Multi-layer Randomized Singular Value Decomposition via Optical Processing Units

XiangYu Wang fkkk@live.com

ABSTRACT

Optical processing unit (OPU) [1] is a computation device that performs operations using light and hundreds of times faster than CPU. Due to the limited computing power of logic circuit, the decomposition of big matrix is very hard. Thus, we propose a greedy algorithm to decompose matrix via OPU. The experiment shows our algorithm performs better than randomized singular value decomposition (RSVD) [2] for non-negative matrix. However, there are still many problems to be solved.

1. INTRODUCE

1.1. Matrix Decomposition

Matrix decomposition plays a very important role in image/signal processing and data analysis. A $d \times n$ matrix $X = [x_1, x_2, \dots, x_n]$ can be regarded as n points in d-dimensional space, then given an orthogonal basis $Q = [q_1, q_2, \dots, q_k]$, the projection of these points on Q is QQ^TX, which is a k-rank approximation of X with a squared error of

$$\|X - QQ^TX\|_F^2 = \|X\|_F^2 - \|X^TQ\|_F^2 = \|X\|_F^2 - \sum_{i=1}^{k} \|X^Tq_i\|_2^2$$

Truncated singular value decomposition (TSVD) [3] can finds the optimal k-rank approximation of X. However, the time and space complexity of TSVD is so high that it cannot be applied to a large matrix. In many cases, we must use approximate algorithm to decompose big matrices. RSVD is one of the most used approximation algorithms for matrix decomposition. However, RSVD still spend many times due to the limited computing power of CPU.

1.2. OPU

An OPU is a device to quickly compute the multiplication by a dense random matrix using light. Given a $d \times n$ matrix X, the OPU will return:

$$Y = |XR|^2$$

Where R is the random matrix of size $(n \times t)$, containing independent and identically distributed (i.i.d.) Gaussian complex values with expectation 0:

$$R_{i,j} = uX + ivY, X \text{ and } Y \text{ are i.i.d. Standard Gaussian Values}, u, v > 0$$

Normally, the OPU operation is hundreds of times faster than CPU.

1.3. Problem Formulation

Given a $d \times n$ matrix X such that XX^T is non-negative, use OPU to find an orthogonal basis $Q = [q_1, q_2, \dots, q_k]$ to make $\sum_{i=1}^{k} \|X^Tq_i\|_2^2$ as large as possible.

2. ALGORITHM

2.1. Multi-layer Randomized Singular Value Decomposition (MRSVD) via OPU

```
Given $X, k, t, \alpha, p$
$S$ = Random sub-columns of $X$, each column will be sampled with probability $\alpha$
$Q$ = []
for $i = 1$ to $k + p$
```

```
        $P$ = Generate $t$ random vectors by OPU
        $P$ = sqrt($P$)
        $P = P - QQ^T P$
        $P = \left[\frac{p}{\|p\|_2} \text{ for } p \text{ in } P\right]$
        $c = \underset{c}{\operatorname{argmax}} \|S^T P^{(c)}\|_2^2$
        Add $P^{(c)}$ to $Q$
    Apply TSVD to get the optional $k$-rank approximation of $QQ^T X$
```

Algorithm 1: MRSVD via OPU

2.2.Error Analysis & Algorithm Explanation

We use RSVD as the benchmark. Let $X_k^{(r)}$ denote the k-rank approximation of X found by RSVD and $X_k^{(m,t')}$ denote that found by our algorithm when $t = t'$. Obviously, both $E\left(\left\|X - X_k^{(r)}\right\|_F^2\right)$ and $E\left(\left\|X - X_k^{(m,t)}\right\|_F^2\right)$ decrease with the increase of k.

