Large-Scale Sparse Principal Component Analysis with Application to Text Data

Authored by:

Laurent E. Ghaoui
Youwei Zhang

Abstract

Sparse PCA provides a linear combination of small number of features that maximizes variance across data. Although Sparse PCA has apparent advantages compared to PCA, such as better interpretability, it is generally thought to be computationally much more expensive. In this paper, we demonstrate the surprising fact that sparse PCA can be easier than PCA in practice, and that it can be reliably applied to very large data sets. This comes from a rigorous feature elimination pre-processing result, coupled with the favorable fact that features in real-life data typically have exponentially decreasing variances, which allows for many features to be eliminated. We introduce a fast block coordinate ascent algorithm with much better computational complexity than the existing first-order ones. We provide experimental results obtained on text corpora involving millions of documents and hundreds of thousands of features. These results illustrate how Sparse PCA can help organize a large corpus of text data in a user-interpretable way, providing an attractive alternative approach to topic models.

1 Paper Body

The sparse Principal Component Analysis (Sparse PCA) problem is a variant of the classical PCA problem, which accomplishes a trade-off between the explained variance along a normalized vector, and the number of non-zero components of that vector. Sparse PCA not only brings better interpretation [1], but also provides statistical regularization [2] when the number of samples is less than the number of features. Various researchers have proposed different formulations and algorithms for this problem, ranging from ad-hoc methods such as factor rotation techniques [3] and simple thresholding [4], to greedy algorithms [5, 6]. Other algorithms include SCoTLASS by [7], SPCA by [8], the regularized SVD method by [9] and the generalized power method by [10]. These algorithms are based on non-convex formulations, and may only converge

to a local optimum. The ‘1 -norm based semidefinite relaxation DSPCA, as introduced in [1], does guarantee global convergence and as such, is an attractive alternative to local methods. In fact, it has been shown in [1, 2, 11] that simple ad-hoc methods, and the greedy, SCoTLASS and SPCA algorithms, often underperform DSPCA. However, the first-order algorithm for solving ? DSPCA, as developed in [1], has a computational complexity of O(n4 log n), with n the number of 1

features, which is too high for many large-scale data sets. At first glance, this complexity estimate indicates that solving sparse PCA is much more expensive than PCA, since we can compute one principal component with a complexity of O(n2). In this paper we show that solving DSPCA is in fact computationally easier than PCA, and hence can be applied to very large-scale data sets. To achieve that, we first view DSPCA as an approximation to a harder, cardinality-constrained optimization problem. Based on that formulation, we describe a safe feature elimination method for that problem, which leads to an often important reduction in problem size, prior to solving the problem. Then we develop a block coordinate ascent algorithm, with a computational complexity of O(n3) to solve DSPCA, which is much faster than the firstorder algorithm proposed in [1]. Finally, we observe that real data sets typically allow for a dramatic reduction in problem size as afforded by our safe feature elimination result. Now the comparison between sparse PCA and PCA becomes O(? n3) v.s. O(n2) with n ? n, which can make sparse PCA surprisingly easier than PCA. In Section 2, we review the ‘1 -norm based DSPCA formulation, and relate it to an approximation to the ‘0 -norm based formulation and highlight the safe feature elimination mechanism as a powerful pre-processing technique. We use Section 3 to present our fast block coordinate ascent algorithm. Finally, in Section 4, we demonstrate the efficiency of our approach on two large data sets, each one containing more than 100,000 features. Notation. R(Y) denotes the range of matrix Y , and Y ? its pseudo-inverse. The notation log refers to the extended-value function, with log x = ?? if x ? 0.

