Accuracy and efficiency of one–sided bidiagonalization algorithm

Nela Bosner
Department of Mathematics, University of Zagreb
Bijenička cesta 30, 10000 Zagreb
Croatia

June 1, 2004

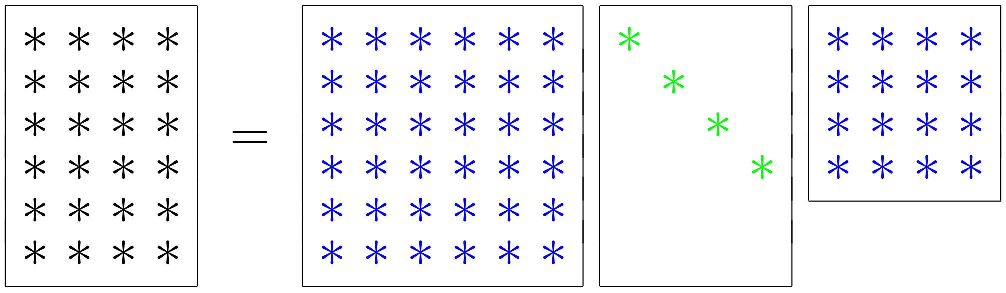
1 The singular value decomposition (SVD)

The singular value decomposition (SVD) of a general matrix is the fundamental theoretical and computational tool in numerical linear algebra.

Definition 1. *Let $A \in \mathbb{R}^{m\times n}$ be a general matrix.* The singular value decomposition *of matrix A is a factorization of form*

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

where $U \in \mathbb{R}^{m\times m}$ and $V \in \mathbb{R}^{n\times n}$ are orthogonal, and $\Sigma = diag(\sigma_1, \ldots, \sigma_n) \in \mathbb{R}^{n\times n}$ is diagonal with $\sigma_1 \geq \sigma_2 \geq \cdots \geq \sigma_n \geq 0$.



$*$ = general rectangular matrix
$*$ = orthogonal matrix
$*$ = diagonal matrix

Application of SVD :

- Solving symmetric definite eigenvalue problem, where matrix is factorized as $A = G^T G$
- Linear least squares
- Total least squares
- Orthogonal Procrustes problem
- Finding intersection of two null–spaces
- Finding principal angles between two subspaces
- Handling rank deficiency and subset selection

2 Numerical computation of SVD

What is important for algorithms that solve SVD numerically?

- Numerical stability — algorithm produces a result with small error.
- Efficiency — algorithm is fast when applied on modern computers.

Computing SVD :

1. Reduction of the matrix to bidiagonal form via orthogonal transformations. SVD of bidiagonal matrix is computed with an iterative method, in full accuracy. (faster)
2. Jacobi SVD algorithm. (more robust)

3 LAPACK routine for solving SVD

LAPACK routine **_GESVD** uses the 1. approach, with **Householder bidiagonalization**. The bidiagonalization produces U and V as products of appropriate Householder reflectors:

