Evaluating and improving streaming methods for large scale SVD problems

Andreas Stathopoulos
Computer Science
William & Mary

Jeremy M. Myers
Sandia National Lab

Toon Tran
William & Mary

Low-rank Approximation of Streaming Matrices

1. Traditional "batch" SVD solvers:
   - Dense & iterative solvers (e.g., LAPACK, Krylov, randomized SVD)
   - Require storage to full matrix
2. Challenges:
   - Matrix seen in online fashion (streaming)
   - Matrix can be read (or generated) in sections once

Matrix Streaming

Matrix $A \in \mathbb{R}^{m\times n}$ streamed as:

$$A = A^{(1)} + A^{(2)} + A^{(3)} + \dots$$

Goal

incrementally approximate low-rank space of

$$A^{(1)} + A^{(2)} + \dots + A^{(i)}, \;\; i = 1, 2, \dots$$

Sketch $B = \tilde{U}\tilde{\Sigma}\tilde{V}^T$ not exact singular space unless stream is stored

Approach may depend on quality metric

- $\|A - B\|$
- $\|A^T A - B^T B\|$
- $\|A\tilde{V} - \tilde{U}\tilde{\Sigma}\|$

Approaches we study

- Incremental SVD/Frequent Directions:
  - $A^{(i)}$ is a window of rows/columns streamed in some order
  - Use $B^{(i-1)}$ and $A^{(i)}$ to form $B^{(i+1)}, i = 0, 1, \ldots$
- Randomized Sketching:
  - $A^{(i)}$ arbitrary pattern/order
  - Form LRA after all A is streamed or at any point

This talk:

- Large window iSVD using iterative solver and initial guesses
- Comparison with randomized sketching
  - complexity
  - execution time for large problems
  - accuracy of space
- iSVD optimizations for symmetric problems
  - Sampling-based accuracy evaluation
  - improving eigenvalue / eigenvector approximations

Incremental SVD (iSVD)   Brand (2002), Baker et al. (2012)

Tracks the rank-r largest singular values and right singular vectors

$$C \leftarrow \begin{bmatrix} \Sigma^{(i-1)} V^{(i-1)T} \\ A^{(i)} \end{bmatrix}$$

$$(U, \Sigma, V) \leftarrow \mathrm{SVD}(C)$$

$$V^{(i)} \leftarrow V_{:,1:r},$$

$$\Sigma^{(i)} \leftarrow \Sigma_{1:r,1:r}$$

Frequent Directions = iSVD with soft thresholding truncation of Σ, Ghashami et al. (2015)

iSVD Using Iterative Methods

Extension 1

Compute $(U, \Sigma, V) \leftarrow \text{svds}(C)$ using iterative solver

Rationale:

- Error bounds improve with large window size, Ghashami et al. (2015)
- reduces the likelihood of adversarial cases
- Larger windows minimizes overhead over batch method
- When no information drift, use initial guesses for left singular vectors:
  $$\begin{bmatrix} I \\ A^{(i)} V_r^{(i-1)} (\Sigma_r^{(i)})^{-1} \end{bmatrix}$$

Big-O Cost of iSVD

Let $A \in \mathbb{R}^{m \times n}$ be dense:

$$\text{Cost} = \mathcal{O}\left(\underbrace{tM + t^2 m}_{\text{batch cost on } A} + \underbrace{\frac{tmr}{\ell}(n + \ell + t + r)}_{\text{sketching cost}} \right)$$

- t: average number of Matvecs per window
- M: cost of Matvec ($O(mn)$ for dense, $O(cm)$ for sparse)
- r: target rank
- ℓ: window size ($m/\ell = \#\text{windows}$)

take away:

Relative additional cost of iSVD over batch solver decreases as

1. density of A increases
2. more iterations are needed
3. window size increases

SketchySVD

Upadhyay (2016), Tropp et al. (2019)

Sketch $A \in \mathbb{R}^{m \times n}$ with left, right and core matrices (X, Y, Z):

$$X \leftarrow \eta X + \nu Y A$$
$$Y \leftarrow \eta Y + \nu A \Omega^*$$
$$Z \leftarrow \eta Z + \nu \Phi A \Psi^*$$

Y, Ω, Φ, Ψ: **random maps**
Rank $r \leq$ range size(X, Y) $\leq$ core size $\leq \min(m, n)$

