Sparse PCA with applications in finance

A. d’Aspremont, L. El Ghaoui, M. Jordan, G. Lanckriet

Princeton University, U.C. Berkeley & U.C. San Diego

Available online at www.princeton.edu/~aspremon
Introduction

Principal Component Analysis (PCA): classic tool in multivariate data analysis

- **Input**: a *covariance* matrix $A$
- **Output**: a sequence of *factors* ranked by *variance*
- Each factor is a *linear* combination of the problem variables

Typical use: reduce the number of *dimensions* of a model while maximizing the *information* (variance) contained in the simplified model.

Numerically, just an eigenvalue decomposition of the covariance matrix:

$$A = \sum_{i=1}^{n} \lambda_i x_i x_i^T$$
Portfolio Hedging

Hedging problem:

• Market is composed of $N$ assets with price $S_{i,t}$ at time $t$

• Let $C$ be the covariance matrix of the assets

• $P_t$ is the value of a portfolio of assets with coefficients $u_i$:

\[ P_t = \sum_{i=1}^{N} u_i S_{i,t} \]

• The market factors and corresponding variances are given by:

\[ C = \sum_{i=1}^{n} \lambda_i x_i x_i^T \]
Portfolio Hedging

- We can hedge some of the risk using the $k$ most important market factors:

$$P_t = \sum_{i=1}^{k} (u^T x_i) F_{i,t} + \varepsilon_t, \quad \text{with } F_{i,t} = x_i^T S_t$$

- Usually $k = 3$. On interest rate markets the first three factors are level, spread and convexity.

- Problem: the factors $x_i$ usually assign a nonzero weight to all assets $S_i$

- This means large fixed transaction costs when hedging.
Sparse PCA: Applications

Can we get *sparse* factors $x_i$ instead?

- **Portfolio hedging**: sparse factors mean less assets in the portfolio, hence less transaction costs.

- **Side effects**: minimize proportional transaction costs, robustness interpretation.

- **Other applications**: image processing, gene expression data analysis, multi-scale data processing.
**Variational formulation**

We can rewrite the previous problem as:

\[
\begin{align*}
\max & \quad x^T Ax \\
\text{subject to} & \quad \|x\|_2 = 1.
\end{align*}
\]  

(1)

This problem is *easy*, its solution is again \( \lambda^{\text{max}}(A) \) at \( x_1 \).

Here however, we want a little bit more... We look for a *sparse* solution and solve instead:

\[
\begin{align*}
\max & \quad x^T Ax \\
\text{subject to} & \quad \|x\|_2 = 1 \\
& \quad \text{Card}(x) \leq k,
\end{align*}
\]  

(2)

where \( \text{Card}(x) \) denotes the cardinality (number of non-zero elements) of \( x \). This is non-convex and *numerically hard*. 
Related literature

Previous work:

• Cadima & Jolliffe (1995): the loadings with small absolute value are thresholded to zero.

• A non-convex method called SCoTLASS by Jolliffe, Trendafilov & Uddin (2003). (Same problem formulation)

• Zou, Hastie & Tibshirani (2004): a regression based technique called SPCA. Based on a representation of PCA as a regression problem. Sparsity is obtained using the LASSO Tibshirani (1996) a $l_1$ norm penalty.

Performance:

• These methods are either very suboptimal (thresholding) or lead to nonconvex optimization problems (SPCA).

• Regression: works for very large scale examples.
Semidefinite relaxation
Semidefinite relaxation

Start from:
\[
\begin{aligned}
\text{max} & \quad x^T A x \\
\text{subject to} & \quad \|x\|_2 = 1 \\
& \quad \text{Card}(x) \leq k,
\end{aligned}
\]

let \( X = xx^T \), and write everything in terms of the matrix \( X \):

\[
\begin{aligned}
\text{max} & \quad \text{Tr}(A X) \\
\text{subject to} & \quad \text{Tr}(X) = 1 \\
& \quad \text{Card}(X) \leq k^2 \\
& \quad X = xx^T.
\end{aligned}
\]

This is a strictly equivalent problem.
Semidefinite relaxation

From

\[
\begin{align*}
\max & \quad \text{Tr}(AX) \\
\text{subject to} & \quad \text{Tr}(X) = 1 \\
& \quad \text{Card}(X) \leq k^2 \\
& \quad X = xx^T.
\end{align*}
\]

We can go a little further and replace \( X = xx^T \) by an equivalent \( X \succeq 0, \ \text{Rank}(X) = 1, \) to get:

\[
\begin{align*}
\max & \quad \text{Tr}(AX) \\
\text{subject to} & \quad \text{Tr}(X) = 1 \\
& \quad \text{Card}(X) \leq k^2 \\
& \quad X \succeq 0, \ \text{Rank}(X) = 1,
\end{align*}
\]

Again, this is the same problem!
Semidefinite relaxation

Numerically, this is still *hard*:

- The Card($X$) $\leq k^2$ is still non-convex
- So is the constraint Rank($X$) = 1

but, we have made *some progress*:

- The objective Tr($AX$) is now *linear* in $X$
- The (non-convex) constraint $\|x\|_2 = 1$ became a *linear* constraint Tr($X$) = 1.