Theorem 1:

Let X be a $d \times n$ matrix such that XX^T is non-negative, when d is large:

$$E\left(\left\|X - X_1^{(r)}\right\|_F^2\right) \geq E\left(\left\|X - X_1^{(m,1)}\right\|_F^2\right)$$

Thus, although $E\left(\left\|X - X_1^{(m,1)}\right\|_F^2\right)$ may be greater than $E\left(\left\|X - X_1^{(r)}\right\|_F^2\right)$ when $k > 1$, $E\left(\left\|X - X_1^{(m,1)}\right\|_F^2\right) - E\left(\left\|X - X_1^{(r)}\right\|_F^2\right)$ should be not big, for example:

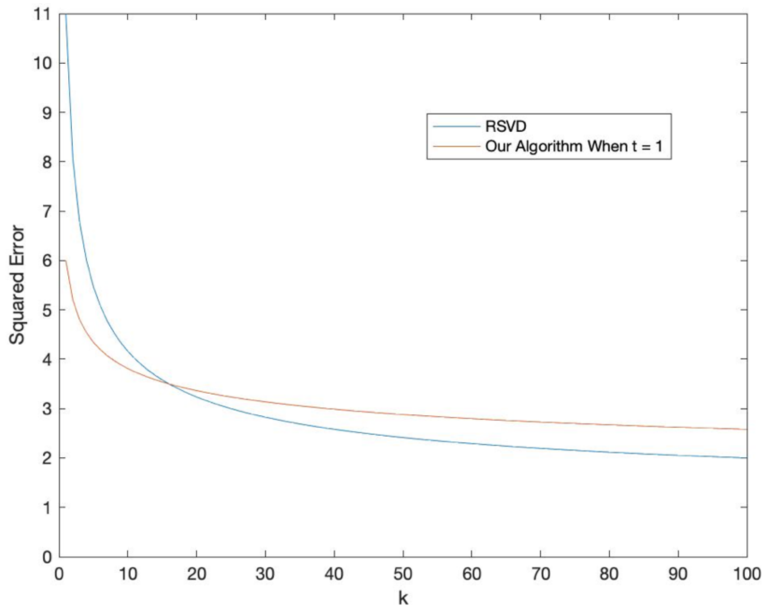

Figure 1: An example of $E\left(\left\|X - X_k^{(m,1)}\right\|_F^2\right) > E\left(\left\|X - X_k^{(r)}\right\|_F^2\right)$ when $k > 1$

It is clear that $E\left(\left\|X - X_k^{(m,1)}\right\|_F^2\right) < E\left(\left\|X - X_k^{(r)}\right\|_F^2\right)$ if $k \ll \min(n, d)$. However, when k is relatively large, $E\left(\left\|X - X_k^{(m,1)}\right\|_F^2\right)$ is likely to larger than $E\left(\left\|X - X_k^{(r)}\right\|_F^2\right)$. To solve such situation, we make $t > 1$ to reduce the squared error: In each iteration, randomly generate multiple bases $P = \{p_1, p_2, \dots, p_t\}$ by OPU, then choose $p = \underset{p \in P}{\operatorname{argmax}} \|X^T p\|^2$.

2.3.Cost Analysis

We use block orthogonal matching pursuit (Block OMP) [4] algorithm (a greedy algorithm proposed by Yining Wang and Aarti Singh in 2018 for matrix decomposition) as the benchmark. Block OMP picks k columns of X as the basis of approximate matrix. The rough idea of Block OMP is:

```
Given $X, k, t, \alpha, p$
$S$ = Random sub-columns of $X$, each column will be sampled with probability $\alpha$
$Q$ = [], $P$ = $X$.copy()
for $i$ = 1 to $k + p$:
    $P = \left[\frac{p}{\|p\|_2} \text{ for } p \text{ in } P\right]$
    $c = \underset{c}{\operatorname{argmax}} \left\|S^T P^{(c)}\right\|_2^2$
    Add $P^{(c)}$ to $Q$
    Remove $P^{(c)}$ from $P$
    $P = P - P^{(c)} {P^{(c)}}^T P$
Apply TSVD to get the optional $k$-rank approximation of $QQ^TX$
```

Algorithm 2: Block OMP

In each iteration of block OMP, the time complexity of normalization is $O(nd)$, that of choosing the unit vector with the maximum norm of projection vector is $O\left((\alpha dn^2)^{\frac{2.373}{3}}\right) + O(\alpha n^2)$[5], and that of removing $P^{(c)}$ and the component of other vectors on $P^{(c)}$ is $O(nd)$.

