Unit ~~7~~ 11: Computing the SVD

Linking the Singular Value Decomposition to the Spectral Decomposition

In week 2, we know that: with matrix $A \in \mathbb{C}^{m\times n}$:

$$\boxed{A = U \Sigma V^H}$$

where
$$\begin{cases} U \in \mathbb{C}^{m\times m}, \quad \Sigma \in \mathbb{C}^{m\times n}, \quad V \in \mathbb{C}^{n\times n} \\ \Sigma = \left(\begin{array}{c|c} \Sigma_{TL} & 0_{r\times(n-r)} \\ \hline 0_{(m-r)\times r} & 0_{(m-r)\times(n-r)} \end{array}\right), \text{ with } \Sigma_{TL} = \operatorname{diag}(\sigma_0, \dots \sigma_{r-1}) \\ \text{and } \sigma_0 \geq \dots \geq \sigma_{r-1} > 0 \end{cases}$$

And if we partition $U = (U_L \mid U_R)$ and $V = (V_L \mid V_R)$, then:

$$\boxed{A = U_L \Sigma_{TL} V_L^H}$$

However, we did not present the pratical algorithms for computing SVD. Now with the knowledge from the QR algorithm, we can now revisit the discussion.

In week ~~10~~, we discovered algorithms for computing the Spectral Decomposition of Hermitian matrix. These ~~hom~~ homeworks will give us hint to the link between SVD of A to the Spectral Decomposition of $B = A^H A$

Hw 11.1.1.1

Let $A \in \mathbb{C}^{m\times n}$ and $A = U\Sigma V^H$ its SVD, the Spectral Decomposition of the matrix $A^H A$ is given by:

$$A^H A = V \begin{pmatrix} \Sigma_{TL}^2 & 0 \\ 0 & 0 \end{pmatrix} V^H$$

Hw 11.1.1.2

Same conditions as hw 11.1.1.1, the Spectral Decomposition of the matrix AA^H is given by:

$$U \begin{pmatrix} \Sigma_{TL}^2 & 0 \\ 0 & 0 \end{pmatrix} U^H.$$

With the hints from these hw, we now discover how to compute SVD of A from Spectral Decomposition of $A^H A$ or AA^H.

Computing SVD from the Spectral Decomposition

Hw 11.2.1.1 When A is square

Let $A \in \mathbb{C}^{m\times m}$ be non singular, and $A^H A = QDQ^H$. Give formula for U, V, Σ so that $A = U\Sigma V^H$.

$$\begin{cases} V = Q \\ \Sigma = D^{1/2} \\ U = AQD^{-1/2} \end{cases}$$

Hw 11.2.1.2 When A is not square

Let $A \in \mathbb{C}^{m\times n}$ have full column rank, and $A^H A = QDQ^H$. Give formula for U, V, Σ so that $A = U\Sigma V^H$

$$\begin{cases} V = Q \\ \Sigma = D^{1/2} \\ U_L = AQD^{-1/2} \end{cases}$$

So far, the algorithm we have to ~~turn SVD into~~ compute SVD from Spectral Decomposition is:

- Form $A^H A$
- Compute Spectral Decomposition of $A^H A = \cancel{V \Sigma^2 V^H}$ $Q D Q^H$
- Compute $\begin{cases} V = Q \\ \Sigma = D^{1/2} \\ U = A Q D^{-1/2} \end{cases}$

However, there are problems with this algorithm, specifically:

- Condition number (form $A^H A$ will square the condition number)
- Numerical stability (is the algorithm implemented stable?)

