Sparse Principal Component Analysis
Formulations And Algorithms

Thomas Rusch    Norbert Walchhofer, Department of Finance, Accounting and Statistics
WU Vienna
June 20, 2011
Outline

- Background
  - Review of Principal Component Analysis (PCA)?

- Generalized Power Method for Sparse PCA

- Problem Formulations and Reformulations
  - Single-unit sparsePCA
    - Single-unit sparsePCA via $\ell_1$-penalty
    - Single-unit sparsePCA via $\ell_0$-penalty
  - Block sparsePCA
  - Power Method

- Proposed Algorithms and their Evaluation
  - Examplary Algorithm
Principle Component Analysis - Motivation

- Method for dimension reduction
- Orthogonal transformation of possibly correlated variables into uncorrelated principal components
- Project a centered data matrix $A$ or a (sample) covariance matrix thereof $\Sigma = A^T A$ from $\mathbb{R}^p$ into $\mathbb{R}^m$ where $q \leq p$
- Aims at finding a few linear combinations the $p$ variables, pointing in orthogonal directions explaining as much variance as possible.
Principal Component Analysis - Problem

PCA - Formulation

\[
z^* = \max_{z^Tz \leq 1} z^T \Sigma z
\]

Extracting the first principal component can be done in two ways:
- computing the first eigenvector of \( \Sigma \)
- or the first right singular value of \( A \).

Usually principal components are linear combinations of all input variables with loading vector \( z^* \) (score).
PCA aims to reduce complexity, however there are some drawbacks:

- principal components depend on many variables
- interpretation of components can be agonizing
- individual loadings can be negligible
Sparse PCA simplifies mass of loadings and therefore

- highlights the most essential structures,
- is easier to interpret,
- amount of input variables can be controlled for
- and it provides a reasonable *trade-off* between *explained variance* and *usability*. 
Journée et al. (2010) provide the following contributions:

- Formulations of for single-unit sparse PCA via $\ell_1$ & cardinality ($\ell_0$)-penalty
- Formulations of for block sparse PCA via $\ell_1$ & cardinality-penalty
- Reformulations to **convex** optimization problems
- Application of the Power Method for sparse PCA
- Development of algorithms to solve the reformulated optimization problems
Single-unit optimization tries to find sparse loadings for one principal component, before calculating the next one.

Consider the following optimization problem

\[
\Phi_{\ell_1}(\gamma) \overset{\text{def}}{=} \max_{z \in B^n} \sqrt{z^T \Sigma z} - \gamma \|z\|_1
\]  

(1)

with sparsity-controlling parameter \( \gamma \geq 0 \) and sample covariance matrix \( \Sigma = A^T A \).

By setting \( \gamma = 0 \) there can be shown that \( \Phi_{\ell_1}(0) \) leads to

\[
\gamma < \|a_{i^*}\|_2,
\]  

(2)

defining the upper bound for \( \gamma \) where \( i^* \) is obtained by \( \max_i \|a_i\|_2 \).
Reformulation I

Reformulating the problem

\[
\Phi_{\ell_1}(\gamma) = \max_{z \in B^n} \|Az\|_2 - \gamma \|z\|_1
\]

\[
= \max_{z \in B^n} \max_{x \in B^n} x^T Az - \gamma \|z\|_1
\]

\[
= \max_{x \in B^n} \max_{z \in B^n} \sum_{i=1}^n z_i (a_i^T x) - \gamma \|z\|_1
\]

\[
= \max_{x \in B^n} \max_{z' \in B^n} \sum_{i=1}^n |z'_i|(|a_i^T x| - \gamma)
\]

where \(z_i = \text{sign}(a_i^T x)z'_i\).

Equation 2 proofs that there is a \(x \in B^n\) for which \(a_i^T x > \gamma\).
In view of (2), there is some $x \in \mathbb{B}^n$ for which $a_i^T x > \gamma$. By fixing $x$, solving the inner maximization problem for $z'$ we obtain a closed solution for $z^*$:

$$z_i^* = z_i^*(\gamma) = \frac{\text{sign}(a_i^T x) \left[ |a_i^T x| - \gamma \right]_+}{\sqrt{\sum_{k=1}^{n} \left[ |a_k^T x| - \gamma \right]_+^2}}, \quad i = 1, \ldots, n. \quad (5)$$
Reformulation III

Adjusting the objective function

Therefore Eq. 4 can be written as

\[ \Phi_{\ell_1}^2 (\gamma) = \max_{x \in S^p} \sum_{i=1}^{n} \left[ |a_i^T x| - \gamma \right]^2. \] (6)

This results in a differentiable and \textbf{convex} objective function, where all local and global maximal must lie in the Euclidean sphere $S^p$, \textbf{reducing the search space} of our initial problem formulation (see Eq. 8) to dimension $p$ with $p \ll n$!
What really happened...

