# Notes for `drve08i`

Zhengbo Zhou*

21 January 2023

**Abstract**

*The paper mentioned in this notes is [6].* This paper gives a detailed preconditioning method on the classical cyclic one-sided Jacobi SVD algorithm. The authors achieved a fast (comparable with the QR algorithm) and accurate Jacobi SVD algorithm for any matrix $X$. Moreover, the authors provided lots of intuitions and motivations on the QR preconditioning methods not only in the sense of reducing dimension, but also in the sense of faster convergence.

The key element of this approach is the efficient QR preconditioners. The authors not only focused on how to reduce the dimension (therefore flops), but also on detecting the "digonality" of the implicit diagonalization, special pivoting to boost the convergence, etc. Moreover, the exciting part is how to construct the right handed singular vector. Normally, one can intuitively accumulate the Jacobi rotations to get the right handed singular vector matrix. However, the authors considered the construction of that singular vector matrix as a solution of the matrix equation $AX = B$ where $X$ is unknown, $B$ is any matrix and $A$ is triangular. This can be done using `STRSM` which is more efficient than just simply multiplying the orthogonal matrices together.

## 1 Introduction

Jacobi first notice this method while solving the following system of equations

$$\begin{cases} \{(a,a)-x\}\alpha & + & (a,b)\beta & + & (a,c)\gamma & +\cdots+ & (a,p)\tilde{\omega} & = 0 \\ (b,a)\alpha & + & \{(b,b)-x\}\beta & + & (b,c)\gamma & +\cdots+ & (b,p)\tilde{\omega} & = 0 \\ (c,a)\alpha & + & (c,b)\beta & + & \{(c,c)-x\}\gamma & +\cdots+ & (c,p)\tilde{\omega} & = 0 \\ (p,a)\alpha & + & (p,b)\beta & + & (p,c)\gamma & +\cdots+ & \{(p,p)-x\}\tilde{\omega} & = 0 \end{cases}$$

where the coefficients are symmetric, i.e. $(a,b) = (b,a), (a,c) = (c,a), \ldots, (b,c) = (c,b)$ etc. In our notation

$$H = \begin{pmatrix} (a,a) & (a,b) & (a,c) & \cdots & (a,p) \\ (b,a) & (b,b) & (b,c) & \cdots & (b,p) \\ (c,a) & (c,b) & (c,c) & \cdots & (c,p) \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ (p,a) & (p,b) & (p,c) & \cdots & (p,p) \end{pmatrix} = (H_{ij})_{i,j=1}^{n}, \quad u = \begin{pmatrix} \alpha \\ \beta \\ \gamma \\ \vdots \\ \tilde{\omega} \end{pmatrix}, \quad (H = H^T)$$

*Department of Mathematics, University of Manchester, Manchester, M13 9PL, England (`zhengbo.zhou@postgrad.manchester.ac.uk`).

we obtain the equation $(H - \lambda I)u = 0$, which is to be solved for $\lambda$ and $u \neq 0$. The algorithm starts with $H^{(0)} = H$ and then it generates a sequence of congruences, $H^{(k+1)} = (V^{(k)})^T H^{(k)} V^{(k)}$, where $V^{(k)}$ is the plane rotation, i.e. $V^{(k)}$ differs from the identity only at the position $(p_k, p_k), (p_k, q_k), (q_k, p_k), (q_k, q_k)$ where

$$\begin{pmatrix} V^{(k)}_{p_k,p_k} & V^{(k)}_{p_k,q_k} \\ V^{(k)}_{q_k,p_k} & V^{(k)}_{q_k,q_k} \end{pmatrix} = \begin{pmatrix} \cos(\phi_k) & \sin(\phi_k) \\ -\sin(\phi_k) & \cos(\phi_k) \end{pmatrix}.$$

The angle $\phi_k$ is determined to annihilate the $(p_k, q_k)$ and $(q_k, p_k)$ positions in $H^{(k)}$, and it is carefully chosen such that we always take $\phi_k$ as the smaller roots which ensures the convergence.

## 1.1 History

The advent of electronic computing machinery opened a new chapter in numerical mathematics and the Jacobi method becomes attractive tool for computing the eigenvalues of symmetric matrices. Around 1950, Goldstine, Murray and von Neumann [10] rediscover the method and had detailed implementation and error analysis. It was used by **Hessenberg** in 1940, by **Wilkinson** in 1947 at the NPL, and by **Bodewig** in 1949.

## 1.2 A Bit Word on the Jacobi

The convergence is easily monitored by using the off-norm,

$$\Omega(H) = \sqrt{\sum_{i \neq j} H_{ij}^2},$$

for which one easily shows the monotonicity

$$\Omega(H^{(k+1)}) = \Omega(H^{(k)}) - 2(H^{(k)})^2_{p_k,q_k} \leq \Omega(H^k)$$

Under suitable pivoting strategy, the sequence $(H^{(k)})_{k=0}^{\infty}$ converges to $\Lambda$ and the accumulated product $V^{(0)}V^{(1)} \cdots V^{(k)} \cdots$ converges to the orthogonal matrix $V$ of eigenvectors of $H$, $HV = V\Lambda$.