2

Safe Feature Elimination

Primal problem. Given a n ? n positive-semidefinite matrix ?, the ?sparse PCA? problem introduced in [1] is : ? = max Tr ?Z ? ?kZk1 : Z 0, Tr Z = 1 (1) Z

where ? ? 0 is a parameter encouraging sparsity. Without loss of generality we may assume that ? 0. Problem (1) is in fact a relaxation to a PCA problem with a penalty on the cardinality of the variable: ? = max xT ?x ? ?kxk0 : kxk2 = 1 x

(2)

Where kxk0 denotes the cardinality (number of non-zero elemements) in x. This can be seen by first writing problem (2) as: p max Tr ?Z ? ? kZk0 : Z 0, Tr Z = 1, Rank(Z) = 1 Z p where kZk is the cardinality (number of non-zero elements) of Z. Since kZk ? kZk0 kZkF = 0 1 p kZk0 , we obtain the relaxation max Tr ?Z ? ?kZk1 : Z 0, Tr Z = 1, Rank(Z) = 1 Z

Further drop the rank constraint, leading to problem (1). By viewing prob-

lem (1) as a convex approximation to the non-convex problem (2), we can leverage the safe feature elimination theorem first presented in [6, 12] for problem (2): Theorem 2.1 Let ? = AT A, where A = (a1 , . . . , an) ? Rm?n . We have ? = max

k?k2 =1

n X ((aTi ?)2 ? ?)+ . i=1

An optimal non-zero pattern corresponds to indices i with ? ¡ (aTi ?)2 at optimum. We observe that the i-th feature is absent at optimum if (aTi ?)2 ? ? for every ?, k?k2 = 1. Hence, we can safely remove feature i ? {1, . . . , n} if ?ii = aTi ai ¡ ? 2

(3)

A few remarks are in order. First, if we are interested in solving problem (1) as a relaxation to problem (2), we first calculate and rank all the feature variances, which takes O(nm) and O(n log(n)) respectively. Then we can safely eliminate any feature with variance less than ?. Second, the elimination criterion above is conservative. However, when looking for extremely sparse solutions, applying this safe feature elimination test with a large ? can dramatically reduce problem size and lead to huge computational savings, as will be demonstrated empirically in Section 4. Third, in practice, when PCA is performed on large data sets, some similar variance-based criteria is routinely employed to bring problem sizes down to a manageable level. This purely heuristic practice has a rigorous interpretation in the context of sparse PCA, as the above theorem states explicitly the features that can be safely discarded.

3

Block Coordinate Ascent Algorithm

The first-order algorithm developed in [1] to solve problem (1) has a computational complexity ? of O(n4 log n). With a theoretical convergence rate of O(1), the DSPCA algorithm does not converge fast in practice. In this section, we develop a block coordinate ascent algorithm with better dependence on problem size (O(n3)), that in practice converges much faster. Failure of a direct method. We seek to apply a ?row-by-row? algorithm by which we update each row/column pair, one at a time. This algorithm appeared in the specific context of sparse covariance estimation in [13], and extended to a large class of SDPs in [14]. Precisely, it applies to problems of the form min f (X) ? ? log det X : L ? X ? U, X 0, (4) X

where X = X T is a n ? n matrix variable, L, U impose componentwise bounds on X, f is convex, and ? ¿ 0. However, if we try to update the row/columns of Z in problem (1), the trace constraint will imply that we never modify the diagonal elements of Z. Indeed at each step, we update only one diagonal element, and it is entirely fixed given all the other diagonal elements. The row-by-row algorithm does not directly work in that case, nor in general for SDPs with equality constraints. The authors in [14] propose an augmented Lagrangian method to deal with such constraints, with a complication due to the choice of appropriate penalty parameters. In our case, we can apply a technique resembling the augmented Lagrangian technique, without this added complication. This is due to the homogeneous nature of the objective function and of

the conic constraint. Thanks to the feature elimination result 2 (Thm. 2.1), we can always assume without loss of generality that ? ¡ ?min := min1?i?n ?ii . Direct augmented Lagrangian technique. We can express problem (1) as 1 2 1 ? = max Tr ?X ? ?kXk1 ? (Tr X)2 : X 0. (5) X 2 2 This expression results from the change of variable X = ?Z, with Tr Z = 1, and ? ? 0. Optimizing 2 over ? ? 0, and exploiting ? ¿ 0 (which comes from our assumption that ? ¡ ?min), leads to the ? result, with the optimal scaling factor ? equal to ?. An optimal solution Z to (1) can be obtained from an optimal solution X ? to the above, via Z ? = X ? /?. (In fact, we have Z ? = X ? / Tr(X ?).) To apply the row-by-row method to the above problem, we need to consider a variant of it, with a strictly convex objective. That is, we address the problem 1 max Tr ?X ? ?kXk1 ? (Tr X)2 + ? log det X, : X 0, X 2