- By introducing a vector $x$ the optimization problem is split in two, solving $x$ and $z$, respectively.
- $x$ is solved in Eq. 6 providing a sparsity pattern for $z^*$. 
- This sparsity pattern indicates which $z_i$ are active, i.e. are not 0.
- Therefore loadings only have to be calculated for $p$ of the $n$ variables of $A$ (for one component).
In contrast to the $\ell_1$-penalty (soft constraint) the $\ell_0$ or cardinality-penalty directly penalizes the number of non-zero components of vector $z$ (hard constraint).

Optimization problem formulated in d’Aspremont et al. (2008)

$$\Phi_{\ell_0}(\gamma) \overset{\text{def}}{=} \max_{z \in B^n} \sqrt{z^T \sum z} - \gamma \|z\|_0$$

(7)

Analogue to the $\ell_1$ case with derive the boundary for $\gamma$, optimization for $z_i^*$ and $x$:

$$\gamma < \|a_i^*\|_2^2,$$

$$z_i^* = z_i^*(\gamma) = \frac{\left[\text{sign}(a_i^T x)^2 - \gamma\right]_+ + a_i^T x}{\sqrt{\left[\text{sign}(a_i^T x)^2 - \gamma\right]_+ + (a_i^T x)^2}}, \quad i = 1, \ldots, n,$$

$$\Phi_{\ell_1}^2(\gamma) = \max_{x \in S^p} \sum_{i=1}^n \left[(a_i^T x)^2 - \gamma\right]_+. $$
Optimization Problem

Block optimization tries to find sparse loadings for \( m \) principal components.

Consider following generalization of Eq. 3

\[
\Phi_{\ell_1,m}(\gamma) \overset{\text{def}}{=} \max_{X \in S_m^p \ Z \in [S^n]^m} \text{Tr}(X^T A Z N) - \sum_{j=1}^{m} \gamma_j \sum_{i=1}^{n} |z_{ij}|
\]

(8)

where \( \gamma = [\gamma_1, \ldots, \gamma_m]^T \) \( \forall \gamma_j \geq 0 \) and \( N = \text{Diag}(\mu_1, \ldots, \mu_m) \) \( \forall \mu_j > 0 \).

Each \( \gamma_j \) controls the sparsity for the corresponding component. For positive \( \gamma_j \) columns of \( Z \) are not expected to be orthogonal anymore! Note that distinct values of \( \mu_j \) ensure the columns of \( X^* \) being the dominant \( m \) components, while also pursuing more sparse and orthogonal vectors.
Since the columns of $Z$ are decoupled the reformulation can be done analogue to the single-unit case. Hence, for every column of $X$ every row element is optimzed, indicating the ’active-status’ for each component of $Z$ (i.e. variable of A) of each row $Z$.

If $\mu_i |a_i^T x_j^*| > \gamma_j$ is fullfilled $z_{ij}^*$ is active.
The power method is an eigenvalue algorithm, given a matrix $A$ trying to find the dominant eigenvalue $\lambda$ and its corresponding eigenvector $v$ such that $Av = \lambda v$. By avoiding a matrix decomposition, it is very favorable for large sparse matrices since the computation afforded is very low. The scalar $q = x^T x$ converges linearly against the dominant eigenvalue.

$$x_{k+1} = \frac{Ax_k}{\|Ax_k\|}$$

$x_0$ can be an approximation or a random vector. The method works under following assumptions:

- $A$ has an eigenvalue strictly greater than others
- Starting vector $x_0$ has a non-zero component in the direction of the eigenvector of the dominant eigenvalue.
Generalization of the Power Method

Based on gradient method for maximizing convex functions the authors show that a convex function

\[ f^* = \max_{x \in Q} f(x) \]  

(10)
can iteratively maximized by a subgradient, even if that \( f(x) \) is not assumed to be differentiable.

