Bounding tree-based Sparse PCA using perturbation theory

Daniel Ringwalt

December 4, 2022

1 Background

Berk et. al [1] created a modular, best-first search design for combinatorial Sparse PCA. Nodes in the search tree have a lower bound (generally stochastic method for a k-sparse eigenvalue approximation), as well as an upper bound (which is more challenging to bound).

Online singular value decomposition [2] is seen as a time-saving optimization, but constructing a k-subset SVD via column updates may not affect our big-O time complexity for the dimensions found here. The outer product is taken of the new column of data, and results in a symmetric rank-one matrix being added to the $k-1$-subset covariance matrix (logically including a zero row and column in the original subset matrix). We propose that once the Sparse PCA algorithm commits to including a variable (in the current tree being searched), then it should eagerly update the online SVD. This solved sub-problem should produce a variety of useful findings.

In a graph search, we will already have i variables committed to a new covariance computation, and there are still $\binom{n-i}{k-i}$ possible results. Upper bounding the k-sparse SVD solution is still challenging. Current bounds (using trace and Gershgorin circle on all possible subsets) avoid any decomposition or partial solution of the problem. We suggest that the diagonalization of the subproblem (i x i covariance matrix) is the best starting point for a more intensive search for an upper bound. There ought to be a less unwieldy, and better performing, bound on this partially decomposed problem, than the approach of successive updates shown here.

2 Sparse PCA for dense random data

As input, we take a fully determined n x n covariance matrix (rank is actually $n-1$ due to the loss of one degree of freedom from centering the data). Later, covariance values can be computed on-the-fly in a separate code path, for datasets which are rank-deficient (few observations) [1]. Our covariance will usually be unit-scaled so that each variable has a sample variance of 1.

The centered data matrix $D : m$ x n will only be considered theoretically. The covariance matrix $\Sigma : n$ x n contains entries $\Sigma_{ij} = \frac{\langle d_i, d_j \rangle}{n-1}$.

$$\Sigma = \frac{1}{n-1} D^T D$$

3 Rank-one updates

Several treatments of sparse PCA/sparse recovery have appended rank-one or low-rank data to a subset of the data matrix, and desired an online SVD. However, fast perturbations of the subsetted covariance matrix are still a novel algorithmic detail.

Consider a subset of $k-1$ variables, and an existing positive definite diagonalized matrix $\Sigma^{k-1} = V^{k-1}\Lambda^{k-1}\left(V^{k-1}\right)^T$. We should treat the matrices as having an additional kth row and column (zeros in the covariance matrix and diagonal matrix, and the basis vector e_k in V).

Next, consider appending the variable to the subset, which will go at location k. For the matrix $\frac{1}{n-1}DD^T$, we have a rank-one update $\frac{1}{n-1}DD^T + \frac{1}{n-1}d_k d_k^T$. This matrix has the same spectrum as the updated covariance matrix Σ^k.

Using the singular vectors of D, we can produce another matrix which has the same spectrum: $\Lambda^{k-1} + \frac{1}{n-1}\left(V^{k-1}\right)^T \sigma_k'(\left(V^{k-1}\right)^T \sigma_k')^T$, where σ_k' is a newly constructed column vector and the first $k-1$ elements come from the kth column of Σ, taken for the $k-1$ subset of covariates seen so far. $(\sigma_k')_k$ is constructed to preserve the trace:

$$\mathrm{Tr}\left(\frac{1}{n-1}DD^T + \frac{1}{n-1}d_k d_k^T\right) = \mathrm{Tr}\left(\Lambda^{k-1} + \frac{1}{n-1}\left(V^{k-1}\right)^T \sigma_k'(\left(V^{k-1}\right)^T \sigma_k')^T\right)$$

Consider the covariance matrix Σ of the subsets of variables Σ^{k-1}, Σ^k. The change in trace is due to inserting a new variance on the diagonal: Σ_{kk}.

$$\mathrm{Tr}\left(\frac{1}{n-1}\sigma_k'\left(\sigma_k'\right)^T\right) = \frac{1}{n-1}\sum_{i=1}^{k}\left(\sigma_k'\right)^2 = \Sigma_{kk}$$

$$(\sigma_k')_k = \sqrt{\Sigma_{kk} - \sum_{i=1}^{k-1}(\sigma_k')_i^2}$$

3.1 Solving the rank-one update

New eigenvalues are found analytically, by expanding the characteristic polynomial. The characteristic polynomial after the rank-one update is divided through, to produce a sum of k rational functions which have no zero, and one pole at exactly one original eigenvalue. After the sum, we have k zeroes, and every zero is separated from the next zero by one of the k poles. The updated

eigenvalues (at the zeroes) are at least slightly perturbed from the original locations. The equations are solved by a root-finding approximation, but quickly converging up to machine epsilon, using the LAPACK `slaed9` helper function.

For the eigenvalue numbered i, the updated eigenvector is:

$$x_i v_i = V^{k-1}(\Lambda^{k-1} - I\left(\Lambda^k\right)_{ii})^{-1}\sigma'_k$$

This is divided through by a scale factor x_i determinied by taking the norm.

This formula performs stably for infinitesimal updates. The inverted diagonal matrix is ill-conditioned due to the large diagonal entry at i, so the result would be infinitesimally close to extracting one original eigenvector using e_i.