Recall that we try to avoid using Method of Normal Equations to solve LLS problem when the matrix is ill-conditioned.
Similarly, we avoid computing SVD from $A^H A$

Thm 11.2.1.3 Formula for reduced SVD of A

$$\begin{cases} V_L = Q_L \\ \Sigma_{TL} = D_{TL}^{1/2} \\ U_L = A Q_{TL} D_{TL}^{-1/2} \end{cases}$$

Thm 11.2.1.4 Compute SVD through Spectral Decomposition

$$A = \begin{pmatrix} \sqrt{2} & 1 \\ 0 & \sqrt{2} \end{pmatrix}$$

A Strategy for computing the SVD

In practice, we usually work with matrix that is tall and skinny, so the reduced SVD is desired.
The method we are discussing to compute SVD is based of the QR implicit shifted algorithm, which costs $O(n^3)$.
However, the constant term of this algorithm is much larger in comparison to QR factorization.
Therefore, we won't modify the algorithm directly. But will:

- Compute QR factorization of A
- Find SVD of R

Thm 11.2.2.1 Reduced SVD of A can be extrated from Q and SVD of R

Let $A \in \mathbb{C}^{m \times n}$, $m \geq n$, and $A = QR$
Assume $R \in \mathbb{C}^{n \times n}$ is nonsingular, $R = \hat{U} \hat{\Sigma} \hat{V}^H$
Then the reduced SVD of A is:

$$A = \underbrace{(Q\hat{U})}_{U_L} \underbrace{\hat{\Sigma}}_{\Sigma_{TL}} \underbrace{\hat{V}^H}_{V_L^H}$$

Observations:-

- In 10.3.1, we know that Hermitian matrix A can be reduced to tridiagonal form via a banch of House holder transformations, this reduce the cost of QR algorithm. In the next unit, we will see that matrix A can also be reduced to bidiagonal form. In other words:

$$A = Q_A B Q_A^H \quad \text{where} \begin{cases} \cdot\ B \text{ is bidiagonal, real-val} \\ \cdot\ Q_A = H_{m-2} \cdots H_0 \end{cases}$$

- If we form $T = B^T B$ then since T is tridiagonal, we can employ the implicit shifted QR algorithm to find its Spectral Decomposition and from that construct SVD of B.

- However, same reason as that we don't form $A^T A$, we don't form $B^T B$ since it will square the condition number.

- In the next units, we'll find that we can employ the Implicit Q Theorem to compute the SVD of B, inspired by the implicit shifted QR algorithm.

Reduction to bidiagonal form

Thr 11.2.3.1 Product of bidiagonal matrices is a tridiagonal matrix

$$B = \begin{pmatrix} \beta_{0,0} & \beta_{0,1} & & \\ & \beta_{1,1} & \ddots & \\ & & \ddots & \beta_{m-2,m-1} \\ & & & \beta_{m-1,m-1} \end{pmatrix}$$

$T = B^T B$ is a tridiagonal matrix

Given we can preprocess our problem by computing its QR factorization, we now focus on the case where $A \in \mathbb{C}^{m \times m}$. The next step is reduce A to bidiagonal.

Algorithm to reduce square matrix to bidiagonal form:-

- Partition $A \rightarrow \begin{pmatrix} \alpha_{11} & a_{12}^T \\ a_{21} & A_{22} \end{pmatrix}$
- Update $\left[\begin{pmatrix} \alpha_{11} \\ a_{21} \end{pmatrix}, \tau_1 \right] := \text{Housev}\left(\begin{pmatrix} \alpha_{11} \\ a_{21} \end{pmatrix} \right)$.

  This overwrites α_{11} with $\pm \left\| \begin{pmatrix} \alpha_{11} \\ a_{21} \end{pmatrix} \right\|_2$ and a_{21} with u_{21}.
  Implicitly, a_{21} in the updated matrix equals zero vector.
- Update $\begin{pmatrix} a_{12}^T \\ A_{22} \end{pmatrix} := H\left(\begin{pmatrix} 1 \\ u_{21} \end{pmatrix}, \tau_1 \right) \begin{pmatrix} a_{12}^T \\ A_{22} \end{pmatrix}$

  (update first column)

Algorithm to reduce square matrix to bidiagonal form (continue)