To solve this problem *efficiently*, we need to relax the two non-convex constraints above.
Semidefinite relaxation

If \( u \in \mathbb{R}^p \), \( \text{Card}(u) = q \) implies \( \|u\|_1 \leq \sqrt{q}\|u\|_2 \). Hence, we can find a convex relaxation:

- Replace \( \text{Card}(X) \leq k^2 \) by the weaker (but convex) \( 1^T|X|1 \leq k \)
- Simply drop the rank constraint

Our problem becomes now:

\[
\begin{align*}
\text{max} & \quad \text{Tr}(AX) \\
\text{subject to} & \quad \text{Tr}(X) = 1 \\
& \quad 1^T|X|1 \leq k \\
& \quad X \succeq 0,
\end{align*}
\]

This is a convex program and can be solved efficiently.
Semidefinite programming

More specifically, we get a semidefinite program in the variable $X \in S^n$, which can be solved using SEDUMI by Sturm (1999) or SDPT3 by Toh, Todd & Tutuncu (1996).

$$\begin{align*}
\text{max} & \quad \text{Tr}(AX) \\
\text{subject to} & \quad \text{Tr}(X) = 1 \\
& \quad 1^T|X|1 \leq k \\
& \quad X \succeq 0.
\end{align*}$$

- Polynomial complexity. . .

- Problem here: the program has $O(n^2)$ dense constraints on the matrix $X$ (sampling fails, . . .).

Solution here: use first order algorithm developed by Nesterov (2005).
Robustness
We look at the penalized problem:

$$\begin{align*}
\max & \quad \text{Tr}(AU) - \rho 1^T |U| 1 \\
\text{s.t.} & \quad \text{Tr} U = 1 \\
& \quad U \succeq 0
\end{align*}$$

which can be written:

$$\begin{align*}
\max_{\{\text{Tr} U = 1, \ U \succeq 0\}} \min_{\{|X_{ij}| \leq \rho\}} \text{Tr}((A + X)U)
\end{align*}$$

or also:

$$\min_{\{|x_{ij}| \leq \rho\}} \lambda_{\max}(A + X)$$

This dual has a very natural interpretation...
Dual: robust PCA

The dual problem is:

$$\min_{\{x_{ij} \leq \rho\}} \lambda_{\text{max}}(A + X)$$

- Worst-case *robust* maximum eigenvalue problem

- Componentwise noise with magnitude $\rho$ on the coefficients of the covariance matrix $A$

Asking for *sparsity* in the primal means solving a *robust* maximum eigenvalue problem with uniform noise on the coefficients.
Numerical results
Sparse factors. . .

Example:

- Use a covariance matrix from forward rates with maturity 1Y to 10Y
- Compute first factor normally (average of rates)
- Use the relaxation to get a sparse second factor
The second factor is much sparser than in the PCA case (5 nonzero components instead of 10), explained variance goes from 16% to 14%...
Portfolio hedging

- Pick a random portfolio of forward rates in JPY, USD and EUR
- Hedge it and compute the residual variance over a three months horizon
- Hedge only using the first factor
- Record the percentage reduction in variance for various levels of sparsity

(Thanks to Aslheigh Kreider for research assistance)
Portfolio hedging

Number of transactions

Reduction in variance

- US
- Japan
- Germany
Cardinality versus $k$: model

Start with a sparse vector $v = (1, 0, 1, 0, 1, 0, 1, 0, 1, 0, 1, 0)$. We then define the matrix $A$ as:

$$A = U^TU + 15vv^T$$

where $U \in S^{10}$ is a random matrix (uniform coefs in $[0, 1]$). We solve:

$$\max \quad \text{Tr}(AX)$$
subject to

$$\text{Tr}(X) = 1$$
$$1^T|X|1 \leq k$$
$$X \succeq 0,$$

- Try $k = 1, \ldots, 10$
- For each $k$, sample a 100 matrices $A$
- Plot *average solution cardinality* (and standard dev. as error bars)
Figure 1: Cardinality versus $k$. ROC curves.
Sparsity versus # iterations

Start with a sparse vector \( v = (1, 0, 1, 0, 1, 0, 1, 0, \ldots, 0) \in \mathbb{R}^{20} \). We then define the matrix \( A \) as:

\[
A = U^T U + 100 \, vv^T
\]

here \( U \in S^{20} \) is a random matrix (uniform coefs in \([0, 1]\)).

We solve:

\[
\begin{align*}
\max & \quad \text{Tr}(AU) - \rho 1^T |U| 1 \\
\text{s.t.} & \quad \text{Tr} U = 1 \\
& \quad U \succeq 0
\end{align*}
\]

for \( \rho = 5 \).
Number of iterations: 10,000 to 100,000. Computing time: 12” to 110”.
Conclusion

- *Semidefinite relaxation* for sparse PCA
- *Robustness & sparsity* at the same time (cf. dual)
- Can solve large-scale problems with first-order method by Nesterov (2005)
- (Approximately) optimal factors when fixed transaction costs are present

Slides and software available *online* at [www.princeton.edu/~aspremon](http://www.princeton.edu/~aspremon)
References