In each iteration of MRSVD, the time complexity of the calculating the square roots and normalization is $O(td)$, that of choosing the unit vector with the maximum norm of projection vector is $O\left((\alpha dnt)^{\frac{2.373}{3}}\right) + O(\alpha nt)$, and that of removing the component of P on Q is $O\left((kdt)^{\frac{2.373}{3}}\right) + O(td)$. Because $t \ll n$, our greedy algorithm run much faster than block OMP.

3. EXPERIMENT

A photo is a non-negative matrix. Thus, MRSVD can be used in image compression. In this work, a black and white photo of Jiang ZeMin is used for the experiment:

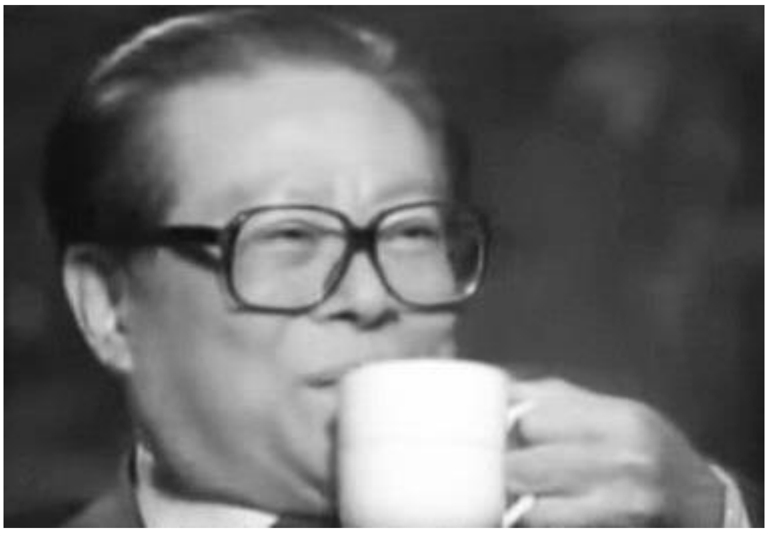

Figure2: Jiang ZeMin in "60 Minutes" on Sep 3, 2000, PC to Columbia Broadcasting System

The configure of the experiment is:

n	d	k	p	α
380	261	26	13	0.3

3.1.Numerical Result

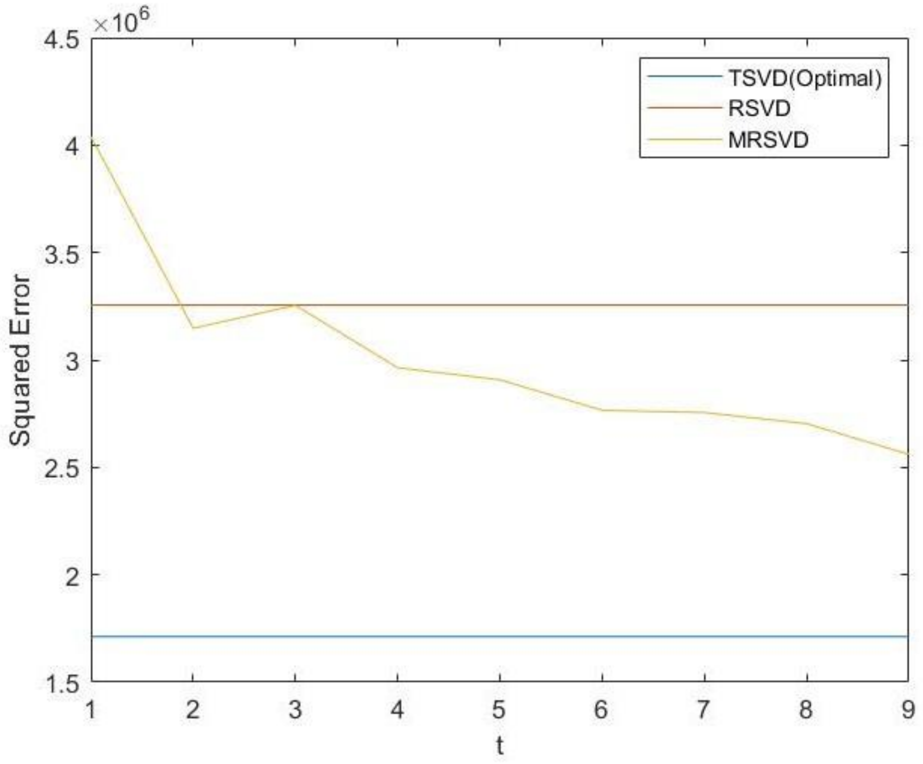

Figure3: The numerical performance

3.2.Visible Result

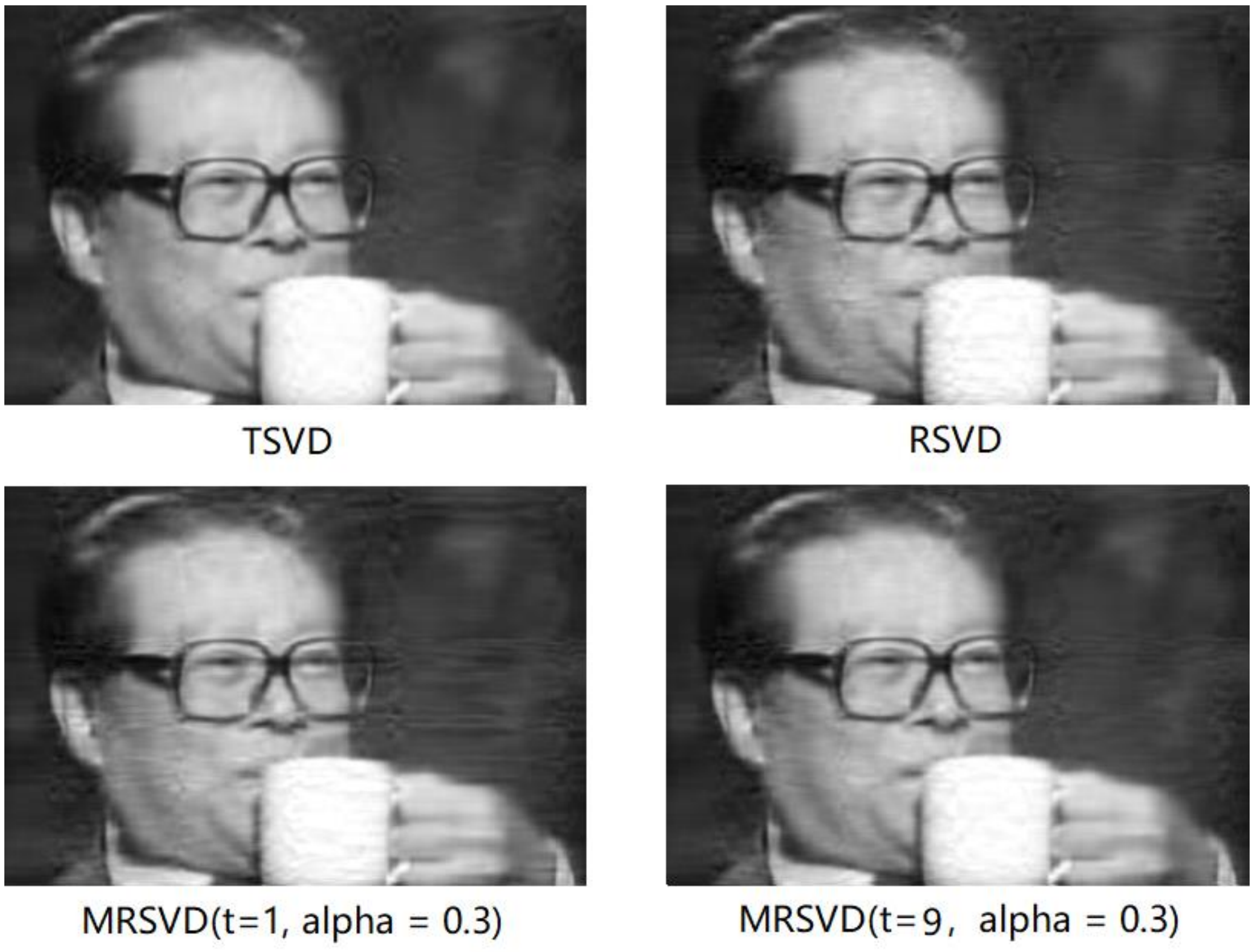

Figure 4: The visible performance

4. DISCUSSION

Although MRSVD via OPU performs well, there are still many problems to be solved:

4.1. Theoretical Analysis

There is no rigorous theoretical analysis:

4.1.1. We show that $E\left(\left\|X - X_k^{(m,1)}\right\|_F^2\right) < E\left(\left\|X - X_k^{(r)}\right\|_F^2\right)$ when $k \ll \min(n, d)$. However, how small is k specifically? Experiments show that $E\left(\left\|X - X_k^{(m,1)}\right\|_F^2\right) < E\left(\left\|X - X_k^{(r)}\right\|_F^2\right)$ if $k \leq 0.05n$, but we do not know why.

4.1.2. We intuitively think $E\left(\left\|X - X_k^{(m,1)}\right\|_F^2\right) - E\left(\left\|X - X_k^{(r)}\right\|_F^2\right)$ is not big. However, how 'not big' is it? Does it hold for any case?

4.1.3. What is the upper bound of $E\left(\left\|X - X_k^{(m,t)}\right\|_F^2\right)$? Does $E\left(\left\|X - X_k^{(m,t)}\right\|_F^2\right)$ always decrease with the increase of t?

So far, we have not found any relevant paper to solve these problems.

4.2. Principal Component Analysis (PCA) [6]