(6)

where ? ¿ 0 is a penalty parameter. SDP theory ensures that if ? = /n, then a solution to the above problem is -suboptimal for the original problem [15]. Optimizing over one row/column. Without loss of generality, we consider the problem of updating the last row/column of the matrix variable X. Partition the latter and the covariance matrix S as

Y y S s X= , ?= , yT x sT ? 3

where Y, S ? R(n?1)?(n?1) , y, s ? Rn?1 , and x, ? ? R. We are considering the problem above, where Y is fixed, and (y, x) ? Rn is the variable. We use the notation t := Tr Y . The conic constraint X 0 translates as y T Y ? y ? x, y ? R(Y), where R(Y) is the range of the matrix Y . We obtain the sub-problem

2(y T s ? ?kyk1) + (? ? ?)x ? 21 (t + x)2 ? := max : y ? R(Y). (7) x,y +? log(x ? y T Y ? y) Simplifying the sub-problem. We can simplify the above problem, in particular, avoid the step of forming the pseudo-inverse of Y , by taking the dual of problem (7). Using the conjugate relation, valid for every ? ¿ 0: log ? + 1 = min z? ? log z, z¿0

and with f (x) := (? ? ?)x ?+?

=

? 21 (t T

2

+ x) , we obtain

max 2(y s ? ?kyk1) + f (x) + ? min z(x ? y T Y ? y) ? log z

z¿0

y?R(Y)

=

min max 2(y s ? ?kyk1 ? ?zy Y ? y) + max (f (x) + ?zx) ? ? log z

=

min h(z) + 2g(z)

T

T

x

z¿0 y?R(Y) z¿0

where, for z ¿ 0, we define h(z) := ?? log z + max (f (x) + ?zx) x

1 ?? log z + max ((? ? ? + ?z)x ? (t + x)2) x 2 1 1 = ? t2 ? ? log z + max ((? ? ? ? t + ?z)x ? x2) x 2 2 1 1 2 = ? t ? ? log z + (? ? ? ? t + ?z)2 2 2 with the following relationship at optimum: x = ? ? ? ? t + ?z. In addition, ?z T ? g(z) := max y T s ? ?kyk1 ? (y Y y) 2 y?R(Y) ?z T ? = max y T s + min y T v ? (y Y y) 2 y?R(Y) v : kvk? ?? ?z T ? = min max (y T (s + v) ? (y Y y)) 2 v : kvk? ?? y?R(Y) ?z T ? = min max (y T u ? (y Y y)) 2 u : ku?sk? ?? y?R(Y) 1 T = min u Y u. u : ku?sk? ?? 2?z with the following relationship at optimum: 1 y= Y u. ?z =

(8)

(9)

Putting all this together, we obtain the dual of problem (7): with ? 0 := ?+?+ 12 t2 , and c := ????t, we have 1 T 1 ? 0 = min u Y u ? ? log z + (c + ?z)2 : z ¿ 0, ku ? sk? ? ?. u,z ?z 2 Since ? is small, we can avoid large numbers in the above, with the change of variable ? = ?z: 1 1 ? 0 ? ? log ? = min uT Y u ? ? log ? + (c + ?)2 : ? ¿ 0, ku ? sk? ? ?. (10) u,? ? 2 4

Solving the sub-problem. Problem (10) can be further decomposed into two stages. First, we solve the box-constrained QP R2 := min uT Y u : ku ? sk? ? ?, u