## 1.3 Convergence

Jacobi proved that the convergence of a greedy approach that annihilates the absolutely largest off-diagonal entry at each step. The greed strategy is usually replaced with the row-cyclic strategy, first used by [11], which is periodic and in one full sweep of $n(n-1)/2$ rotations it rotates row-by-row at the pivot positions $(1,2), (1,3), \ldots, (1,n); (2,3), \ldots, (2,n); \ldots; 1,n)$. Similarly, column-cyclic strategy scans the strict upper triangular of the matrix in column-by-column fashion. Forsythe and Henrici [9] proved the convergence of the serial Jacobi algorithm. [16] and [18] proved the quadratic convergence in case of the simple eigenvalues and Hari [12] extended the result to the general case of multiple eigenvalues. Mascarenhas [13] and Rhee Hari [14] showed that certain modification of row-cyclic strategy achieves cubic convergence. Rutishauser [15] described detailed implementation of the method for real symmetric matrices.

Goal: **Mathematical software implementing the new algorithm should be numerically sound and competitive in efficiency with the LAPACK's implementations of the QR and the divide-and-conquer algorithms, or any other bidiagonalization based procedure.**

Previous Works

- Jacobi method is more accurate than QR. [3]
- Computing the Singular and the Generalized Singular Values. [4]
- Implementation of Jacobi rotations for accurate singular value computation in floating point arithmetic. [5]
- A posteriori computation of the singular vectors in a preconditioned Jacobi SVD algorithm [8]

The development is divided into the following

(i) Substantial modifications of the classical Jacobi SVD algorithm are necessary to reduce its complexity. Improve the convergence by preconditioner. Seek for zero and almost diagonal situation.

(ii) The design of the algorithm should be open for further improvements. It should be based on building blocks which can benefit from the development of basic matrix computational routines (BLAS, LAPACK etc.) and blocked versions of Jacobi rotations, but without trading the numerical accuracy.

The structure of the paper:

- Section 2: Quick introduction to the numerical analysis of the symmetric definite eigenvalue problem and the SVD.
- Section 3: Description of the preconditioning.
- Section 4: Discuss whether put $A^T$ or $A$ into the algorithm.
- Section 5: Basic structure of the new Jacobi SVD problem.

# 2 Accuracy and Backward Stability in SVD Computation

Let $H$ be $n \times n$ symmetric positive definite matrix. An algorithm that computes the singular values $\widetilde{\sigma}_1 \geq \cdots \geq \widetilde{\sigma}_n$ of a nearby matrix $A + \delta A$, where $\|\delta A\| \leq \epsilon A$ with some small $\epsilon$. Weyl's theorem implies

$$\max_{i=1:n} \frac{|\widetilde{\sigma}_i - \sigma_i|}{\|A\|} \leq \frac{\|\delta A\|}{\|A\|} \leq \epsilon,$$

and in the case of full column rank $A$, $A + \delta A = (I + \delta A A^\dagger)A$ yields the bound

$$\max_{i=1:n} \frac{|\widetilde{\sigma}_i - \sigma_i|}{\sigma_i} \leq \left\|\delta A A^\dagger\right\|. \tag{2.1}$$

In general, $\delta A$ is dense with no particular structure which means that in the expression $\delta A A^\dagger = \delta A V \Sigma^\dagger U^T$, large singular values of $A^\dagger = V \Sigma^\dagger U^T$ may get excited by $\delta A$.

**Example 2.1.** Bidiagonalization based SVD algorithm produces $\delta A$ for which the best general upper bound is $\|\delta A\| \leq \epsilon_B \|A\|$ and thus

$$\left\|\delta A A^{\dagger}\right\| \leq \frac{\|\delta A\|}{\|A\|}\kappa(A) \leq \epsilon_B \kappa(A), \quad \kappa(A) = \|A\| \left\|A^{\dagger}\right\|.$$

In other words, if $\sigma_1 \geq \cdots \geq \sigma_k \gg \sigma_{k+1} \geq \cdots \geq \sigma_n > 0$, then the dominant singular values $\sigma_1, \ldots, \sigma_k$ will be computed accurately in the sense that

$$\max_{i=1:k} \frac{|\widetilde{\sigma}_i - \sigma_i|}{\sigma_i} \leq \frac{\|\delta A\|}{\|A\|}\frac{\sigma_1}{\sigma_k}, \qquad \|\delta A\| \leq O(\epsilon)\|A\|,$$

but the smallest one will have the error bound

$$\frac{|\widetilde{\sigma}_n - \sigma_n|}{\sigma_n} \leq \frac{\|\delta A\|}{\|A\|}\frac{\sigma_1}{\sigma_n}.$$

Thus, if for some $i$, it holds $\sigma_i \approx \varepsilon\sigma_1 \ll \sigma_1 = \|A\|$, then we *cannot expect any correct digit in* $\widetilde{\sigma}_i$

**Why should we care about small singular values?** The question can also become, how small is small? What if we there is no clear cut such that we can make a low rank approximation, or we meet the devil's stairs,

$$\sigma_1 \geq \cdots \geq \sigma_{k_1} \gg \sigma_{k_1+1} \geq \cdots \geq \sigma_{k_2} \gg \sigma_{k_2+1} \geq \cdots \geq \sigma_{k_3} \gg \sigma_{k_3+1} \geq \cdots \gg \sigma_n$$

with $\sigma_{k_j+1}/\sigma_{k_j} \approx O(\epsilon)$.

> In many important applications the smallest singular values are really the noise excited by the uncertainty in the data and computing them to high relative accuracy is meaningless and illusory. In such cases, the Jacobi SVD algorithm has no advantage with respect to accuracy.
>
> But given the adaptivity and of the Jacobi algorithm to modern serial and parallel computing machinery, it is exciting and challenging task to improve the efficiency of the algorithm to make it competitive with bidiagonalization based algorithms in terms of speed and memory usage, even if the high relative accuracy of the smallest singular values is not an issue.