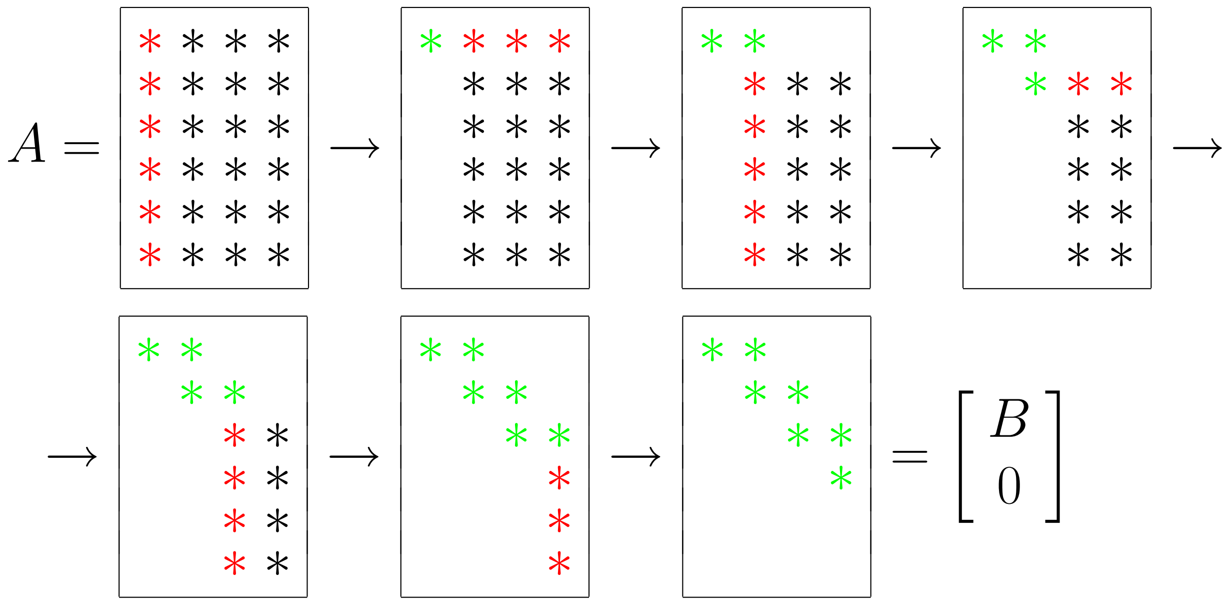
$$U^T A V = \begin{bmatrix} B \\ 0 \end{bmatrix}, \quad B = \begin{bmatrix} \alpha_1 & \beta_1 & & \\ & \ddots & \ddots & \\ & & \ddots & \beta_{n-1} \\ & & & \alpha_n \end{bmatrix} \in \mathbb{R}^{n\times n},$$

where

$$U = U_1 \cdots U_n \in \mathbb{R}^{m\times m}, \; V = V_1 \cdots V_{n-2} \in \mathbb{R}^{n\times n},$$

U_i, V_j are Householder reflectors.

Algorithm 1 (Householder bidiagonalization).



3.1 Numerical analysis of Householder bidiagonalization

When the bidiagonalization is performed on computer, in finite precision arithmetic, then

- bidiagonalization is numerically backward stable:

$$A + \delta A = \tilde{U} \begin{bmatrix} \tilde{\Sigma} \\ 0 \end{bmatrix} \tilde{V}^T,$$

  where $\tilde{U}$, $\tilde{\Sigma}$, $\tilde{V}$ are computed matrices, and

$$\|\delta A\| \le \mathcal{O}(\epsilon)\|A\|.$$

- computed matrices $\tilde{U}$ and $\tilde{V}$ are numerically orthogonal:

$$\begin{aligned} \tilde{U} &= \hat{U} + \delta U, \quad \|\delta U\| \le \mathcal{O}(\epsilon), \\ \tilde{V} &= \hat{V} + \delta V, \quad \|\delta V\| \le \mathcal{O}(\epsilon), \end{aligned}$$

(ϵ is unit roundoff.)

3.2 Drawbacks of Householder bidiagonalization

- Bidiagonalization involves much more computational work than iterative method for computing singular values of B.

- Bidiagonalization is **two-sided** algorithm, which employs both pre-multiplication and post-multiplication of A by Householder reflections. It is **difficult** to implement it efficiently on **multiprocessor systems with distributed memory**.

4 Ralha one-sided bidiagonalization

Main steps of the algorithm:

- **Triorthogonalization**
  matrix is post-multiplied with a sequence of $n-2$ Householder reflectors:
  $$A_0 = A,\ A_i = A_{i-1}H_i, \quad i = 1, \ldots, n-2.$$
  Householder reflectors are chosen so that
  $$F = A_{n-2} \quad \text{is triorthogonal:}$$
  for columns f_i, f_j of F
  $$f_i^T f_j = 0, \quad |i-j| > 1.$$
  This implies that $F^T F$ is tridiagonal.
- **A variant of the Gram–Schmidt orthogonalization**
  The columns of F are orthogonalized only against adjacent columns, producing
  $$F = QB,$$
  where $Q \in \mathbb{R}^{m \times n}$ is orthogonal and $B \in \mathbb{R}^{n \times n}$ is required upper bidiagonal matrix, whose singular values are those of A.

