsilon)
    p = ones(size(M, 1), 1); % Init to all ones
    q = ones(size(M, 2), 1); % Init to all ones
    objective_function = inf;
    while (true)
        p_new = M * q;
        [p, ind_p] = utility_threshold(p_new, threshold_column_vector);
        p = p / norm(p);

        q_new = M' * p;
        [q, ind_q] = utility_threshold(q_new, threshold_row_vector);

        new_objective_function = norm(M - p * q', 'fro');
        if (abs(new_objective_function - objective_function) ≤ epsilon)
            break;
        end
        objective_function = new_objective_function;
    end
end

function [x, ind_v] = utility_threshold(v, count)
    [~,ind_v]=sort(full(abs(v)),'descend');
    ind_v = ind_v(1: min(size(ind_v, 1), count));
    x=sparse(size(v,1), size(v,2));
    x(ind_v) = v(ind_v);
end
```

Python library. We also provide our Python library for Sparse principal component analysis problem. This Python implementation was based on Python 2.7 with two scientific computing packages scipy and numpy.

```
import scipy.sparse
import math
import numpy
import csv
import scipy.sparse

class Options:
    def __init__(self, threshold_m, threshold_n,
        max_iteration, tolerance, num_pc):
        self.threshold_m = threshold_m
        self.threshold_n = threshold_n
        self.max_iteration = max_iteration
        self.tolerance = tolerance
        self.num_pc = num_pc

class Parser:
    @staticmethod
    def load_matrix(matrix_file, offset=0):
        cr = csv.reader(open(matrix_file))
        rows=[]
        cols=[]
        entries=[]
        for triplet in cr:
            rows.append(int(triplet[0])+offset)
            cols.append(int(triplet[1])+offset)
            entries.append(int(triplet[2]))
        rows = numpy.array(rows)
        cols = numpy.array(cols)
        entries = numpy.array(entries)

        return scipy.sparse.csc_matrix((entries,(rows,cols)))

    @staticmethod
    def load_dict(dict_file):
        cr = csv.reader(open(dict_file))
        words=[]
        for word_count_pair in cr:
            words.append(word_count_pair[0])
        return words
    @staticmethod
    def load_title(title_file):
        cr = csv.reader(open(title_file))
        titles=[]
        for title_count_pair in cr:
            titles.append(title_count_pair[0])
        return titles
    @staticmethod
    def load_category(category_file):
        cr = csv.reader(open(category_file))
        categories=[]
        for category_count_pair in cr:
            categories.append(category_count_pair[0])
        return categories

class Iterator:
    def __init__(self, options, M):
        self.M = scipy.sparse.csc_matrix(M)
        self.options = options
```

```
def thresh(self, s_vector, num_entries):
    listOfValues = s_vector.T.todense().tolist()[0]
    indices = self.sort(listOfValues)[1]
    thresholded_vec = numpy.zeros(shape=s_vector.T.shape)
    for i in range(num_entries):
        index = indices[i]
        thresholded_vec[0, index] = listOfValues[index]
    return scipy.sparse.csr_matrix(thresholded_vec).T

def top_k_ind(self, s_vector, num_entries):
    listOfValues = s_vector.T.todense().tolist()[0]
    return self.sort(listOfValues)

def sort(self, vector):
    indices = range(len(vector))
    val_ind = map (lambda x : (abs(vector[x]),x), indices)
    sorted_val_ind = sorted(val_ind)
    sorted_val_ind.reverse()
    sorted_indices = map (lambda x : x[1], sorted_val_ind)
    sorted_values = map (lambda x : x[0], sorted_val_ind)
    return (sorted_values, sorted_indices)
def single_iteration (self):