(11)

using a simple coordinate descent algorithm to exploit sparsity of Y . Without loss of generality, we consider the problem of updating the first coordinate of u. Partition u, Y and s as

y1 y?T ? s1 u= , Y = , s = , u ? s? y? Y? Where, Y? ? R(n?2)?(n?2) , u ?, y?, s? ? Rn?2 , y1 , s1 ? R are all fixed, while ? ? R is the variable. We obtain the subproblem min y1 ? 2 + (2? yT u ?)? : k? ? s1 k ? ? ?

for which we can solve for ? analytically using the formula given below. ? T y?T u ? ? if ks1 + y?y1u? k ? ?, y1 ¿ 0, ? ? ? y1 ?=

s1 ? ? ? ? ? s +? 1

if ? if ?

y?T u ? y1 T y? u ? y1

¡ s1 ? ?, y1 ¿ 0 or if y?T u ? ¿ 0, y1 = 0,

(12)

(13)

T

¿ s1 + ?, y1 ¿ 0 or if y? u ? ¡= 0, y1 = 0.

Next, we set ? by solving the one-dimensional problem: R2 1 ? ? log ? + (c + ?)2 . ? ¿0 ? 2 The above can be reduced to a bisection problem over ? , or by solving a polynomial equation of degree 3. min

Obtaining the primal variables. Once the above problem is solved, we can obtain the primal variables y, x, as follows. Using formula (9), with ?z = ? , we set y = ?1 Y u. For the diagonal element x, we use formula (8): x = c + ? = ? ? ? ? t + ? . Algorithm summary. We summarize the above derivations in Algorithm 1. Notation: for any symmetric matrix A ? Rn?n , let Aij denote the matrix produced by removing row i and column j. Let Aj denote column j (or row j) with the diagonal element Ajj removed. Convergence and complexity. Our algorithm solves DSPCA by first casting it to problem (6),

which is in the general form (4). Therefore, the convergence result from [14] readily applies and hence every limit point that our block coordinate ascent algorithm converges to is the global optimizer. The simple coordinate descent algorithm solving problem (11) only involves a vector product and can take sparsity in Y easily. To update each column/row takes O(n2) and there are n such columns/rows in total. Therefore, our algorithm has a computational complexity of O(Kn3), where K is the number of sweeps through columns. In practice, K is fixed at a number independent of problem size (typically ? K = 5). Hence our algorithm has better dependence on the problem size compared to O(n4 log n) required of the first order algorithm developed in [1]. Fig 1 shows that our algorithm converges much faster than the first order algorithm. On the left, both algorithms are run on a covariance matrix ? = F T F with F Gaussian. On the right, the covariance matrix comes from a ?spiked model? similar to that in [2], with ? = uuT +V V T /m, where u ? Rn is the true sparse leading eigenvector, with Card(u) = 0.1n, V ? Rn?m is a noise matrix with Vij ? N (0, 1) and m is the number of observations.

4

Numerical Examples

In this section, we analyze two publicly available large data sets, the NY-Times news articles data and the PubMed abstracts data, available from the UCI Machine Learning Repository [16]. Both 5

Algorithm 1 Block Coordinate Ascent Algorithm Input: The covariance matrix ?, and a parameter ? ¿ 0. 1: Set X (0) = I 2: repeat 3: for j = 1 to n do 4: Let X (j?1) denote the current iterate. Solve the box-constrained quadratic program (j?1)

R2 := min uT Xjj u : ku ? ?j k? ? ? u

using the coordinate descent algorithm Solve the one-dimensional problem

5:

min ? ¿0

R2 1 (j?1) ? ? log ? + (?jj ? ? ? Tr Xjj + ?)2 ? 2

using a bisection method, or by solving a polynomial equation of degree 3. (j) (j?1) First set Xjj = Xjj , and then set both X (j) ?s column j and row j using

6:

(j)

Xj

=

1 (j?1) X u ? jj (j?1)

(j)