## 2.1 Basic floating point error analysis

**Fact 1.** If numerically orthogonal matrix $\widetilde{Q}$ ($\left\|\widetilde{Q}^T\widetilde{Q} - I\right\| \ll 1$) is applied to an $m \times 1$ vector in floating point arithmetic. Then $computed(\widetilde{Q}x) = \widehat{Q}(x + \delta x)$, where $\widehat{Q}$ is orthogonal matrix close to $\widetilde{Q}$, and $\|\delta x\| \leq \epsilon \|x\|, \epsilon \leq f(k)\varepsilon$. Here $k$ is the number of coordinate directions changed under the action of $\widetilde{Q}$.
**Fact 2.** If numerically orthogonal transformations $\widetilde{Q}_1, \ldots, \widetilde{Q}_p$ are applied to *disjoint parts of an vector* $x$:

$$x \mapsto y = (\widetilde{Q}_1 \oplus \cdots \oplus \widetilde{Q}_p)x, \quad x = \begin{pmatrix} x^{(1)} \\ \vdots \\ x^{(p)} \end{pmatrix}, computed(\widetilde{Q}_i x^{(i)}) = \widehat{Q}_i(x^{(i)} + \delta x^{(i)}),$$

then we have

$$\widetilde{y} \equiv computed(y) = (\widehat{Q}_1 \oplus \cdots \oplus \widehat{Q}_p)(x + \delta x), \qquad \frac{\|\delta x\|}{\|x\|} \leq \max_{i=1:p} \frac{\left\|\delta x^{(i)}\right\|}{\|x^{(i)}\|}.$$

**Fact 3.** If $\widetilde{Q}_1, \ldots, \widetilde{Q}_r$ are numerically orthogonal transformations, and if we need to compute $y = \widetilde{Q}_r \ldots \widetilde{Q}_1 x$, then the computed approximation $\widetilde{y}$ satisfies

$$\widetilde{y} = \widehat{Q}_r \ldots \widehat{Q}_1 (x + \delta x), \quad \|\delta x\| \leq ((1+\epsilon)^r - 1)\|x\|,$$

where $\epsilon$ is maximal relative backward error in application of any $\widetilde{Q}_1, \ldots, \widetilde{Q}_r$, and $\widehat{Q}_i$ is orthogonal matrix close to $\widetilde{Q}_i$.

**Proposition 2.2.** *Let the Givens or Householder QR factorisation* $A = Q \begin{pmatrix} R \\ 0 \end{pmatrix}$ *of* $A \in \mathbb{R}^{m \times n}$, $m \geq n$, *be computed in the IEEE floating point arithmetic with rounding relative error* $\varepsilon < 10^{-7}$. *Let the computed approximations of* $Q$ *and* $R$ *be* $\widetilde{Q}$ *and* $\widetilde{R}$, *respectively. Then, there exist an orthogonal matrix* $\widetilde{Q}$ *and a backward perturbation* $\delta A$ *such that* $A + \delta A = \widetilde{Q} \begin{pmatrix} \widetilde{R} \\ 0 \end{pmatrix}$, *where* $\left\|\widetilde{Q}(:,i) - \widehat{Q}(:,i)\right\|_F \leq \varepsilon_{qr}$ *and* $\|\delta A(:,i)\| \leq \varepsilon_{qr} \|A(:,i)\|$, $1 \leq i \leq n$, *hold with*

1. $\varepsilon_{qr} \leq O(mn)\varepsilon$ *for the Householder QR factorisation.*
2. $\varepsilon_{qr} \leq (1+6\varepsilon)^p - 1$ *for the Givens factorisation.*

## 2.2 Condition Number and Scaling

It is often that the matrix is composed as $A = BD$, $D = \mathrm{diag}(d_i)_{i=1}^n$, and the uncertainty is $A + \delta A = (B + \delta B)D$. In such case, $\delta A A^\dagger = \delta B B^\dagger$. Moreover, the scaling, diagonal matrix $D$, which might be ill-conditioned, does not influence the forward perturbation of the singular values in (2.1), i.e.

$$\max_{i=1:n} \frac{|\widetilde{\sigma}_i - \sigma_i|}{\sigma_i} \leq \left\|\delta A A^\dagger\right\| = \left\|\delta B B^\dagger\right\|.$$

Similarly, if $A = DB$ (might be row scaling, instead of previous column scaling), and the uncertainty is $A + \delta A = D(B + \delta B)$, then we have the bound can also reduced to $B$ and $\delta B$. Hence, no matter how $A$ is ill-conditioned, as long as the condition number of the scaled $A$ is low, we can have a sharp bound.

