Shrinking the Sparse PCA graph search problem

Daniel Ringwalt

December 17, 2022

1 Background

PCA is understood to produce the eigenvector having the largest eigenvalue in a positive definite matrix (the covariance matrix Σ). The eigenvector is written as:

$$\max_v v^T \Sigma v \text{ such that } ||v||_2 = 1$$

k-sparse PCA makes a cut to the system (applying a rank-k projection matrix using k standard basis vectors), such that $||v||_0 = k$. Selecting k standard basis vectors is equivalent to selecting a subset of k variables, out of n variables.

Consider increasing k by one (including another row and column in Σ after projection). In general, all eigenvalues, all eigenvectors, and the new data to be added all influence the new spectrum, but the previous state can be used to compute the new state (rank-one update; the original matrix will be diagonalized, and the added data will undergo a change-of-basis). We believe that eagerly computing the diagonalization, each time that we increment k, will lead to novel tactics for Sparse PCA. However, if new tactics are not used based on the properties of the subsystem, then it is better not to solve the data SVD/covariance eigenproblem incrementally, waiting until all k variables are chosen.

Here, we will propose one application of the updated diagonalization, for making a cut to the system by projecting onto the current principal component. Simpler tactics could be possible.

2 Tree search Sparse PCA

Berk et. al described a tree search for the Sparse PCA problem. Each node of the tree contains a lower bound and upper bound on each entry of the principal component. At the root, the lower bound at the node is $\mathcal{L} = \mathbf{0}$ and the upper bound is $\mathcal{U} = \mathbf{1}$. Child nodes are found by disallowing exactly one variable where $\mathcal{L}_i \neq \mathcal{U}_i$ (updating $\mathcal{U}_i$ from 1 to 0), or committing to that variable's inclusion (updating $\mathcal{L}_i$ from 0 to 1). The root has an empty support for $\mathcal{L}$. Denote the

support of $\mathcal{L}$ as i. This represents a subsystem which we can update using a rank-one update for a child node, if $\mathcal{L}$ has changed.

Time complexity is expected to be average-case $O\left(\binom{n}{k}\right)$. Berk et. al's branch-and-bound of the tree ought to perform better on real data sets suited to PCA, where there should already be some k-variable subset of the system where all variables have good correlation with each other.

2.1 Lower bounds

The global lower bound on any tree node is stored, for evaluating each node which will be explored. This lower bound could be continuously updated using any stochastic Sparse PCA or sparse recovery algorithm. In particular, there is not a data dependency between this lower bound exploration and the upper bound exploration.

2.2 Upper bounds

We will move on to upper bounds, which require analysis to prove upper bound. Performance depends on skipping nodes where the node upper bound is lower than the global lower bound. Positive definite eigenvalues are bounded by the matrix trace, but without an improved upper bound, the runtime of the system would degrade to $O(\left(\binom{n}{k}\right))$. After taking a projection of Σ, identities such as the Gershgorin circle theorem can be applied to bound all eigenvalues.

3 Cut system using first i variables

In our proposal, we would already solve a subsystem incrementally having $k_{\text{subsystem}} = i$. It would be nice to make $k - i$ more updates to the diagonalization and use this to upper-bound the solution to our problem. However, we need an upper-bound after all possible $\binom{n-i}{k-i}$ sequences of updates.

4 Cut system along incremental principal component

After making i updates to the system, we are left with a subsystem where the remaining variables still have high correlation with one another.

5 Recursive Sparse PCA problem

We can apply our approximate Sparse PCA solver recursively, after reducing k.

6 Strengthening subproblem using rank-one update