Algorithm 2 (Ralha one-sided bidiagonalization). *Implicitly:*

$$A^TA = \boxed{\begin{array}{cccc} * & \color{red}{*} & \color{red}{*} & \color{red}{*} \\ \color{red}{*} & * & * & * \\ \color{red}{*} & * & * & * \\ \color{red}{*} & * & * & * \end{array}} \rightarrow \boxed{\begin{array}{cccc} \color{blue}{*} & \color{blue}{*} & & \\ \color{blue}{*} & * & \color{red}{*} & \color{red}{*} \\ & \color{red}{*} & * & * \\ & \color{red}{*} & * & * \end{array}} \rightarrow \boxed{\begin{array}{cccc} \color{blue}{*} & \color{blue}{*} & & \\ \color{blue}{*} & \color{blue}{*} & \color{blue}{*} & \\ & \color{blue}{*} & \color{blue}{*} & \color{blue}{*} \\ & & \color{blue}{*} & \color{blue}{*} \end{array}} \rightarrow F^TF$$

Explicitly:

$$A \rightarrow F \rightarrow \boxed{\begin{array}{c} \color{green}{*} \\ \\ \\ \\ \end{array}} \rightarrow \boxed{\begin{array}{cc} \color{green}{*} & \color{green}{*} \\ & \color{green}{*} \\ & \\ & \end{array}} \rightarrow \boxed{\begin{array}{ccc} \color{green}{*} & \color{green}{*} & \\ & \color{green}{*} & \color{green}{*} \\ & & \color{green}{*} \\ & & \end{array}} \rightarrow \boxed{\begin{array}{cccc} \color{green}{*} & \color{green}{*} & & \\ & \color{green}{*} & \color{green}{*} & \\ & & \color{green}{*} & \color{green}{*} \\ & & & \color{green}{*} \end{array}} = B$$

4.1 Improvements in Ralha one-sided bidiagonalization

- The version of Gram–Schmidt orthogonalization in this algorithm substitutes pre-multiplication with a sequence of Householder reflection in Householder bidiagonalization. Thus, the second part of the algorithm requires only $\mathcal{O}(\mathbf{mn})$ flops instead of $\mathcal{O}(n^2m)$ flops.
- Algorithm is one-sided: the most of the algorithm can be expressed in simple operations on columns of transformed matrix. Thus, it can be implemented on **multiprocessor systems with distributed memory**.

4.2 Drawbacks of Ralha one-sided bidiagonalization

- Possible great **loss of triorthogonality** of the computed matrix $\tilde{F}$. This means that $\tilde{F}^T\tilde{F}$ can be far from tridiagonal form. Thus, this method **cannot be numerically backward stable**.
- Possible great **loss of orthogonality** of the computed matrix $\tilde{U}$.

5 Barlow one-sided bidiagonalization

Main steps of the algorithm:

- **Direct bidiagonalization**
  one step of Gram–Schmidt orthogonalization and post-multiplication with one Householder reflector are performed simultaneously:

  u_i is produced from orthogonalization against adjacent column,

$$U_i = [u_i, \ldots, u_i] \quad i = 1, \ldots, n$$

$$A_0 = A, \quad A_i = A_{i-1} H_i \quad i = 1, \ldots, n-2,$$

  and H_i i chosen so that

$$U_i^T A_{i-1} H_i = B_i \in \mathbb{R}^{i \times n}$$

  is bidiagonal.

Algorithm 3 (Barlow one-sided bidiagonalization). *Simultaneous computation:*

$$A = \boxed{\begin{array}{cccc} * & * & * & * \\ * & * & * & * \\ * & * & * & * \\ * & * & * & * \\ * & * & * & * \\ * & * & * & * \end{array}} \rightarrow \boxed{\begin{array}{c} * \\ * \\ * \\ * \\ * \\ * \end{array}} \rightarrow \boxed{\begin{array}{cc} * & * \\ * & * \\ * & * \\ * & * \\ * & * \\ * & * \end{array}} \rightarrow \boxed{\begin{array}{ccc} * & * & * \\ * & * & * \\ * & * & * \\ * & * & * \\ * & * & * \\ * & * & * \end{array}} \rightarrow \boxed{\begin{array}{cccc} * & * & * & * \\ * & * & * & * \\ * & * & * & * \\ * & * & * & * \\ * & * & * & * \\ * & * & * & * \end{array}} = F$$

$$B = \square \rightarrow \boxed{*} \rightarrow \boxed{\begin{array}{cc} * & * \\ & * \end{array}} \rightarrow \boxed{\begin{array}{ccc} * & * & \\ & * & * \\ & & * \end{array}} \rightarrow \boxed{\begin{array}{cccc} * & * & & \\ & * & * & \\ & & * & * \\ & & & * \end{array}}$$

5.1 Improvements in Barlow one-sided bidiagonalization

- The same number of flops as in the Ralha one-sided bidiagonalization.
- It can be implemented on multiprocessor systems with distributed memory.
- <u>NUMERICALLY BACKWARD STABLE</u>.