- The matrix is $A \rightarrow \begin{pmatrix} \alpha_{11} & a_{12}^T \\ 0 & A_{22} \end{pmatrix}$, where the zeroes have been overwritten with u_{21}
- ~~let~~ Compute $[u_{12}, \tau_1] := \text{Houseν1}\left((a_{12}^T)^T \right)$. The first element of u_{12} now holds $\pm \| (a_{12}^T)^T \|_2$ and the rest of vector u_{12}^T ~~that~~ defines the Householder transformation, now overwrites a_{12}^T
- After setting the first entry of u_{12} explicitly to one, we update $A_{22} := A_{22} H(u_{12}, \rho_1)$

$$\begin{pmatrix} \times & \times & \times & \times \\ 0 & \times & \times & \times \\ 0 & \times & \times & \times \\ 0 & \times & \times & \times \end{pmatrix} \longrightarrow \begin{pmatrix} \times & \times & 0 & 0 \\ 0 & \times & \times & \times \\ 0 & \times & \times & \times \\ 0 & \times & \times & \times \end{pmatrix}$$

(Update first row)

Continue with these steps, we will end up with a matrix in the form of:

$$\begin{pmatrix} \times & \times & & \\ & \times & \times & \\ & & \times & \times \\ & & & \times \end{pmatrix} \rightarrow \text{bidiagonal matrix}$$

Implicitly shifted bidiagonal QR algorithm

Recall from the shifted ~~b~~ QR algorithm, we would:

- Form $T^{(k)} = B^{(k)T} B^{(k)}$ (B is 4×4 matrix)
- Compute first Givens' rotation s.t:

$$\underbrace{\begin{pmatrix} \gamma_0 & -\sigma_0 \\ \sigma_0 & \gamma_0 \end{pmatrix}^T}_{G_0^T} \begin{pmatrix} \tau_{0,0} - \tau_{3,3} \\ \tau_{1,0} \end{pmatrix} = \begin{pmatrix} \times \\ 0 \end{pmatrix}$$

- Since we introduce a ~~bud~~ bulge in the last step, now we chase the bulge out with set of Givens' rotations:

$$\cancel{T^{(k+1)} = G_{T_2}^T G_{T_1}^T G_{T_0}^T T_1}$$

$$T^{(k+1)} = \begin{pmatrix} I & \\ & G_{T_2}^T \end{pmatrix} \begin{pmatrix} 1 & & \\ & G_{T_1}^T & \\ & & 1 \end{pmatrix} \begin{pmatrix} G_{T_0}^T & \\ & I \end{pmatrix} T^{(k)} \begin{pmatrix} G_{T_0} & \\ & I \end{pmatrix} \begin{pmatrix} 1 & & \\ & G_{T_1} & \\ & & 1 \end{pmatrix} \begin{pmatrix} I & \\ & G_{T_2}^T \end{pmatrix}$$

Now, instead of applying on T, we apply on $B^T B$, which would resulted in:

$$(B^T B)^{(k+1)} = \left[B^{(k)} \begin{pmatrix} G_{T_0} & \\ & I \end{pmatrix} \begin{pmatrix} 1 & & \\ & G_{T_1} & \\ & & 1 \end{pmatrix} \begin{pmatrix} I & \\ & G_{T_2} \end{pmatrix} \right]^T$$
$$\times \left[B^{(k)} \begin{pmatrix} G_{T_0} & \\ & I \end{pmatrix} \begin{pmatrix} 1 & & \\ & G_{T_1} & \\ & & 1 \end{pmatrix} \begin{pmatrix} I & \\ & G_{T_2} \end{pmatrix} \right]$$

This insight show us that, if we can find 2 sequences of Givens' rotations s.t:

$$B^{(k+1)} = \underbrace{\begin{pmatrix} I & \\ & \hat{G}_2^T \end{pmatrix} \begin{pmatrix} 1 & & \\ & \hat{G}_1^T & \\ & & 1 \end{pmatrix} \begin{pmatrix} \hat{G}_0^T & \\ & I \end{pmatrix}}_{\tilde{Q}^T} \times B^{(k)} \times \underbrace{\begin{pmatrix} \tilde{G}_0 & \\ & I \end{pmatrix} \begin{pmatrix} 1 & & \\ & \tilde{G}_1 & \\ & & 1 \end{pmatrix} \begin{pmatrix} I & \\ & \tilde{G}_{T_2} \end{pmatrix}}_{Q}$$