    M = self.M
    m = self.M.shape[0]
    n = self.M.shape[1]
    k_p = self.options.threshold_m
    k_q = self.options.threshold_n
    tolerance = self.options.tolerance
    max_iteration = self.options.max_iteration


    ini = self.ini_mat();
    p = ini[0]
    q = ini[1]
    obj0 = float("inf")
    converged = False
    iter = 0
    while not converged:
        p_new = self.thresh((M*q), k_p)
        p = p_new
        q_new = self.thresh((p.T * M).T, k_q)
        q = q_new/math.sqrt((q_new.T.dot(q_new)).data[0])
        q_test = (p.T * M)
        updated_mat = M-p_new*(q_new.T)
        obj1 = updated_mat.data.dot(updated_mat.data)
        if (abs(obj1-obj0) ≤ tolerance or iter ≥ max_iteration):
            converged = 1
        iter = iter + 1
        obj0 = obj1
    return (p,q)

def multiple_iterations (self):
    term_inds = []
    doc_inds = []
    term_vals = []
    doc_vals = []
    for i in range(self.options.num_pc):
        print 'handling pc# ' + str(i)
        (p,q) = self.single_iteration()
        (p_val, p_ind) = self.top_k_ind(p, self.options.threshold_m)
        (q_val, q_ind) = self.top_k_ind(q, self.options.threshold_n)
        doc_inds.append(p_ind[0:self.options.threshold_m])
        term_inds.append(q_ind[0:self.options.threshold_n])
        doc_vals.append(p_val[0:self.options.threshold_m])
        term_vals.append(q_val[0:self.options.threshold_n])
        self.remove_cols(q_ind[0:self.options.threshold_n])
        self.remove_rows(p_ind[0:self.options.threshold_m])

    return ((doc_vals,doc_inds),(term_vals,term_inds))

@staticmethod
def run(matrix_path, dict_path, title_path, category_path):
    print 'running'
    matrix = Parser.Parser.load_matrix(matrix_path,-1)
    dict = Parser.Parser.load_dict(dict_path)
    titlelist = Parser.Parser.load_title(title_path)
    categorylist = Parser.Parser.load_title(category_path)
    opt = Options.Options(15, 15, 50, 0.0001, 10)
    it = Iterator(opt, matrix)
    word_pcs = []
    title_pcs = []
    ((p_vals,p_inds),(q_vals,q_inds)) = it.multiple_iterations()
    outputfile = open('./output/output.txt','w')
    for pc, pcv in zip(q_inds, q_vals):
        word_pc = []
        for q_ind, q_val in zip(pc, pcv):
            word_pc.append('%.4f\t%s' % (q_val, dict[q_ind]))
        word_pcs.append(word_pc)
        todel = sorted(pc);
        todel.reverse();
        for q_ind in todel:
            del dict[q_ind]
    for pc, pcv in zip(p_inds, p_vals):
        title_pc = []
        for p_ind,p_val in zip(pc,pcv):
            title_pc.append('%.4f\t%s\t%s' %
                (p_val,titlelist[p_ind],categorylist[p_ind]))
        title_pcs.append(title_pc)
        todel = sorted(pc);
        todel.reverse();
        for p_ind in todel:
            del titlelist[p_ind]
            del categorylist[p_ind]
    topic_num = 0
    for wpc,tpc in zip(word_pcs,title_pcs):
        outputfile.write( '***** TOPIC %d *****\n' % topic_num)
        for row in wpc:
            outputfile.write(row+'\n')
        outputfile.write('------\n')
        for row in tpc:
            outputfile.write(row+'\n')
        outputfile.write('\n')
        topic_num = topic_num + 1

def remove_rows(self, inds_to_remove):
    self.M = self.M.T
    self.remove_cols(inds_to_remove)
    self.M = self.M.T
def remove_cols(self, inds_to_remove):
    inds_to_remove = sorted(inds_to_remove)
    inds_to_remove.reverse()
    for ind in inds_to_remove:
        self.remove_col(ind)

def remove_col(self, ind):
    if ind == 0:
        self.M = self.M[:,ind+1:].tocsc()
    elif ind == (self.M.shape[1]-1):
        self.M = self.M[:,0:ind].tocsc()
    else:
        self.M = scipy.sparse.hstack(
            [self.M[:,0:ind],self.M[:,ind+1:]]).tocsc()

def ini_mat(self):
    p = scipy.sparse.csc_matrix(numpy.ones(shape=(self.M.shape[0],1)))
    q = scipy.sparse.csc_matrix(numpy.ones(shape=(self.M.shape[1],1)))
    p = p/math.sqrt(p.T.dot(p).data[0])
    q = q/math.sqrt(q.T.dot(q).data[0])
    return (p,q)
```