Therefore, after we compute the QR factorisation of an $m \times n$ full column rank matrix $A$, we can conclude that the backward perturbation satisfies

$$\left\|\delta A A^\dagger\right\| = \left\|(\delta A D^{-1})(A D^{-1})^\dagger\right\| \leq \left\|\delta A D^{-1}\right\| \left\|(A D^{-1})^\dagger\right\| \leq \sqrt{n}\epsilon_{qr} \left\|A_c^\dagger\right\|,$$

where $D = \mathrm{diag}(\|A(:,i)\|)_{i=1}^n$ and $A_c = AD^{-1}$. Moreover, we have the following proposition

**Proposition 2.3.** *Let* $A$ *and* $\widetilde{R}$ *be as in Proposition 2.2. If* $\sigma_1 \geq \cdots \geq \sigma_n > 0$ *and* $\widetilde{\sigma}_1 \geq \cdots \geq \widetilde{\sigma}_n > 0$ *are the singular values of* $A$ *and* $\widetilde{R}$, *respectively, then*

$$\max_{i=1:n} \frac{|\widetilde{\sigma}_i - \sigma|}{\sigma_i} \leq \sqrt{n}\varepsilon_{qr} \left\|A_c^\dagger\right\| \leq n\varepsilon_{qr} \min_{\substack{S=\mathrm{diag} \\ \det(S)\neq 0}} \kappa(AS).$$

*Conclusion: If $A$ is such that $\kappa(AS)$ is moderate for some diagonal matrix $S$, then floating point QR factorisation preserves all singular values – they are all safely passed to the computed triangular factor $R$.*

Also note that, $R_c = RD^{-1}$ has unit columns and $\|R_c^{-1}\| = \left\|A_c^{\dagger}\right\|$. The note can stopped here, since all we need to know is that: doing QR factorizations, as long as the scaled $A$ is not too wild, then we can safely computing the SVD of $R$, its upper triangular QR factor, instead of the SVD of $A$.

# 3 Preconditioned Jacobi SVD algorithm

In case of $m \gg n$, the QR factorization of $A$, $A = Q\begin{pmatrix} R \\ 0 \end{pmatrix}$ reduces the computational complexity of all classical SVD method.

*The big picture from Bidiagonalization SVD.* Bidiagonalization based SVD algorithms reduce $A$ or $R$ to bidiagonal form in $4mn^2 - 4n^3/3$ flops. Recent implementation of bidiagonalization substantially reduces the data transfer between main memory and cache. Thus iterative part of those algorithms runs on a bidiagonal matrix, and completes by assembling the orthogonal matrix QR factorization bidiagonalization from the bidiagonal SVD. If we add the fact that the full SVD of bidiagonal matrix can be computed very efficiently, then the picture is complete. **Efficient preprocessing reduces the problem to the one with super fast and ingenious solution. Efficient postprocessing assembles all elements of the SVD.**

Therefore, for Jacobi SVD algorithm, we can transform the problem from $m \times n$ matrix to $n \times n$. In the case of $m \gg n$, the QR factorization is an attractive preprocessor. Since the most expensive iterative part of the Jacobi SVD algorithm transforms $n$-dimensional vectors instead of $m$-dimensional one.

## 3.1 The QR factorisation as preprocessor

Using QR factorisation as efficient preprocessor for the Jacobi SVD routine is more subtle for several reasons:

1. First, the matrix $R$ is not only smaller than $A$, in case $m > n$, but it is also triangular which allows additional savings. For instance, if we partition $R$ as

$$R = \begin{pmatrix} R_{[11]} & R_{[12]} \\ 0 & R_{[22]} \end{pmatrix},$$

   where $R_{[11]}$ is $k \times k$, $k = \lfloor n/2 \rfloor$, then during the first sweep $k(k-1)/2$ rotations of the columns $R_{[11]}$ can be performed in a canonical $k$-dimensional subspace.

2. Second, since the Jacobi algorithm iterates on the full matrix, we are interested not only in preprocessing (in the sense of dimensional reduction described above), but also in preconditioning in the sense of inducing fast convergence. This raises the question of *How to use QR factorization(s) to precondition the initial matrix A?*

3. Let $R = U_R \Sigma V_R^T$ be the SVD of $R$, where $V_R$ is the (infinite) product of Jacobi rotations, and let both sets of singular vectors be required ($U$ and $V$). If $R$ is nonsingular, then $V_R = R^{-1} U_R \Sigma$. It is tempting to implement Jacobi algorithm

without accumulation of Jacobi rotations and to compute $V_R$ from triangular matrix equation using BLAS 3 operation STRSM A fast rotation for two $n \times 1$ vectors has $4n$ flops, one sweep of $n(n-1)/2$ rotations has $2n^3 - 2n^2$ BLAS 1 flops, while STRSM has $n^3$ BLAS 3 flops.

*How can the QR factorization serve as a preconditioner for better, faster, convergence of the Jacobi algorithm?* This is achieved if the factorization is computed with column pivoting of Businger and Golub [1]:

$$AP = Q \begin{pmatrix} R \\ 0 \end{pmatrix}, \quad P \text{ permutation such that } |R_{ii}| \geq \sum_{k=i}^{j} R_{kj}^2, \quad 1 \leq i < j \leq n. \tag{3.1}$$

Now, note that SVD of $R$ is implicit diagonalization of the matrix $R^T R$, and apply one-step of Rutishauser LR diagonalization method $R^T R \implies R R^T$ (may be like QR algorithm?), which has a nontrivial diagonalizing effect.

This means, $RR^T$ is closer to diagonal form than $R^T R$. Note that

$$\begin{pmatrix} RR^T & 0 \\ 0 & 0 \end{pmatrix} = Q^T (AA^T) Q, \qquad R^T R = P^T (A^T A) P,$$

these two orthogonal similarities are substantially different.