This technique is under-utilized so far, for accelerating this combinatorial linear algebra problem. Generally, combinatorial sparse PCA has not moved from research into practicum (productionizing and optimizing a particular algorithm). Time and space complexity is dominated by tracking and multiplying through a k x k eigenvector matrix, and big-O performance of each step in a combinatorial procedure has not changed.

4 Bounding the rank-one update

Using bounds on the dominant eigenvalue(s) (absolute value upper and lower), and eigenvector(s) (maximum disturbance θ), we can explore the problem space iteratively in polynomial time instead of factorial time. We need the two most dominant eigenvalues (largest: $\lambda_1^{k-1}, v_1^{k-1}$, second largest: $\lambda_2^{k-1}, v_2^{k-1}$).

At this point, the σ' update vector needs to be multiplied on the left by $\left(V^{k-1}\right)^T$, producing updates w to be added to the diagonal matrix. The enhanced bounds on λ^k [3] are the bounds which yield closed intervals:

$$\lambda_1^{k-1} < \lambda_1^k \leq \lambda_1^{k-1} + u_1(w_1, \Sigma_{kk}, \lambda_1^{k-1} - \lambda_2^{k-1})$$

$$\lambda_2^{k-1} + l_2(w_1, w_2, \lambda_1^{k-1} - \lambda_2^{k-1}) \leq \lambda_2^k < \lambda_1^{k-1}$$

$$\lambda_2^k \leq \lambda_2^{k-1} + u_2(w_1, w_2, \Sigma_{kk}, \lambda_2^{k-1} - \lambda_3^{k-1})$$

Zhu's functions will be included as a supplement. We bound u_2 using any possible gap $\lambda_2^{k-1} - \lambda_3^{k-1}$, and we find that u_2 varies as a quadratic, with one extreme point. Therefore, we need to evaluate u_2 at 2 or 3 possible values of the eigenvalue difference (minimum possible, maximum possible, and an extreme point). However, this shows that we do not need to track bounds on all eigenvalues, in order to effectively bound the dominant eigenvalue(s) after several updates.

4.1 Eigenvector disturbance

Calculate the perturbed eigenvectors within our diagonal system of $k-1$ variables: $x_i v_i = (\Lambda^{k-1} - I(\Lambda^k)_{ii})^{-1} w$

For no disturbance, we would have $v = e_1$ when considering eigenvalue λ_1^k. The contribution of other components is given by $|\sin\theta|$, where $\cos\theta = |v_1|$.

We would maximize $\sin\theta$ by considering $\lambda_1^k = \lambda_1^{k-1} + u_1$ (perturb the eigenvalue as far as possible). Within the same formula, we want to consider a smaller distance from λ_i^{k-1} to λ_1^k before taking the reciprocal (those eigenvalues have a larger contribution to rotating the new eigenvector). This provides an upper bound on $|v_1|$ for any possible update vector.

Then, we can retain the basis that we have been using, from V^{k-1} and appending basis vector e_k. Using the system Σ^k (with uncertainty present), we may rotate updates in σ'_{k+1} into the first component. This gives us a minimum and maximum w_1. Furthermore, the u_i function may vary as a qudratic with w_i, so we will consider a possible extrema point, or either bound on w_1 may produce the upper bound u_1.

5 Sparse PCA upper bounding algorithm

We presented a first iteration, going from an i-subset to an $i+1$-subset. We would like to get to a k-subset, in not too many steps (or this decomposition will perform worse than much cheaper upper bounds). PCA is sensitive to the scales or duplication of variables, so repeated applications would greedily choose one new variable from the available set, and keep choosing it over and over again. We should remove the variable which was used to maximize u_1 from consideration for u_1 again. This is starting to look like a greedy algorithm that could be used to select all k variables. However, a different variable might have produced the most extreme bounds for l_2, u_2.

The algorithm is not greedy, because we keep considering the entire search space when growing l_2, u_2. Therefore, we don't constrain λ_2 incorrectly, and behavior will arise where λ_2 influences our λ_1 (the updates could have the sophistication that we expect from eigenvalue updates, with very loose terms on the possible updates).

6 Limitations

Our λ_2 bounds could grow more quickly than our λ_1 bounds, and when our formula suggests a lower bound on λ_1 and upper bound on λ_2 which intersect, then the validity would probably break down. Without an adequate depth in our search tree, we might terminate without a bound because our updates are larger than the size of the subproblem (e.g. $i = 3, k = 10$). Therefore, we might only apply this bound when $\frac{i}{k} >> \frac{1}{2}$. The procedure is particularly useful, and should always be used, for evaluating a new node at $i = k - 1$. It is cheaper

than diagonalizing $n-k$ matrices (switching over to brute-force at the last level of the tree).

Spectral theory lower bounding the possible upper bound would be a useful finding. We could show that branch-and-bound sparse PCA will never constrain its upper bounds beyond a certain level of performance.

References

[1] Berk, Lauren, & Dimitris Bertsimas. "Certifiably optimal sparse principal component analysis." Mathematical Programming Computation 11.3 (2019): 381-420.

[2] Bunch, James R., & Christopher P. Nielsen. "Updating the singular value decomposition." Numerische Mathematik 31.2 (1978): 111-129.

[3] Zhu, L., Peng, X. & Liu, H. Rank-one perturbation bounds for singular values of arbitrary matrices. J Inequal Appl 2019, 138 (2019).