Incremental Methods for Computing Extreme Singular Subspaces

Christopher G. Baker[1]
Kyle A. Gallivan[2]
Paul Van Dooren[3]

[1]Computational Engineering & Energy Sciences, Oak Ridge National Laboratory, USA
[2]Department of Mathematics, Florida State University, USA
[3]CESAME, Universitè catholique de Louvain, Belgium

2012 SIAM Conference on Linear Algebra
Valencia, Spain

June 18, 2011

Dominant SVD

Definition

Singular Value Decomposition

The singular value decomposition of an $m \times n$ matrix A is

$$A = U \begin{bmatrix} \Sigma \\ 0 \end{bmatrix} V^T$$

with orthogonal U and V, Σ diagonal with non-negative entries.

Dominant SVD

The dominant SVD refers to the vectors of U and V corresponding the largest singular values. It has use in numerous applications:

- model reduction
- data compression
- statistics

This is largely due to its optimality in approximating A.

Dominant SVD

Computation

Computing the Dominant SVD

This can be done by:

- computing the full SVD and truncating (dense)
- computing the dominant eigenvectors of $A^T A$, AA^T or $[0, A^T; A, 0]$
- non-linear attacks on $f(U, V, \Sigma) = AV - U\Sigma = 0$
- low-rank incremental SVD methods

Incremental/Updating SVD Approach

- Basic Idea: given $B = U\Sigma V^T$ and B_+, compute the SVD of $\begin{bmatrix} B & B_+ \end{bmatrix}$.
- Do this for all columns of a matrix A and you get the SVD of A.
- But it costs more than the direct SVD of A. So why do it that way?
  - you need an online calculation, and that's how the data arrives
  - ...

Dominant SVD

Computation

Computing the Dominant SVD

This can be done by:

- computing the full SVD and truncating (dense)
- computing the dominant eigenvectors of $A^T A$, AA^T or $[0, A^T; A, 0]$
- non-linear attacks on $f(U, V, \Sigma) = AV - U\Sigma = 0$
- low-rank incremental SVD methods

Incremental/Updating SVD Approach

- Basic Idea: given $B = U\Sigma V^T$ and B_+, compute the SVD of $\begin{bmatrix} B & B_+ \end{bmatrix}$.
- Do this for all columns of a matrix A and you get the SVD of A.
- But it costs more than the direct SVD of A. So why do it that way?
  - you need an online calculation, and that's how the data arrives
  - ...

Dominant SVD

Low-Rank Incremental Computation

A low-rank approximation

- Relax the previous incremental approach.

1) Take a rank-k approximation $B \approx U\Sigma V^T$ and new vectors B_+
2) Update the SVD of $\begin{bmatrix} U\Sigma V^T & B_+ \end{bmatrix}$
3) Keep only the rank-k dominant part: $U_+\Sigma_+V_+^T$

- Result is $U_+\Sigma_+V_+^T \approx \begin{bmatrix} U\Sigma V^T & B_+ \end{bmatrix} \approx\approx \begin{bmatrix} B & B_+ \end{bmatrix}$

A Low-Rank Incremental SVD Method

Input matrix A.

0) Initial rank-k factorization $U\Sigma V^T$ from the first few columns of A
1) For new columns A_+ from A, compute SVD of $\begin{bmatrix} U\Sigma V^T & A_+ \end{bmatrix}$
2) Keep the dominant part, truncate $U\Sigma V^T$ back to rank-k
3) If more columns in A, goto 1.

Dominant SVD

Low-Rank Incremental Computation

A low-rank approximation

- Relax the previous incremental approach.

1) Take a rank-k approximation $B \approx U\Sigma V^T$ and new vectors B_+
2) Update the SVD of $\begin{bmatrix} U\Sigma V^T & B_+ \end{bmatrix}$
3) Keep only the rank-k dominant part: $U_+\Sigma_+V_+^T$

- Result is $U_+\Sigma_+V_+^T \approx \begin{bmatrix} U\Sigma V^T & B_+ \end{bmatrix} \approx\approx \begin{bmatrix} B & B_+ \end{bmatrix}$

A Low-Rank Incremental SVD Method

Input matrix A.

0) Initial rank-k factorization $U\Sigma V^T$ from the first few columns of A
1) For new columns A_+ from A, compute SVD of $\begin{bmatrix} U\Sigma V^T & A_+ \end{bmatrix}$
2) Keep the dominant part, truncate $U\Sigma V^T$ back to rank-k
3) If more columns in A, goto 1.

References

Numerous Independent Descriptions

- **B. S. Manjunath, S. Chandrasekaran and Y. F. Wang**. An eigenspace update algorithm for image analysis, 1995.
- **S. Chandrasekaran, B. S. Manjunath, Y. F. Wang, J. Winkeler and H. Zhang**. An eigenspace update algorithm for image analysis, 1997.
- **A. Levy and M. Lindenbaum**. Sequential Karhunen-Loeve basis extraction and its application to images, 2000.
- **Y. Chahlaoui, K. Gallivan and P. Van Dooren**. An incremental method for computing dominant singular spaces, 2001.
- **M. Brand**. Incremental singular value decomposition of uncertain data with missing values, 2002.
- Y. Chahlaoui, K. Gallivan and P. Van Dooren. Recursive calculation of dominant singular subspaces, 2003.
- C. G. Baker. A block incremental algorithm for computing dominant singular subspaces, 2004.
- M. Brand. Fast low-rank modifications of the thin singular value decomposition, 2006.
- C. G. Baker, K. A. Gallivan, P. Van Dooren. Low-Rank Incremental Methods for Computing Dominant Singular Subspaces. 2012.