At the end (or at any point) of streaming use (X, Y, Z) to compute LRA

Updates: expensive   LRA: inexpensive

Cost of SketchySVD

With $k = 4r, s = 8r$, complexity:

$$\mathcal{O}\Big(\underbrace{16r^2(m+n)}_{QR} + \underbrace{512r^3}_{LeastSq.} + \underbrace{64r^3}_{SVD} + \underbrace{\sum_{i=1}^{6} M_i}_{\text{matrix multiplies}} \Big)$$

M_i depend on structure of A and choice of sketch maps

Random Map $\Xi \in \mathbb{R}^{d\times n}$

- Gaussian:
  - Dense but efficient through GEMM
  - Requires storing $\mathcal{O}(2d(n+m))$ entries
- Sparse sign (count sketch):
  - 4 nonzeros per column, each entry is ± 1
  - SpMV requires $\mathcal{O}(r \cdot \text{nnz})$ FLOPs but sparse memory access patterns

Comparison

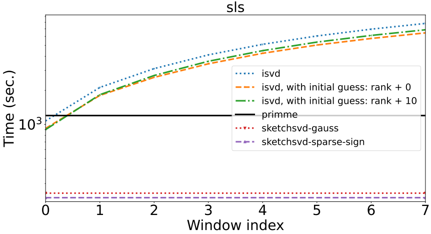
Complexity

- $A \in \mathbb{R}^{n \times n}$ dense, 8 matvecs per singular value:
  - iSVD cost $\approx$ SketchySVD (Gaussian maps)
  - SketchySVD (sparse maps) $\approx r\times$ faster
- $A \in \mathbb{R}^{n \times n}$ sparse:
  - SketchySVD (sparse maps, 8 nnz/col) $\approx 4r\times$ faster than iSVD

HPC experiments

- Implemented both methods based on **Kokkos**
- Window SVDs solved with **PRIMME** to relative residual 1E-4
- Compare time and accuracy against **PRIMME** on A (rel. res 1E-4)
- Normalized spectra $\sigma_1 = 1$

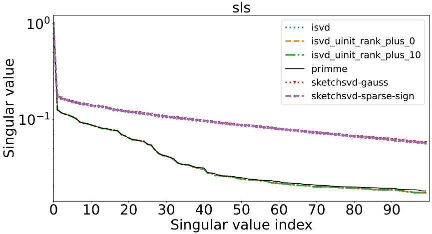
SuiteSparse Comparisons: SLS

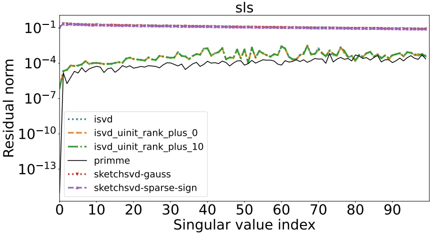


1,748,122×62,729 3.9 nnz/row

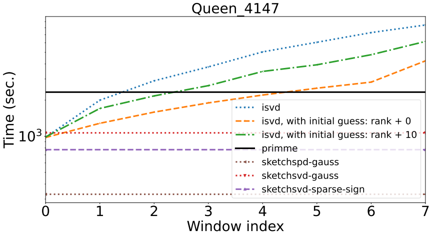
Sketchy	$\frac{1}{10}\times$, but inaccurate
iSVD	$6\times$, but accurate

Initial guesses do not help





SuiteSparse Comparisons: Queen_4147

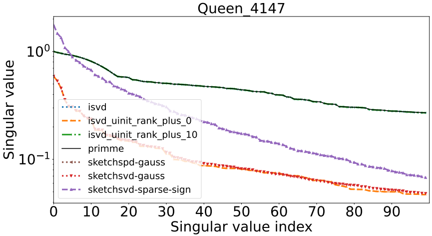


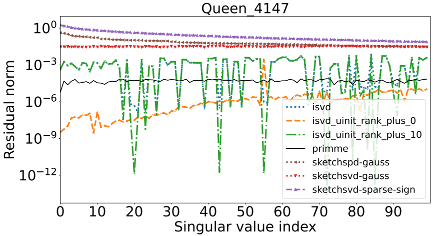
4,147,110 symmetric

76 nnz/row, 149447 connect. comp.