5.2 Drawbacks of Barlow one-sided bidiagonalization

- Possible great **loss of orthogonality** of the computed matrix $\tilde{U}$.

6 Backward stability of Barlow bidiagonalization

- shown by Barlow
- our proof, different approach

Theorem 1. *If $\tilde{B}$ is the bidiagonal matrix computed by Algorithm 3 without breakdown, then there exist an $(m+n)\times(m+n)$ orthogonal matrix $\hat{P}$, an orthogonal $n\times n$ matrix $\hat{V}$ and backward perturbations ΔA, δA such that*

$$\begin{bmatrix} \tilde{B} \\ 0 \end{bmatrix} = \hat{P}^T \begin{bmatrix} \Delta A \\ A+\delta A \end{bmatrix} \hat{V}, \quad \left\| \begin{bmatrix} \Delta A \\ \delta A \end{bmatrix} \right\|_F \le \xi \|A\|_F, \tag{1}$$

where $0 \le \xi \le \mathcal{O}(mn^2\epsilon) + \mathcal{O}(\epsilon^2)$. The computed approximation $\tilde{V}$ of the matrix $\hat{V}$ satisfies

$$\|\tilde{V} - \hat{V}\|_F \le \mathcal{O}(n^2\epsilon). \tag{2}$$

Further, there exist an orthonormal $\hat{U}$ and perturbation $\delta\hat{A}$ such that

$$A + \delta\hat{A} = \hat{U}\tilde{B}\hat{V}^T, \quad \|\delta\hat{A}\|_F \le \sqrt{2}\xi\|A\|_F. \tag{3}$$

Corollary 1. *If $\sigma_i \ge \cdots \ge \sigma_n$ are the singular values of A, then the singular values $\tilde{\sigma}_1 \ge \cdots \ge \tilde{\sigma}_n$ of $\tilde{B}$ from Theorem 1 satisfy*

$$\max_{i=1,\ldots,n} |\tilde{\sigma}_i - \sigma_i| \le \xi \sqrt{\sum_{j=1}^{n} \sigma_j^2}.$$

Corollary 2. *If the Algorithm 3 is implemented in parallel with certain data distribution, then the assertion of Theorem 1 remains true for both Householder and Givens transformation based eliminations.*

7 Efficiency of Barlow bidiagonalization

Computing full SVD.

7.1 Floating point operation count

Algorithm 1	Algorithm 1 with QR	Algorithms 2 & 3
$8mn^2 + \frac{4}{3}n^3$	$6mn^2 + \frac{20}{3}n^3$	$5mn^2 + \frac{10}{3}n^3$

7.2 Numerical computations

- Computations were performed in Advanced computing laboratory on Department of Mathematics, University of Zagreb.
- The laboratory consists of 20 computers, connected in local 1Gb network.
- The propositions of computers are:

2 processors	Athlon Mp 1800+
Frequency	1533MHz
L1 Cache	64Kb
L2 Cache	256Kb
RAM	1Gb

7.3 Execution time

Numerical results (time in seconds)

$m \times n$	Barlow	LAPACK	$\Delta t\%$
100×100	0.01	0.01	0.00%
200×200	**0.14**	0.15	6.67%
500×50	0.01	0.01	0.00%
500×100	0.05	**0.04**	-25.00%
500×500	3.87	**3.47**	-11.53%
1000×100	0.14	**0.09**	-55.56%
1000×500	6.19	**4.43**	-39.73%
1000×1000	39.19	**36.95**	-6.06%
2000×200	1.46	**0.61**	-139.34%
2000×1000	55.25	**41.63**	-32,72
2000×2000	359.05	**326,75**	-9.89%
3000×3000	1514.46	**1300.94**	-16.41%

7.4 Conclusion

- Despite the fact that SVD solver with Barlow one-sided bidiagonalization require less floating point operations, execution time is longer than the execution time of LAPACK routine.
- This happens because LAPACK routine has optimized usage of cache memory, and that's not the case with Barlow bidiagonalization.