↳ this will disappear

Then, by Implicit Q Theorem

$$B^{(k+1)T} B^{(k+1)} = \begin{pmatrix} \times & \times & & \\ & \times & \times & \\ & & \times & \times \\ & & & \times \end{pmatrix}^T \begin{pmatrix} \times & \times & & \\ & \times & \times & \\ & & \times & \times \\ & & & \times \end{pmatrix}$$

$$= \begin{pmatrix} \times & \times & & \\ \times & \times & \times & \\ & \times & \times & \times \\ & & \times & \times \end{pmatrix}$$

$$= T^{(k+1)} = Q^T T^{(k)} Q$$

If we continue to iterate, then:
- $T^{(k)}$ converge to diagonal matrix
- $B^{(k)}$ converge to diagonal matrix Σ_B
- Givens rotation accumulates to U_B and V_B

$$\Rightarrow \boxed{B = U_B \Sigma_B V_B^T}$$

SVD of matrix B

How to find 2 sequences of Givens Rotations mentioned?
- Compute G_0 with a shift, the shift is the bottom right element of matrix $T = B^T B$
- Apply G_0 to the right $B^{(k)}$, introduce a bulge
- Compute $\hat{G}_0^T$, apply to the left of $B^{(k)}$, changes the nonzero (the bulge) that was introduced back to zero
- Continue to chase the bulge out with $\tilde{G}_1, \hat{G}_1^T, \ldots$

Implicit Shifted Bidiagonal QR algorithm

Jacobi Rotation

Given matrix $A = \begin{pmatrix} \alpha_{0,0} & \alpha_{0,1} \\ \alpha_{1,0} & \alpha_{1,1} \end{pmatrix}$

There exists a rotation $J = \begin{pmatrix} \gamma & \sigma \\ \sigma & \gamma \end{pmatrix}$ such that:

$$J^T A J = \begin{pmatrix} \lambda_0 & 0 \\ 0 & \lambda_1 \end{pmatrix} = \Lambda$$

$\Rightarrow A = J \Lambda J^T$ is the Spectral Decomposition of A. The columns of J are eigenvectors of length one and the diagonal elements of Λ are eigenvalues.

How to find J ?
- Form the charateristic polynomial
$$\det(\lambda I - A) = (\lambda - \alpha_{0,0})(\lambda - \alpha_{1,1}) - \alpha_{1,0}^2$$
$$= \lambda^2 - (\alpha_{0,0} + \alpha_{1,1})\lambda + (\alpha_{0,0}\alpha_{1,1} - \alpha_{1,0}^2)$$
- Solve for eigenvalues
- Find associated eigenvectors, and scale it to have unit length and lie in Quadrant I or Quadrant II :

$$\begin{cases} \text{Quadrant I} : j_1 = \begin{pmatrix} \gamma \\ \sigma \end{pmatrix} \\ \text{Quadrant II} : j_2 = \begin{pmatrix} -\sigma \\ \gamma \end{pmatrix} \end{cases}$$

Jacobi's method for computing the Spectral Decomposition

Jacobi's original idea went as follows:

- Start with symmetric matrix $A = \begin{pmatrix} \alpha_{0,0} & \cdots & \alpha_{0,3} \\ \vdots & \ddots & \vdots \\ \alpha_{3,0} & \cdots & \alpha_{3,3} \end{pmatrix}$
- Find off-diagonal entry with largest magnitude, let say its $\alpha_{3,1}$
- Compute a Jacobi rotation so that:

$$\begin{pmatrix} \gamma_{3,1} & \sigma_{3,1} \\ -\sigma_{3,1} & \gamma_{3,1} \end{pmatrix} \begin{pmatrix} \alpha_{1,1} & \alpha_{1,3} \\ \alpha_{3,1} & \alpha_{3,3} \end{pmatrix} \begin{pmatrix} \gamma_{3,1} & -\sigma_{3,1} \\ \sigma_{3,1} & \gamma_{3,1} \end{pmatrix} = \begin{pmatrix} X & 0 \\ 0 & X \end{pmatrix}$$