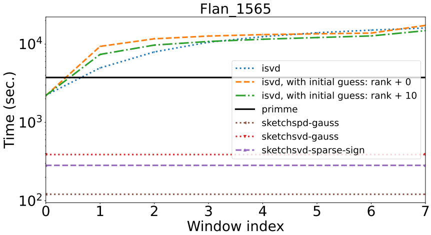
Sketchy	$\frac{1}{4}\times$, less inaccurate
iSVD	$3\times$, *"accurate"*

Initial guesses help but need $+10$





SuiteSparse Comparisons: FLAN

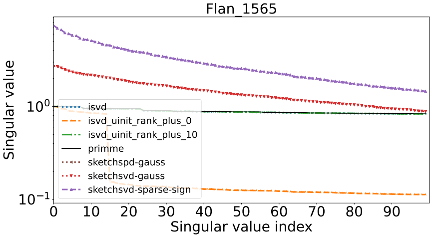


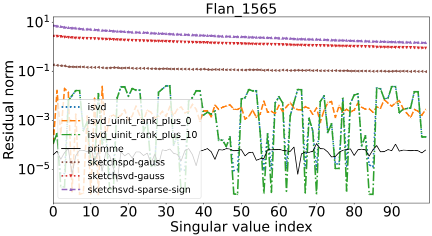
1,564,794 symmetric

73 nnz/row, 1 connect. comp.

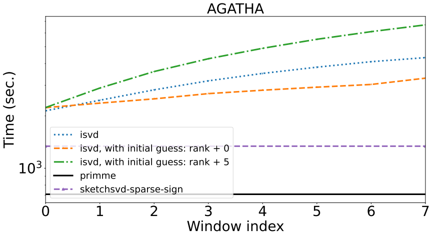
Sketchy	$\frac{1}{30}\times$, SPD a bit better	
iSVD	$5\times$, accurate	

Initial guesses no help (drift)





SuiteSparse Comparisons: Agatha

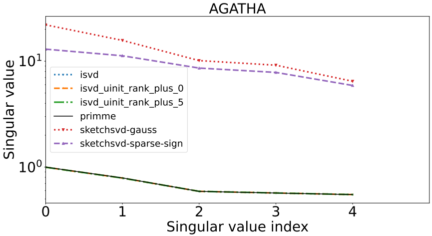


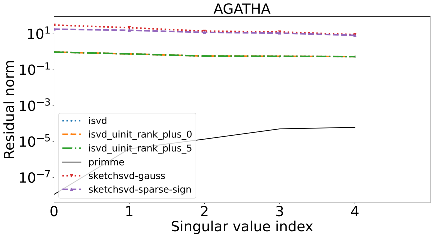
183,964,077 undirected graph

63 nnz/row

Sketchy	$2\times$ (!), inaccurate
iSVD	$4\times$, gets 1 rel. digit

Initial guesses help ($2.5\times$)





Optimization roadmap

If low accuracy sufficient, iSVD with iterative method promising

Potential optimizations:

If we could monitor $R_t^{(i)} = ||A(I - V_t^{(i)} V_t^{(i)T})||$ (window i, iteration t) possible early stopping for:

1. solver, if $R_t^{(i)}$ stagnates—oblivious to final LR accuracy
2. iSVD, if at some window we met $R^{(i)} <$ userTol
   - advantage over iterative with on-the-fly Matvec

Optimization roadmap

If low accuracy sufficient, iSVD with iterative method promising

Potential optimizations:

If we could monitor $R_t^{(i)} = ||A(I - V_t^{(i)} V_t^{(i)T})||$ (window i, iteration t) possible early stopping for:

1. solver, if $R_t^{(i)}$ stagnates—oblivious to final LR accuracy
2. iSVD, if at some window we met $R^{(i)} <$ userTol
   - advantage over iterative with on-the-fly Matvec

$\boxed{A \text{ not available}}$

Uniform sampling of rows of A

Extension 2

During streaming, build $\bar{A}$, a uniform sample of s rows of A

Reservoir Sampling (JSVitter 85):

- Pick the first s rows of A
- Streamed row i replaces a row in $\bar{A}$ with probability s/i

At any row i, $\bar{A}$ is a uniform sample of $A(1:i,:)$

Estimate residual norm:

$$\bar{R} = \frac{m}{s} \sum_{j \in S} \|\bar{A}_j - (\bar{A}_j V) V^T\|^2$$

Optimizations for symmetric (eigenproblems)

Motivation: Kernel, Distance, Covariance matrices
Large scale data can be stored $O(N)$ but not their covariance $O(N^2)$

Estimate eigenvalue residual norm:

$$\bar{R} = \frac{m}{s} \sum_{j \in S} \|\bar{A}_j V_t^{(i)} - V_t^{(i)}(j,:)\Sigma_t^{(i)}\|^2$$

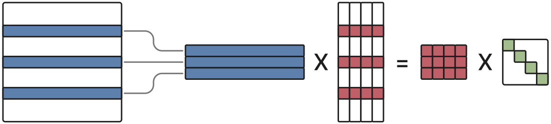
at iteration t of window i

However, even if exact eigenspace V is captured, Σ may not correspond to Λ till last window.