8 Block version of Barlow one-sided bidiagonalization

Optimizes usage of cache memory for Barlow one-sided bidiagonalization.

- Transformations by Householder reflectors are aggregated. Matrix A is updated in every b steps. Algorithm uses more BLAS-3 operations.
  (b="dimension of the block")
- Algorithm implements BLAS-2.5 approach proposed by G.W.Howell. Operations on same data, that were performed on different places in Algorithm 3, are now performed at once. These operations are

$$\begin{aligned} x &\leftarrow x + A^T y \\ w &\leftarrow w + Ax \end{aligned} \qquad \text{or} \qquad \begin{aligned} A &\leftarrow A + uv^T \\ x &\leftarrow A^T y \\ w &\leftarrow w + Ax \end{aligned}$$

9 Backward stability of block Barlow bidiagonalization

Theorem 2. *If $\tilde{B}$ is the bidiagonal matrix computed by block version of Algorithm 3 with block dimension b without breakdown, then there exist an $(m+n) \times (m+n)$ orthogonal matrix $\hat{P}$, an orthogonal $n \times n$ matrix $\hat{V}$ and backward perturbations ΔA, δA such that*

$$\begin{bmatrix} \tilde{B} \\ 0 \end{bmatrix} = \hat{P}^T \begin{bmatrix} \Delta A \\ A + \delta A \end{bmatrix} \hat{V}, \quad \left\| \begin{bmatrix} \Delta A \\ \delta A \end{bmatrix} \right\|_F \leq \xi \|A\|_F, \tag{4}$$

where $0 \leq \xi \leq \mathcal{O}(bmn^2\epsilon) + \mathcal{O}(\epsilon^2)$. The computed approximation $\tilde{V}$ of the matrix $\hat{V}$ satisfies

$$\|\tilde{V} - \hat{V}\|_F \leq \mathcal{O}(n^2\epsilon). \tag{5}$$

Further, there exist an orthonormal $\hat{U}$ and perturbation $\delta\hat{A}$ such that

$$A + \delta\hat{A} = \hat{U}\tilde{B}\hat{V}^T, \quad \|\delta\hat{A}\|_F \leq \sqrt{2}\xi\|A\|_F. \tag{6}$$

10 Efficiency of block Barlow bidiagonalization

Numerical results (time in seconds)

$m \times n$	block Barlow	LAPACK	$\Delta t\%$	$\Delta t_B\%$
100×100	0.01	0.01	0.00%	0.00%
200×200	**0.13**	0.15	13.33%	7.14%
500×50	*0.01	0.01	0.00%	0.00%
500×100	*0.04	0.04	0.00%	20.00%
500×500	**3.40**	3.63	6.34%	16.02%
1000×100	*0.09	0.09	0.00%	35.71%
1000×500	***4.04**	4.45	9.21%	35.06%
1000×1000	**34.55**	37.43	7.69%	12.96%
2000×200	***0.58**	0.60	3.33%	46.30%
2000×1000	***39.22**	41.20	4.81%	28.27%
2000×2000	**324.59**	336.49	3.54%	12.22%
3000×3000	**1261.34**	1318.24	4.32%	17.81%

* — QR factorization is performed before SVD.

11 Parallel version of Barlow one-sided bidiagonalization

Respectable decrease in execution time.