PCA is one of the most important uses of matrix decomposition. In PCA, Matrix X will be first centralized and then decomposed. Let X' denote X after centralization, then we desire that $X'X'^T$ is non-negative, i.e., any two dimensions of X are positively correlated. In many cases, it is impossible to make any pair of variables positively correlated by adding signs, for example:

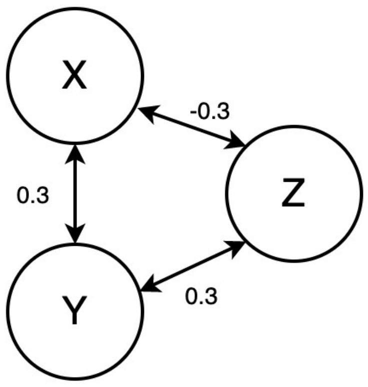

Figure 5: It is impossible to make any pair of variables in above picture positively correlated by adding signs

Thus, further research is needed to analyze the principal components via OPU.

5. REFERENCE

[1] LightOn Company, *What is an OPU?,* https://docs.lighton.ai/notes/opu.html

[2] N. Halko , P. G. Martinsson , J. A. Tropp, *Finding structure with randomness: Probabilistic algorithms for constructing approximate matrix decompositions*, 2009

[3] Scott Beaver, *The Singular Value Decomposition in Symmetric (Lowdin) Orthogonalization and Data Compression*, https://people.wou.edu/~beavers/Talks/Willamette1106.pdf

[4] Yining Wang and Aarti Singh. *Provably Correct Algorithms for Matrix Column Subset Selection with Selectively Sampled Data*, Journal of Machine Learning Research 18 (2018) 1-42

[5] F. Le Gall. *Powers of tensors and fast matrix multiplication*. In Proceedings of the 39th international

symposium on symbolic and algebraic computation, pages 296–303. ACM, 2014.

[6] Pearson, K. *On Lines and Planes of Closest Fit to Systems of Points in Space*. Philosophical Magazine. 1901, 2 (6): 559–572 [2012-01-24].

