Fast algorithms for sparse principal component analysis based on Rayleigh quotient iteration

Volodymyr Kuleshov
Department of Computer Science, Stanford University

Highlights

- New algorithms for sparse PCA that
  - Require $O(k^3 + nk)$ flops/step, for a sparsity of $k$;
  - In practice, use 10-100x fewer flops than current state-of-the-art methods;
  - Produce eigenvectors that are as good or better than ones from existing algorithms.
- Generalize Rayleigh quotient iteration;

Sparse PCA

And instance of sparse PCA is defined as

$$\max_{x \in \mathbb{R}^n} \frac{1}{2} x^T \Sigma x$$ \quad \text{s.t.} \quad ||x||_1 \leq k\tag{1}$$

for $\Sigma \in \mathbb{R}^{n \times n}$, $\Sigma = \Sigma^T$ and $k > 0$.

Current state-of-the-art

Most popular methods are variations of the generalized power method: Zou et al. (2006), Witten et al., (2009), Journee et al. (2010).

Algorithm 1 GPower0/GPower1($D$, $x_0$, $\gamma$, $\epsilon$)

1. $j \leftarrow 0$
2. repeat
   1. $y \leftarrow S(D^T x(j), \gamma) ||S(D^T x(j), \gamma)||_2$
   2. $x(j) \leftarrow D y ||D y||_2$
   3. $j \leftarrow j + 1$
   until $||x(j) - x(j-1)|| < \epsilon$
   return $S(D^T x(j), \gamma) ||S(D^T x(j), \gamma)||_2$

where $S : \mathbb{R}^n \times \mathbb{R} \rightarrow \mathbb{R}^n$ is

$$S(a, \gamma) : = \begin{cases} a \cdot |a| & \text{for GPower0} \\ a \cdot |a| - \gamma & \text{for GPower1} \end{cases}$$

GPower outperforms other sparse PCA algorithms (Journee et al., 2010), including:
- SDP formulations (D’Aspremont et al., 2007);
- Greedy search (Moghaddam et al., 2006).

Generalized Rayleigh quotient iteration

The generalized power method is based on two rather unsophisticated algorithms: the power method for computing eigenvalues and gradient ascent. We introduce an algorithm that extends the state-of-the-art Rayleigh quotient iteration algorithm and that can be interpreted as a form of Newton’s method.

**Comparison to GPower**

Basically, same results using 10-100x fewer flops:

- GPower2
- GPower5
- GRQI

Applications of the sparse SVD

Finding leading sparse principal component for a gene expression dataset using up to 10x fewer flops.

Sparse SVD

Sparse SVD generalizes objective (1) as follows.

$$\max_{x \in \mathbb{R}^n} x^T R v$$

s.t. $||x||_1 \leq 1$, $||v||_1 \leq k$,

Problem (2) reduces to problem (1) using

$$u^T R v = \frac{1}{2} \left( v^T \left( \begin{array}{c} 0 \\ R \end{array} \right) \right) u$$

Sparse SVD also uncovers Gabor filters from a matrix of image patches taken from the STL-10 dataset.

Pseudocode

Algorithm 2 GRQI($\Sigma$, $x_0$, $k$, $J$, $\epsilon$)

1. $j \leftarrow 0$
2. repeat
   1. $\mu \leftarrow (x(j))^T \Sigma x(j)/(x(j))^T x(j)$
   2. $W \leftarrow \{i | x(i) \neq 0\}$
   3. $x(j) \leftarrow S(\Sigma W - \mu I)^{-1} x(j) / \text{RQI Update on } W$
   4. $x_{\text{new}} \leftarrow x(j)/||x(j)||_2$
   if $j < J$
      $x_{\text{new}} \leftarrow x_{\text{new}} / ||x_{\text{new}}||_2$ // Power met. update
      $x(j) \leftarrow \text{Project}_{k}(x_{\text{new}})$
      $j \leftarrow j + 1$
   until $||x(j) - x(j-1)|| < \epsilon$
   return $x(j)$

At every iteration, Algorithm 2 updates all non-zero indices using Rayleigh quotient iteration; for the first $J$ iterations, it also performs a Power method step. It projects iterates on the $l_0$ ball.

When $J < \infty$, the iterates $(x(j))_{j=1}^{\infty}$ of Algorithm 2 converge to a limit $x^*$ at a cubic rate: $||x_{j+1} - x^*|| = O(||x_j - x^*||^3)$.