Xjj = ?jj ? ? ? Tr Xjj + ? 7: end for 8: Set X (0) = X (n) 9: until convergence

5

5

10

10

4

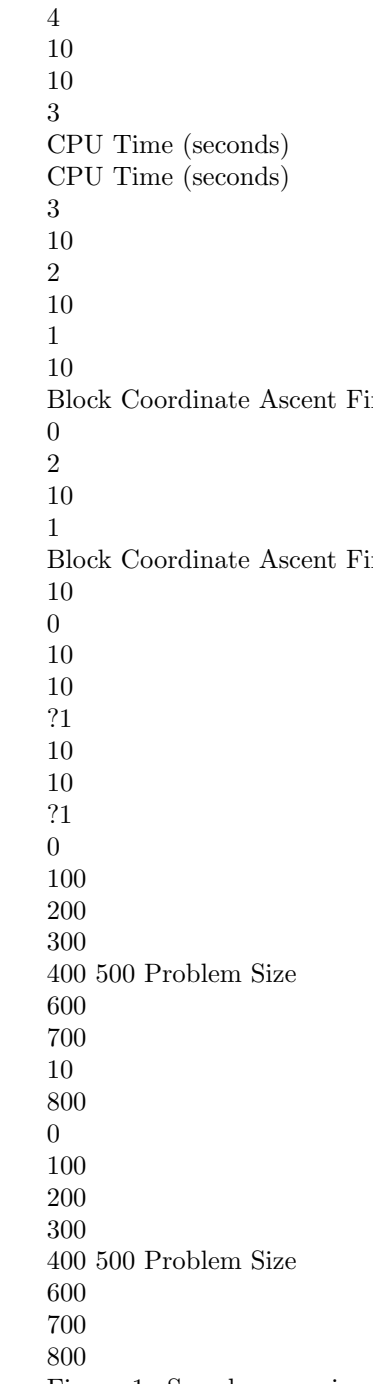

Figure 1: Speed comparisons between Block Coordinate Ascent and First-Order

text collections record word occurrences in the form of bag-of-words. The NYTtimes text collection contains 300, 000 articles and has a dictionary of 102, 660 unique words, resulting in a file of size 1 GB. The even larger PubMed data set has 8, 200, 000 abstracts with 141, 043 unique words in them, giving a file of size 7.8 GB. These data matrices are so large that we cannot even load them into memory all at once, which makes even the use of classical PCA difficult. However with the preprocessing technique presented in Section 2 and the block coordinate ascent algorithm developed in Section 3, we are able to perform sparse PCA analysis of these data, also thanks to the fact that variances of words decrease drastically when we rank them as shown in Fig 2. Note that the feature elimination result only requires the computation of each feature?s variance, and that this task is easy to parallelize. By doing sparse PCA analysis of these text data, we hope to find interpretable principal components that can be used to summarize and explore the large corpora. Therefore, we set the target cardinality for each principal component to be 5. As we run our algorithm with a coarse range of ? to search for 6

0
0
10
10
?1
?1
10
10
?2
?2
10 Variance
Variance
10
?3
10
?4
?3
10
?4
10
10
?5
?5
10
10
?6
?6
10
10 0
2

4
6 Word Index
8
10
12
0
5
10 Word Index
4
x 10
15 4
x 10

Figure 2: Sorted variances of 102,660 words in NYTimes (left) and 141,043 words in PubMed (right)

a solution with the given cardinality, we might end up accepting a solution with cardinality close, but not necessarily equal to, 5, and stop there to save computational time. The top 5 sparse principal components are shown in Table 1 for NYTimes and in Table 2 for PubMed. Clearly the first principal component for NYTimes is about business, the second one about sports, the third about U.S., the fourth about politics and the fifth about education. Bear in mind that the NYTimes data from UCI Machine Learning Repository ?have no class labels, and for copyright reasons no filenames or other document-level metadata? [16]. The sparse principal components still unambiguously identify and perfectly correspond to the topics used by The New York Times itself to classify articles on its own website. Table 1: Words associated with the top 5 sparse principal components in NYTimes 1st PC (6 words) 2nd PC (5 words) 3rd PC (5 words) 4th PC (4 words) 5th PC (4 words) million point official president school percent play government campaign program business team united states bush children company season us administration student market game attack companies