In terms of the Jacobi algorithm, it will translates to: *One-sided Jacobi SVD algorithm on $R^T$ should have better convergence than applied to $R$.* Therefore, by Veselić and Hari [17, 1989, p. 631], we have the following preconditioning method: Given a general $A \in \mathbb{R}^{m \times n}$,

$$AP = Q \begin{pmatrix} R \\ 0 \end{pmatrix}, \qquad R^T = Q_1 R_1, \qquad R_1^T P_1 = Q_2 R_2.$$

Moreover, it is known that *simply sorting the columns of $A$ in non-increasing order (in Euclidean norm) improves the convergence.* The best elaborated pivoting is the one by de Rijk [2, Sec. 2.4].

The following proposition shows how the pivoting (3.1) influences the condition number of the row-scaled matrix $R$.

**Proposition 3.1.** *Let $R$ be a nonsingular upper triangular matrix and let (3.1) hold. Let $R = \Delta_R R_r$, where $\Delta_R$ is the diagonal matrix of the Euclidean lengths of the rows of $R$, and let $R = R_c D_R$, $D_R = \operatorname{diag}(\|R(:,i)\|_2)$. Then*

$$\begin{aligned} \left\| |R_r^{-1}| \right\|_2 &\leq \sqrt{n} + \max_{i<j} \frac{(\Delta_R)_{jj}}{(D_R)_{ii}} \cdot \left\| |R_c^{-1}| - \operatorname{diag}(R_c^{-1}) \right\|_2 \\ &\leq \sqrt{n} \left( 1 + \max_{i<j} \frac{|R_{jj}|}{|R_{ii}|} \cdot \left\| |R_c^{-1}| - \operatorname{diag}(R_c^{-1}) \right\|_2 \right), \\ \left\| |R_r^{-1}| \right\|_2 &\leq \sqrt{n} \left\| |R_c^{-1}| \right\|_2 \end{aligned}$$

*where the matrix absolute value is defined element-wise. Moreover $\|R_r^{-1}\|$ is bounded by a function of $n$, independent of $A$.*

**Remark 3.2.** Our understanding of the rank revealing property of the QR factorization is different. *If upper triangular factor in the column pivoted QR factorization is written as $R = DY$ with diagonal $D$, where $D = \operatorname{diag}(|R_{ii}|)_{i=1}^n$ or $D = \operatorname{diag}(\|R(i,:)\|)_{i=1}^n, D_{ii} \geq D_{i+1,i=1}$, then determine the pivoting to minimize $\kappa(Y) = \|Y^{-1}\| \, \|Y\|$.*

Since the choice of $D$ and the diagonal dominance ensure that $\|Y\|$ is bounded by $O(n)$, the problem reduces to *Minimising* $\|Y^{-1}\|$. Gu and Eisenstat has theoretical bound for $\|Y^{-1}\|$ which is comparable to the Wilkinson's pivot growth factor $O(n^{\log_2 n/4})$ in the Gaussian elimination with complete pivoting. The bound in case of the Businger-Golub pivoting is exponential in $n$. However, in practice, $\|Y^{-1}\|_2$ is typically bounded by $O(n)$.

**Remark 3.3.** If $m \gg n$, then the cost of the RRQR dominates the overall complexity of the SVD computation. Since the pivoting can slow down the QR factorization by a significant amount. *It is reasonable that in case $m \gg n$, the computation starts with the QR factorization without pivoting just to reduce the dimension.*

# 4 Simple Question: $A$ or $A^T$?

For the SVD of $A^T$ is trivial to obtain from the SVD of $A$ and vice versa. If $A$ is $m \times n$ with $m > n$, then we prefer starting the computation with the RRQR of $A$. If $m < n$, then we choose to start with $A^T$. But what if $A$ is square non-singular $n \times n$ matrix?

Consider an extreme example: $A = DQ$, where $D$ is diagonal and $Q$ is orthogonal. In that case, working with $A$ is implicit diagonalization of $Q^T D^2 Q$, while taking $A^T$ implicitly diagonalizes diagonal matrix $D^2$. Therefore, we would like to find out *which of the matrices $A^TA$ and $AA^T$ is closer to the diagonality?* This is particularly useful when $A$ is not normal.

Due to computational constraints, we only allow to make a decision at most $O(n^2)$ flops. The complexity corresponding to *computing the diagonal entries of $H = A^TA$ and $M = AA^T$.* Notice that $M$ and $H$ are orthogonally similar, therefore $\|H\|_F = \|M\|_F$. Moreover, we can compute the values $s(H) = \sum_{i=1}^{n} h_{ii} = \mathrm{trace}(H \circ H)$ and $s(M)$. *Larger value of $s(\cdot)$ corresponding to smaller* off$(\cdot)$. However, $s(\cdot)$ in floating point computation with round-off $\varepsilon$ completely ignores diagonal entries below $\sqrt{\varepsilon}$.

# 5 The Algorithms

Now, describe the structure of the Jacobi SVD algorithm with QR factorization as preconditioner and preprocessor. We consider the one-sided Jacobi as the following "box":

$$\text{Full column rank } X \implies \boxed{\text{One-sided Jacobi}} \implies X_\infty = XV.$$

The notation $\langle\cdot\rangle$ indicate the matrix that is not computed and no information about it is stored. The algorithms are designed such that avoiding accumulation of Jacobi rotations whenever possible.

## 5.1 Computing only $\Sigma$

In Algorithm 1, we use two QR factorizations with pivoting and then apply the one-sided Jacobi SVD algorithm.

**Remark 5.1.** The pivoting in the second QRF is optional, and $P_1 = I$ works well. If the columns of $A$ are nearly orthogonal, the second QR factorization is unnecessary. Such situation is easily detected by inspecting the matrix $R$, see [7].