Least-Squares Method

For symmetric:

$$A \xrightarrow{\text{Sample rows}} A_S \times V = V_S \times \Lambda$$



Extension 3

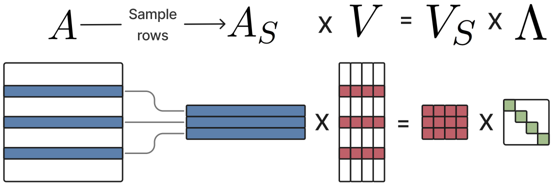
For a given V: $\min_\lambda \|V_S\lambda - A_S V\|_F^2$

Solution: $\lambda = v_S^+ A_S v = \frac{v_S^T A_S v}{v_S^T v_S}$ (Rayleigh quotient-like)

For nonsymmetric: $u_S = A_S v/\sigma$, and $\sigma' = u_S^+ A_S v$

Demix Method

For symmetric only:



Scenario: Accurate space may still mix eigenvectors

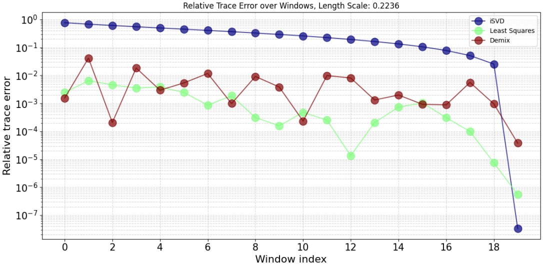
Extension 4

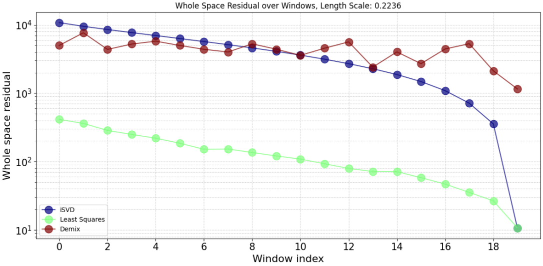
For a given V, $\min\limits_{c,\Lambda} \|A_S V c - V_S c \Lambda\|_F^2$

Solution: Let $V_S = QR$, $\min\limits_{c,\Lambda} \|Q^T(A_S V c - R c \Lambda)\|_F^2$, by solving a **GEP**

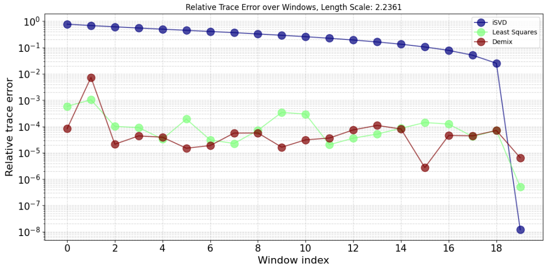
Complexity: $O(r^3)$ for rank r

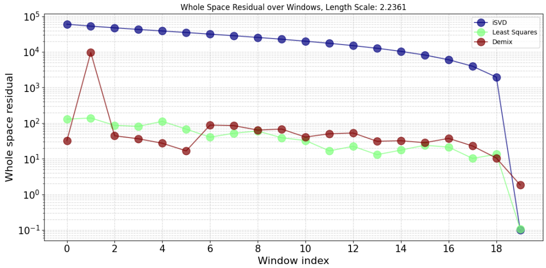
Kernel matrix (stock data). 100K. Slow decay, high rank





Kernel matrix (stock data). 100K. Fast decay, low rank





Conclusions and current work

- iSVD more expensive but much more accurate than SketchySVD
- Several extensions to improve iSVD

Next steps:

- Dynamic stopping for PRIMME eigensolver
- Early stopping of streaming
- Least squares (Ext 3) allows the use of Frequent Directions
- Better: a Randomized row order iSVD convergence analysis
- Sketch incrementally provides leverage scores to change streaming order