- Apply rotations to A:

$$\begin{pmatrix} \alpha_{0,0} & X & \alpha_{0,2} & X \\ X & X & X & 0 \\ \alpha_{2,0} & X & \alpha_{2,2} & X \\ X & 0 & X & X \end{pmatrix} = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & \gamma_{3,1} & 0 & \sigma_{3,1} \\ 0 & 0 & 1 & 0 \\ 0 & -\sigma_{3,1} & 0 & \gamma_{3,1} \end{pmatrix} A \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & \gamma_{3,1} & 0 & -\sigma_{3,1} \\ 0 & 0 & 1 & 0 \\ 0 & \sigma_{3,1} & 0 & \gamma_{3,1} \end{pmatrix}$$

- This process repeats, reducing off-diagonal element that is largest in magnitude to zero in each iteration.
- Eventually, all the off-diagonal becomes zero which give us the diagonal matrix Λ with accumulated ~~ac~~ matrix J with eigenvectors as its columns.

Key insight: Applying Jacobi rotation to zero an element, α_{ij} reduces the square of Frobenius norm of the off-diagonal elements of the matrix by α_{ij}^2.

In other words, let off (A) equals matrix A but with diagonal elements set to zero. If J_{ij} zeroes out α_{ij} (and α_{ji}), then:

$$\| \text{off}(J_{ij}^T A J_{ij}) \|_F^2 = \| \text{off}(A) \|_F^2 - 2\alpha_{ij}^2$$

This means 2 things:
- Every time a Jacobi rotation is applied, off(A) ↘ by twice the square of that element.
- A previously zeroed elements may become nozero in the process

The cost of this algorithm is:
- Searching for the largest off-diagonal element: $O(m^2)$
- Compute and apply Jacobi rotation: $O(m)$

For large m this is not pratical.

A pratical algorithm zero out the off-diagonal elements by columns (or rows), one pair at a time.

This is the column-cyclic Jacobi algorithm.

Jacobi's method for computing the SVD

Key insight:
We know that applying a bunch of Jacobi rotations to the both sides of A^TA can diagonalize it:

$$\cancel{A} \underbrace{\cdots J_{3,1}^T J_{3,1}^T}_{Q^T} \cancel{A}B \underbrace{J_{3,1} J_{3,1} \cdots}_{Q} = D$$

where $B = A^TA$

Recall that after permutate columns of Q and diagonal elements of D carefully, then choosing $V = Q$ and $\Sigma = D^{1/2}$ yields:

$$A = U \Sigma V^T = U D^{1/2} Q^T$$

$$\Rightarrow AQ = U\Sigma = UD^{1/2}$$

This means that if we apply Jacobi rotations $J_{2,1}, J_{3,1}, \dots$ from the right to A

$$UD^{1/2} = \underbrace{((A J_{2,1}) J_{3,1}) \dots}_{\hat{A}}$$

then once B become diagonal, the columns of $\hat{A}$ are mutually orthogonal. By scaling them to have length one, setting $\Sigma = \operatorname{diag}(\|\hat{a}_0\|_2, \|\hat{a}_1\|_2, \dots, \|\hat{a}_{n-1}\|_2)$, we find that:

$$U = \hat{A} \Sigma^{-1} = AQ(D^{1/2})^{-1}.$$

The only problem is by forming B, we squares the condition number.

A more practical algorithm is as follows:

- Start with A, compute a sequence of Jacobi rotations until the off-diagonal elements of A^TA become small. Every time a Jacobi rotation is computed, it updates appropriate colums of A
- Accumulate the Jacobi rotations into matrix V:

$$V = ((I \times J_{2,1}) J_{3,1}) \dots$$

- Upon completion, $\Sigma = \operatorname{diag}(\|a_0\|_2, \|a_1\|_2, \dots \|a_{n-1}\|_2)$ and $U = A\Sigma^{-1}$ (each columns of A is divided by its length)