In our case we have to solve a quadratic objective function \( f(x) = \frac{1}{2} x^T C x \) for \( C \in S^{p}_{++} \), which can be solved by

\[ x_{k+1} = \frac{C x_k}{\|C x_k\|}, \quad k \geq 0. \]  

(11)
Algorithm 4: Block sparse PCA algorithm based on the $\ell_1$-penalty (16)

input : Data matrix $A \in \mathbb{R}^{p \times n}$
Sparsity-controlling vector $[\gamma_1, \ldots, \gamma_m]^T \geq 0$
Parameters $\mu_1, \ldots, \mu_m > 0$
Initial iterate $X \in S_m^p$

output: A locally optimal sparsity pattern $P$

begin
  repeat
    for $j = 1, \ldots, m$ do
      $x_j \leftarrow \sum_{i=1}^n \mu_j [\mu_j |a_i^T x_i| - \gamma_j] + \text{sign}(a_i^T x) a_i$
      $X \leftarrow \text{Polar}(X)$
    until a stopping criterion is satisfied
  Construct matrix $P \in \{0, 1\}^{n \times m}$ such that
  \[
  p_{ij} = \begin{cases} 
  1 & \text{if } \mu_j |a_i^T x_j| > \gamma_j \\
  0 & \text{otherwise.}
  \end{cases}
  \]
end
Algorithms Evaluated

- All four GPower algorithms, two single-unit and two block sparse PCA each with $\ell_0$ and $\ell_1$ penalty
- Greedy search algorithm of d’Aspremont et al. (2008) (non-convex)
- SPCA from Zhou et al. (2006) (lasso penalty)
- $rSVD_{\ell_0}$ and $rSVD_{\ell_1}$ by Shen and Huang (2008)
Evaluation on Real Data - Explained Variance

![Graph showing explained variance versus cardinality for two groups of algorithms. The red line represents Group 1: All 4 GPower codes and rSVD_{\ell_0}, while the black dashed line represents Group 2: SPCA, rSVD_{\ell_1}. The x-axis represents cardinality in units of 10^4, and the y-axis represents the proportion of explained variance.]}
Evaluation on Real Data - Computation Time I

Proposed Algorithms and their Evaluation

![Graph showing computational time versus cardinality](image-url)
<table>
<thead>
<tr>
<th>$p \times n$</th>
<th>50 $\times$ 500</th>
<th>100 $\times$ 1000</th>
<th>250 $\times$ 2500</th>
<th>500 $\times$ 5000</th>
<th>750 $\times$ 7500</th>
</tr>
</thead>
<tbody>
<tr>
<td>GPower$_{1}$</td>
<td>0.22</td>
<td>0.56</td>
<td>4.62</td>
<td>12.6</td>
<td>20.4</td>
</tr>
<tr>
<td>GPower$_{0}$</td>
<td>0.06</td>
<td>0.17</td>
<td>2.15</td>
<td>6.16</td>
<td>10.3</td>
</tr>
<tr>
<td>GPower$_{1,m}$</td>
<td>0.09</td>
<td>0.28</td>
<td>3.50</td>
<td>12.4</td>
<td>23.0</td>
</tr>
<tr>
<td>GPower$_{0,m}$</td>
<td>0.05</td>
<td>0.14</td>
<td>2.39</td>
<td>7.7</td>
<td>12.4</td>
</tr>
<tr>
<td>SPCA</td>
<td>0.61</td>
<td>1.47</td>
<td>13.4</td>
<td>48.3</td>
<td>113.3</td>
</tr>
<tr>
<td>rSVD$_{1}$</td>
<td>0.29</td>
<td>1.12</td>
<td>7.72</td>
<td>22.6</td>
<td>46.1</td>
</tr>
<tr>
<td>rSVD$_{0}$</td>
<td>0.28</td>
<td>1.03</td>
<td>7.21</td>
<td>20.7</td>
<td>41.2</td>
</tr>
</tbody>
</table>

Table 8: Average computational time for the extraction of $m = 5$ components (in seconds).
GPower algorithms show competitive behavior in terms of

- explained variance
- computation time
- control for sparsity pattern
- usability (data matrix and sample covariance matrix)


Thank you for your attention

Thomas Rusch
Norbert Walchhofer
WU Wirtschaftsuniversität Wien
Augasse 2–6, A-1090 Wien