**Remark 5.2.** Determining the numerical rank is tricky. Since if high accuracy is required, then QR factorisation is not authorised to declare rank deficiency.

**Algorithm 1** $\sigma = \texttt{svd}(A)$

1: $(P_r A)P = \langle Q \rangle \begin{pmatrix} R \\ 0 \end{pmatrix}; \quad \rho = \mathrm{rank}(R);$
2: $R(1:\rho, 1:n)^T P_1 = \langle Q_1 \rangle R_1;$
3: $X = R_1^T; X_\infty = X\langle V_x \rangle;$
4: $\sigma_i = \|X_\infty(:,i)\|, \quad i = 1, \ldots, \rho;$
5: $\boxed{\sigma = (\sigma_1, \ldots, \sigma_p, 0, \ldots, 0).}$

## 5.2 Computing $\Sigma$ and $V$

If we need *singular values and the right singular vectors*. Direct application of right-handed Jacobi rotation to $A$ or $R$ requires the accumulated product of rotation. To avoid this explicit multiplication, we consider $R \rightsquigarrow R^T$.

**Algorithm 2** $(\sigma, V) = \texttt{svd}(A)$

1: $(P_r A)P = \langle Q \rangle \begin{pmatrix} R \\ 0 \end{pmatrix}; \quad \rho = \mathrm{rank}(R);$
2: $X = R(1:\rho, 1:n)^T; X_\infty = X\langle V_x \rangle;$
3: $\sigma_i = \|X_\infty(:,i)\|, \quad i = 1, \ldots, \rho; \quad \boxed{\sigma = (\sigma_1, \ldots, \sigma_\rho, 0, \ldots, 0)};$
4: $U_x(:,i) = \frac{1}{\sigma_i} X_\infty(:,i), \quad i = 1, \ldots, \rho; \quad \boxed{V = PU_x}$

Notice that, we only have one QR factorization appeared. In some cases, such as $\rho \ll n$, the second QR factorization is advisable.

## 5.3 Computing $\Sigma$ and $U$

If $\Sigma$ and $U$ are needed, we need to think harder. Clearly, if we apply the right handed Jacobi on $X = A$ or $X = R$, then we do not need the product of Jacobi rotations. The problem is that *in case $m \gg n$, the rotations on $A$ are too expensive and the convergence may be slow, much slower than in the case $X = R^T$.*

On the other hand, in some cases $X = A$ is the perfect choice. For instance, if $H$ is symmetric positive definite, and we have the pivoted Cholesky fractorization $P^T H P = AA^T$ with lower triangular matrix $A$, then $A^T$ has the same properties as $R$ from Proposition 3.1. Thus $AV = U\Sigma$ will be efficient Jacobi SVD and since $H = (PU)\Sigma^2(PU)^T$ is the spectral decomposition of $H$ and $V$ is not needed.

To simplify the notation, in Algorithm 3, we define for a matrix $M$ its property $\tau(M)$ to be true if $M$ is of full column rank and the right-handed Jacobi algorithm applied to $M$ converges quickly. Such as the Cholesky factor of positive definite matrix, computed with pivoting, then $\tau(A) = \text{true}$. If evaluation of $\tau(A)$ would require more than $O(mn)$ flops, or if we do not know how to judge $A$, then by definition $\tau(A) = \text{false}$.

In the case, $X = R^T$, we need the accumulated product of Jacobi rotations and the cost for only one sweep of rotations is $2n\rho(\rho - 1)$. All of this is avoid by an extra QR factorisation.

**Algorithm 3** $(\sigma, U) = \texttt{svd}(A)$

**if** $\tau(A)$ **then**
  $X = A; X_\infty = X\langle V_x\rangle;$
  $\sigma_i = \|X_\infty(:,i)\|, \quad i = 1,\dots,n, \quad \boxed{\sigma = (\sigma_1,\dots,\sigma_n}$;
**else**
  $(P_r A)P = \langle Q\rangle \begin{pmatrix} R \\ 0 \end{pmatrix}; \quad \rho = \mathrm{rank}(R);$
  **if** $\tau(R)$ **then**
    $X = R; X_\infty = X\langle V_x\rangle;$
    $\sigma_i = \|X_\infty(:,i)\|, \quad i = 1,\dots,\rho; \boxed{\sigma = (\sigma_1,\dots,\sigma_p,0,\dots,0)};$
    $U_x(:,i) = \frac{1}{\sigma_i} X_\infty(:,i), \quad i = 1,\dots,\rho; \quad \boxed{U = P_r^T Q \begin{pmatrix} U_x \\ 0_{(m-\rho)\times\rho} \end{pmatrix}};$
  **else**
    $R(1:\rho, 1:n)^T P_1 = \langle Q_1\rangle R_1;$
    $X = R_1^T; X_\infty = X\langle V_x\rangle;$
    $\sigma_i = \|X_\infty(:,i)\|, \quad i = 1,\dots,\rho; \boxed{\sigma = (\sigma_1,\dots,\sigma_p,0,\dots,0)};$
    $U_x(:,i) = \frac{1}{\sigma_i} X_\infty(:,i), \quad i = 1,\dots,\rho; \quad \boxed{U = P_r^T Q \begin{pmatrix} P_1 U_x \\ 0_{(m-\rho)\times\rho} \end{pmatrix}};$
  **end if**
**end if**

## 5.4 Computation of $U$, $\Sigma$ and $V$

Classical implementation of the Jacobi SVD algorithm transforms two matrices, one of them approaching the matrix of left singular vectors scaled by the corresponding singular values, and the second one is the accumulated product of the Jacobi rotations.