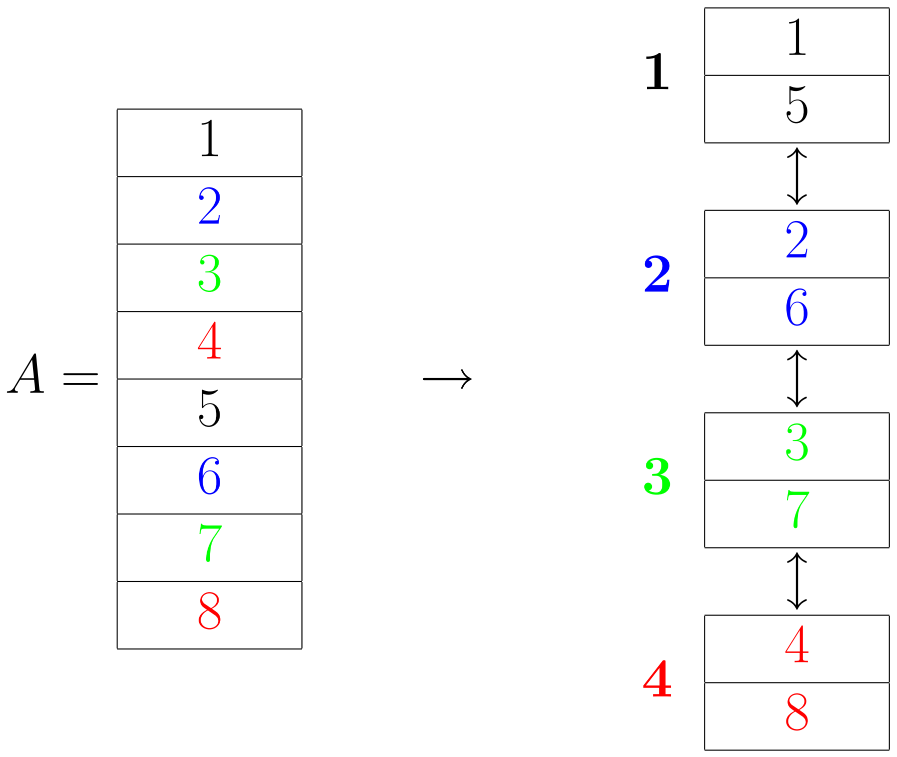
1. Algorithm is performed on more processors simultaneously.
2. Each matrix is distributed over the memories of every processor.
3. Communication between processors is optimized. (Interprocessors communication is most time consumable.)

In our case we used following propositions.

1. Processors were organized in linear order.

$$\begin{tikzcd}\boxed{\mathbf{1}} \arrow[r, leftrightarrow] & \boxed{\mathbf{2}} \arrow[r, leftrightarrow] & \boxed{\mathbf{3}} \arrow[r, leftrightarrow] & \boxed{\mathbf{4}}\end{tikzcd}$$

2. Matrix layout was **one-dimensional block-cyclic row distribution**. Each $m \times n$ matrix is divided in $m_b \times n$ blocks of continuous rows. (m_b=block row dimension.) Blocks are distributed across the processors in cyclic order.
   Good load balancing.



3. Interprocessors communication required only for broadcasting Householder vectors and computing scalar products.

Parallel version of algorithm performs the same operations as serial non-block version. (Parallel block version has too big overhead.) Results of **Theorem 1** hold for this version as well.

12 Efficiency of parallel Barlow bidiagonalization

Numerical results
(time in seconds – the worst time on all #p processors)

$m \times n$	# p	parallel Barlow	ScaLAPACK	$\Delta t\%$
1000×100	4	**0.25**	0.51	50.98%
1000×500	4	**3.00**	4.24	29.25%
1000×1000	4	**13.42**	16.90	20.59%
	8	**9.11**	17.12	46.79%
	16	**8.15**	17.60	53.69%
2000×200	4	**0.81**	1.46	44.52%
2000×1000	4	**17.50**	18.38	4.79%
	8	**11.75**	18.63	36.93%
2000×2000	4	**105.38**	106.97	1.49%
	8	**53.77**	70.68	23.92%
	16	**33.92**	61.18	44.56%
4000×200	8	**1.11**	3.12	64.42%
4000×1000	8	**18.59**	23.20	19.87%
	16	**12.47**	21.26	41.35%
4000×4000	8	**427.87**	473.78	9.69%
	16	**196.28**	254.07	22.75%
5000×100	8	**0.54**	2.03	73.40%
5000×1000	16	**14.03**	21.79	35.61%
5000×5000	16	**400.02**	478.53	16.41%
8000×1000	16	**17.78**	24.77	28.22%
8000×8000	16	**2348.45**	2546.79	7.79%
10000×1000	16	**22.04**	25.87	14.80%
10000×10000	16	**3663.53**	3735.78	1.93%

13 Conclusion

- Ralha's assumption that the one-sided bidiagonalization is much more efficient on parallel computers than ScaLAPACK routine is proved to be **right** by numerous numerical examples.
- Barlow's assumption about numerical stability of his modification of one-sided bidiagonalization is also proved to be **right**.
- Our block version of one-sided bidiagonalization detains the **numerical stability** property, and decreases execution time by optimizing communication between fast and slow memory.
- In future work we can use some gained time in improving orthogonality of matrix $\tilde{U}$.