After the pre-processing steps, it takes our algorithm around 20 seconds to search for a range of ? and find one sparse principal component with the target cardinality (for the NYTimes data in our current implementation on a MacBook laptop with 2.4 GHz Intel Core 2 Duo processor and 2 GB memory). Table 2: 1st PC (5 words) patient cell treatment protein disease

Words associated with the top 5 sparse principal components in PubMed 2nd PC (5 words) 3rd PC (5 words) 4th PC (4 words) 5th PC (4 words) effect human tumor year level expression mice infection activity receptor cancer age concentration binding maligant children rat carcinoma child

A surprising finding is that the safe feature elimination test, combined with the fact that word variances decrease rapidly, enables our block coordinate ascent algorithm to work on covariance matrices of order at most n = 500, instead of the full order (n = 102660) covariance matrix for NYTimes, so as to find a solution with cardinality of around 5. In the case of PubMed, our algorithm only needs to work on covariance matrices of order at most n = 1000, instead of the full order (n = 141, 043) 7

covariance matrix. Thus, at values of the penalty parameter ? that target cardinality of 5 commands, we observe a dramatic reduction in problem sizes, about 150 ? 200 times smaller than the original sizes respectively. This motivates our conclusion that sparse PCA is in a sense, easier than PCA itself.

5

Conclusion

The safe feature elimination result, coupled with a fast block coordinate ascent algorithm, allows to solve sparse PCA problems for very large scale, real-life data sets. The overall method works especially well when the target cardinality of the result is small, which is often the case in applications where interpretability by a human is key. The algorithm we proposed has better computational complexity, and in practice converges much faster than, the first-order algorithm developed in [1]. Our experiments on text data also show that the sparse PCA can be a promising approach towards summarizing and organizing a large text corpus.

2 References

[1] A. d?Aspremont, L. El Ghaoui, M. Jordan, and G. Lanckriet. A direct formulation of sparse PCA using semidefinite programming. SIAM Review, 49(3), 2007. [2] 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. [3] I. T. Jolliffe. Rotation of principal components: choice of normalization constraints. Journal of Applied Statistics, 22:29?35, 1995. [4] J. Cadima and I. T. Jolliffe. Loadings and correlations in the interpretation of principal components. Journal of Applied Statistics, 22:203?214, 1995. [5] B. Moghaddam, Y. Weiss, and S. Avidan. Spectral bounds for sparse PCA: exact and greedy algorithms. Advances in Neural Information Processing Systems, 18, 2006. [6] 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. [7] I. T. Jolliffe, N.T. Trendafilov, and M. Uddin. A modified principal component technique based on the LASSO. Journal of Computational and Graphical Statistics, 12:531?547, 2003. [8] H. Zou, T. Hastie, and R. Tibshirani. Sparse Principal Component Analysis. Journal of Computational & Graphical Statistics, 15(2):265?286, 2006. [9] Haipeng Shen and Jianhua Z. Huang. Sparse principal component analysis via regularized low rank matrix approximation. J. Multivar. Anal., 99:1015?1034, July 2008. [10] M. Journ?ee, Y. Nesterov, P. Richt?arik, and R. Sepulchre. Generalized power method for sparse principal component analysis. arXiv:0811.4724, 2008. [11] 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 Semide nite, Cone and Polynomial Optimization: Theory, Algorithms, Software and Applications. Springer, 2011. To appear. [12] L. El Ghaoui. On the quality of a semidefinite programming bound for sparse principal component analysis. arXiv:math/060144, February 2006. [13] 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. [14] Zaiwen Wen, Donald Goldfarb, Shiqian Ma, and Katya Scheinberg. Row by row methods for semidefinite programming. Technical report, Dept of IEOR, Columbia University, 2009. [15] Stephen Boyd and Lieven Vandenberghe. Convex Optimization. Cambridge University Press, New York, NY, USA, 2004. [16] A. Frank and A. Asuncion. UCI machine learning repository, 2010.

8