Following [8], we will try not to accumulate Jacobi rotations,and the right singular vectors will be computed a posterior from a *well-conditioned matrix equation*.

### 5.4.1 Classical computation of $V$ by accumulation

Let the Jacobi iterations stop at index $\overline{k}$ and let $\widetilde{X}_\infty = \widetilde{X}^{(\overline{k})}$. Let $\widetilde{V}$ be the computed accumulated product of Jacobi rotations used to compute $\widetilde{X}_\infty$. Row-wise backward stability implies

$$\widetilde{X}_\infty = (X + \delta X)\widehat{V}$$

where $\widehat{V}$ is orthogonal, $\left\|\widehat{V} - \widetilde{V}\right\| \le O(n\epsilon)$ and $\|\delta X(i,:)\| \le \epsilon_J \|X(i,:)\|, \ \epsilon_J \le O(n\varepsilon)$. The matrix $\widetilde{V}$ can be written as $\widetilde{V} = (I + E_0)\widehat{V}$, where $\|E_0\|$ is small. Therefore, to recover $\widehat{V}$, the best we can do is

$$X^{-1}\widetilde{X}_\infty = (I + E_1)\widetilde{V}, \quad E_1 = X^{-1}\delta X.$$

Although we do not have $\delta X$, we can let $E_1$ close to zero. To estimate $E_1$, we write $X = DY$, where $D$ is diagonal scaling, $D_{ii} = \|X(i,:)\|$, and $Y$ has unit rows in Euclidean norm. We obtain

$$\|E_1\| = \left\|Y^{-1}D^{-1}\delta X\right\| \le \left\|Y^{-1}\right\| \left\|D^{-1}\delta X\right\| \le \left\|Y^{-1}\right\| \sqrt{n}\epsilon_J \le \left\|Y^{-1}\right\| O(n^{3/2}\varepsilon).$$

Therefore, as long as the $X$'s row scaled part is well-conditioned, we can have

$$\frac{\left\|X^{-1}\widetilde{X}_\infty - \widehat{V}\right\|}{\left\|\widehat{V}\right\|} \le \|E\|$$

Finally, the matrix $\widetilde{X}_\infty$ is written as $\widetilde{U}\widetilde{\Sigma}$. The diagonal entries of $\widetilde{\Sigma}$ are computed as $\widetilde{\sigma}_i = \mathit{computed}(\left\|\widetilde{X}_\infty(:,i)\right\|) = \left\|\widetilde{X}_\infty(:,i)\right\|(1+\nu_i)$, $|\nu_i| \le O(n\varepsilon)$, and then $\widetilde{U}(:,i)$ is computed by dividing $\widetilde{X}_\infty(:,i)$ by $\widetilde{\sigma}_i$, thus

$$\widetilde{U}\widetilde{\Sigma} = \widetilde{X}_\infty + \delta\widetilde{X}_\infty, \qquad \left|\delta\widetilde{X}_\infty\right| \le 3\varepsilon\left|\widetilde{X}_\infty\right|.$$

The following proposition explains how well the computed SVD resembles the matrix $X$.

**Proposition 5.3.** *The matrix $\widetilde{U}, \widetilde{\Sigma}, \widetilde{V}$ and $\widehat{V}$ satisfy the residual relations*

$$\widetilde{U}\widetilde{\Sigma}\widehat{V}^T = X + F$$
$$\widetilde{U}\widetilde{\Sigma}\widetilde{V}^T = (X+F)(I+E_0^T),$$

*where for all $i$,*

$$\|F(i,:)\| \le (\varepsilon_J + 3\varepsilon(1+\varepsilon_J))\|X(i,:)\|,$$
$$\|E_0\| \le \sqrt{n}\varepsilon_J \le O(n^{3/2}\varepsilon),$$
$$\left\|X^{-1}F\right\| \le \left\|Y^{-1}\right\|\sqrt{n}(\varepsilon_J + 3\varepsilon(1+\varepsilon_J)).$$

### 5.4.2 Computation of $V$ from matrix equation

Suppose we decide to use an approximation of $\widehat{V}$ instead of $\widetilde{V}$. The matrix $X^{-1}\widetilde{X}_\infty$ is a good candidate, but we cannot have the exact value of $X^{-1}\widetilde{X}_\infty$. Instead, *we solve the matrix equation $\widetilde{X}_\infty = X\breve{V}$[1] and take $\breve{V} = \mathit{computed}(X^{-1}\widetilde{X}_\infty)$*

Since $X$ is triangular, the residual bound for $\breve{V}$ is

$$E_2 = X\breve{V} - \widetilde{X}_\infty, \quad |E_2| \le \epsilon_T|X|\left|\breve{V}\right|, \quad \epsilon_T \le \frac{n\varepsilon}{1-n\varepsilon}.$$

The following proposition shows that we have also computed a rank revealing decomposition of $X$,

**Proposition 5.4.** *The matrix $\widetilde{U}, \widetilde{\Sigma}, \breve{V}$ satisfy the following residual relations*

$$\widetilde{U}\widetilde{\Sigma}\breve{V}^T = (X+F)(I+E_3^T), \quad E_3 = E_1 + X^{-1}E_2\widetilde{V}^T,$$
$$\widetilde{U}\widetilde{\Sigma}\breve{V}^{-1} = X + F_1, \quad F_1 = E_2\breve{V}^{-1} + \delta\widetilde{X}_\infty\breve{V}^{-1},$$

*where $F$ is in Proposition 5.3, $\|E_3\| \le \|Y^{-1}\|(\sqrt{n}\epsilon_J + n\epsilon_T)$, and it holds for all $i$, $\|F_1(i,:)\| \le (\varepsilon_T\left\|\left|\breve{V}\right|\right\| + 3\varepsilon_J(1+\varepsilon_J))\left\|\breve{V}^{-1}\right\|\|X(i,:)\|$*

[1] This can be solved using BLAS 3 `STRSM`.