Low-Rank Incremental SVD

Algorithmic Motivation

Benefits

- Reduced cost: rank-k IncSVD of $m \times n$ matrix in $O(mnk)$ flops.
- Reduced storage: $O(mk + nk)$, compared to $O(mn)$ for full SVD+trunc.
- Pass efficient: streaming access to A, for online analysis, distant storage
- The algorithm is rich in BLAS3 routines.
- Can be used to compute subordinate (smallest) SVD as well.

Downside

- The efficiency comes from truncating data, maintaining low rank.
- But truncated data introduces errors.
- The resulting factorization only approximates the dominant SVD.
- We would like to know:
  1. how well does it work?
  2. what exactly is it doing?

Low-Rank Incremental SVD

What is it doing?

An Optimization Explanation [BGVD12]

- Consider the IncSVD of the matrix AD, for some orthogonal D:

$$D = \begin{bmatrix} D_1 & D_2 & \cdots & D_b \end{bmatrix}$$

- At each step j, the method is shown to select V_j that optimizes

$$\mathrm{RQ}(Y) = \mathrm{trace}\left(Y^T A^T A Y\right),$$

  for $Y \in \mathrm{span}(\begin{bmatrix} V_{j-1} & D_j \end{bmatrix})$

- For "standard" $D = I$, this is a sweep over the coordinate axes:

$$D_j = \begin{bmatrix} 0 & \cdots & I & \cdots & 0 \end{bmatrix}^T$$

- What else can we do with D?

Restarting the IncSVD

A Multi-pass Method

Restarting with D

- Choosing $D = \begin{bmatrix} V & \cdots \end{bmatrix}$ allows the procedure to be restarted.
- Representing $D = I + WY^T$ (rank-k update) maintains pass efficiency.

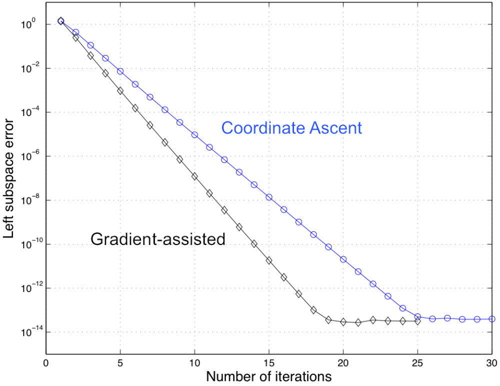
Accelerating with D

- Additionally, gradient information

$$\nabla RQ(Y) = A^T AV$$

can be injected into D to speed convergence.

- Limited information can be efficiently injected into D in this way.



Convergence Properties

Does it work?

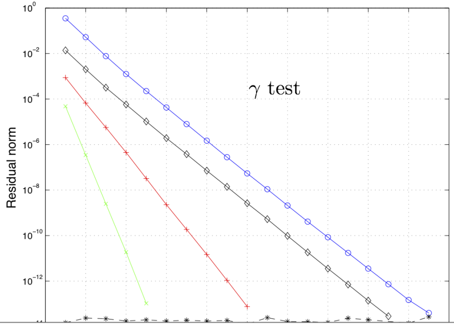
Provable Convergence [BGVD12]

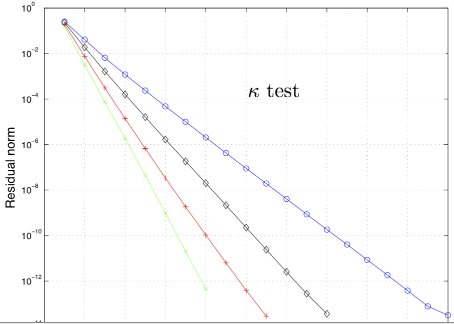
a) Global convergence, with stable convergence only to dominant subspaces.

b) Linear convergence, with a rate $c = \gamma/(\kappa^2 - 1)$, where:

- γ concerns the subspace information of truncated data
- $\kappa = \sigma_k/\sigma_{k+1}$ is the gap between dominant and dominated

- These are expected of an ascent method.





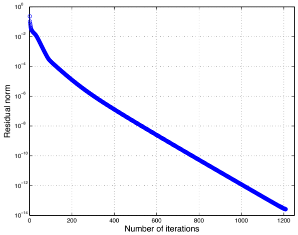
Best and Worse Case Performance

Best Case

- For certain classes of matrices, a single pass will perfectly compute the dominant singular values and subspaces.
- This occurs when $\sigma_{k+1} = \sigma_{k+2} = \cdots = \sigma_n$.
- Analogous result holds for smallest singular values with $\sigma_1 = \cdots = \sigma_{n-k}$
- Consequences:
  - $O(mnk)$ rank-$k+1$ IncSVD is capable of identifying all $\sigma \in \sigma(A)$
  - *???*

Worst Case

- The worst case performance seems to correspond to $\sigma_k = \sigma_{k+1}$, no gap.
- Current analysis doesn't apply.
- Convergence still seems to occur, albeit very slowly.



Extensions/Future Work

Extensions

- Other factorizations:
  - tensor SVD/higher-order SVD [O'Hara 2010]
  - CX factorization, where C samples columns of A (data-driven apps.)
  - symmetry-preserving SVD [Shah, Sorensen 2006], other structured SVDs
- Sparsification procedures:
  - If A is sparse and the factorization is sparsified, sub-linear $O(\alpha nk)$ work
  - See [O'Hara 2010]

Future Work

- Global convergence is nice, but fast convergence is nice, too.
- Would like tighter bounds on single pass error.
- Need good stopping criteria for multi-pass method.
- Wanted: application.