6. APPENDIX

6.1. Proof of Theorem 1

When $k = 1$ and d is large, the only base found by RSVD is:

$$\frac{Xa}{\|Xa\|_2} \approx \frac{Xa}{u\sqrt{d}}$$

Where $a = [a_1, a_2, \dots, a_n]^T$ is a random vector contains n i.i.d. Gaussian values with expectation 0 and standard deviation u.
And if $t = 1$, the base found by MRSVD via OPU is:

$$\frac{Xf}{\|Xf\|_2} \approx \frac{Xf}{\sqrt{u^2+v^2}\sqrt{d}}$$

Where $f = \left[\sqrt{b_1^2+c_1^2}, \sqrt{b_2^2+c_2^2}, \dots, \sqrt{b_n^2+c_n^2}\ \right]^T$ is a random vector: $b_1, b_2, \dots, b_n$ are n i.i.d. Gaussian values with expectation 0 and standard deviation u, and $c_1, c_2, \dots, c_n$ are n i.i.d. Gaussian values with expectation 0 and standard deviation v
Let $S = XX^T$, then:

$$\left\|X^T\frac{Xf}{\|Xc\|_2}\right\|_2^2 \approx \frac{\|X^TXf\|_2^2}{d(u^2+v^2)} = \frac{1}{d(u^2+v^2)}\sum_{i=1}^{n}\left(\sum_{j=1}^{d}X_{ji}f_j\right)^2 = \frac{1}{d(u^2+v^2)}\sum_{j=1}^{d}\sum_{k=1}^{d}f_jf_k\sum_{i=1}^{n}X_{ji}X_{ki}$$

$$= \frac{1}{d}\sum_{j=1}^{d}\sum_{k=1}^{d}S_{jk}\frac{f_if_j}{(u^2+v^2)}$$

$$\left\|X^T\frac{Xa}{\|Xa\|_2}\right\|_2^2 \approx \frac{\|X^TXa\|_2^2}{du^2} = \frac{1}{du^2}\sum_{i=1}^{n}\left(\sum_{j=1}^{d}X_{ji}a_j\right)^2 = \frac{1}{du^2}\sum_{j=1}^{d}\sum_{k=1}^{d}a_ja_k\sum_{i=1}^{n}X_{ji}X_{ki} = \frac{1}{d}\sum_{j=1}^{d}\sum_{k=1}^{d}S_{jk}\frac{a_ia_j}{u^2}$$

When $i = j$:

$$E\left(\frac{f_if_j}{u^2+v^2}\right) = E\left(\frac{b_i^2+c_i^2}{u^2+v^2}\right) = 1 = E\left(\frac{a_ia_j}{u^2}\right)$$

When $i \neq j$:

$$E\left(\frac{f_if_j}{u^2+v^2}\right) = E\left(\frac{\sqrt{b_i^2+c_i^2}\sqrt{b_j^2+c_j^2}}{u^2+v^2}\right) \geq E\left(\frac{|b_ib_j|+|c_ic_j|}{u^2+v^2}\right) = \frac{2}{\pi} > 0 = E\left(\frac{a_ia_j}{u^2}\right)$$

Thus,

$$E\left(\frac{f_if_j}{u^2+v^2}\right) \geq E\left(\frac{a_ia_j}{u^2}\right)$$

Because S is non-negative, so:

$$E\left(\left\|X - X_1^{(r)}\right\|_F^2\right) = \|X\|_F^2 - E\left(\left\|X^T \frac{Xa}{\|Xa\|_2}\right\|_2^2\right) = \|X\|_F^2 - \frac{1}{d}\sum_{j=1}^{d}\sum_{k=1}^{d} E\left(\frac{a_i a_j}{u^2}\right) S_{jk}$$

$$\geq \|X\|_F^2 - \frac{1}{d}\sum_{j=1}^{d}\sum_{k=1}^{d} E\left(\frac{f_i f_j}{u^2 + v^2}\right) S_{jk} = E\left(\left\|X - X_1^{(m,1)}\right\|_F^2\right)$$

6.2. Code:

https://raw.githubusercontent.com/xyw6/DataScience/master/MRSVD/MRSVD.ipynb

6.3. Data:

https://raw.githubusercontent.com/xyw6/DataScience/master/MRSVD/elder.JPG