Analysis shows that the quality of the computed right singular vector matrix $\breve{V}$ depends on the condition number of $Y^{-1}$, where $X = DY$. This means, the rows of the triangular $X$ must be well-conditioned in the scaled sense. If $X$ is computed from the initial $A$ using the QR factorization with column pivoting, $AP = Q \begin{pmatrix} R \\ 0 \end{pmatrix}$, then $X = R$ can be written as $X = DY$ with well-conditioned $Y$. Thus, we expect that $\breve{V}$ can be computed accurately.

Conclusion: The initial matrix should be of the form $X = DY = ZC$ where $D, C$ are diagonal and both $Y$ and $Z$ well-conditioned. Well-conditioned $Z$ implies fast convergence, while well-conditioned $Y$ ensures a stable posterior computation of the right singular vectors. Therefore, we define $X$ in the following way:

$$AP = Q \begin{pmatrix} R \\ 0 \end{pmatrix}; \quad R^T P_1 = Q_1 R_1; \quad X = R_1^T.$$

# 6 Assessing the accuracy of the computed SVD (Skip for now)

# References

[1] Peter Businger and Gene H. Golub. Linear least squares solutions by householder transformations. *Numerische Mathematik*, 7(3):269–276, 1965. (Cited on p. 7.)

[2] P. P. M. de Rijk. A one-sided Jacobi algorithm for computing the singular value decomposition on a vector computer. *SIAM Journal on Scientific and Statistical Computing*, 10(2):359–371, 1989. (Cited on p. 7.)

[3] James Demmel and Krešimir Veselić. Jacobi's method is more accurate than QR. *SIAM Journal on Matrix Analysis and Applications*, 13(4):1204–1245, 1992. (Cited on p. 3.)

[4] Zlatko Drmač. *Computing the Singular and the Generalized Singular Values*. Phd thesis, Lehrgebiet Mathematische Physik, Fernuniversität Hagen, September 1994. (Cited on p. 3.)

[5] Zlatko Drmač. Implementation of jacobi rotations for accurate singular value computation in floating point arithmetic. *SIAM Journal on Scientific Computing*, 18(4): 1200–1222, 1997. (Cited on p. 3.)

[6] Zlatko Drmač and Krešimir Veselić. New fast and accurate Jacobi SVD algorithm. I. *SIAM Journal on Matrix Analysis and Applications*, 29(4):1322–1342, 2008. (Cited on p. 1.)

[7] Zlatko Drmač and Krešimir Veselić. New fast and accurate Jacobi SVD algorithm. II. *SIAM Journal on Matrix Analysis and Applications*, 29(4):1343–1362, 2008. (Cited on p. 8.)

[8] Z Drmač. A posteriori computation of the singular vectors in a preconditioned jacobi svd algorithm. *IMA Journal of Numerical Analysis*, 19(2):191–213, 1999. (Cited on pp. 3 and 10.)

[9] G. E. Forsythe and P. Henrici. The cyclic Jacobi method for computing the principal values of a complex matrix. *Transactions of the American Mathematical Society*, 94 (1):1–23, 1960. Publisher: American Mathematical Society. (Cited on p. 2.)

[10] H. H. Goldstine, F. J. Murray, and J. von Neumann. The Jacobi method for real symmetric matrices. *Journal of the ACM*, 6(1):59–96, 1959. (Cited on p. 2.)

[11] Robert T. Gregory. Computing eigenvalues and eigenvectors of a symmetric matrix on the ILLIAC. *Mathematics of Computation*, 7(44):215–220, 1953. (Cited on p. 2.)

[12] Vjeran Hari. On sharp quadratic convergence bounds for the serial Jacobi methods. *Numerische Mathematik*, 60(1):375–406, 1991. (Cited on p. 2.)

[13] Walter F. Mascarenhas. On the convergence of the Jacobi method for arbitrary orderings. *SIAM Journal on Matrix Analysis and Applications*, 16(4):1197–1209, 1995. (Cited on p. 2.)

[14] Noah H. Rhee and Vjeran Hari. On the global and cubic convergence of a quasi-cyclic jacobi method. *Numerische Mathematik*, 66(1):97–122, 1993. (Cited on p. 2.)

[15] H. Rutishauser. The Jacobi method for real symmetric matrices. *Numerische Mathematik*, 9(1):1–10, 1966. (Cited on p. 2.)

[16] A Schönhage. Zur konvergenz des Jacobi-verfahrens. *Numerische Mathematik*, 3(1): 374–380, 1961. (Cited on p. 2.)

[17] K. Veselić and V. Hari. A note on a one-sided Jacobi algorithm. *Numerische Mathematik*, 56(6):627–633, 1989. (Cited on p. 7.)

[18] J. H. Wilkinson. Note on the quadratic convergence of the cyclic Jacobi process. *Numerische Mathematik*, 4(1):296–300, 1962. (Cited on p. 2.)