Lecture 19: SVD Versus Eigendecomposition

July 23, 2025

Outline

1. Alternative Formulation
    1. Alternate Approach Example
2. Proof of Existence of SVD
3. Stability Comparison
4. Golub-Kahan Bidiagonalization

Introduction

- Recall that SVD is the decomposition of any matrix A into $U\Sigma V^T$, where Σ is diagonal with non-negative entries, and U, V are orthogonal.
- In the previous lecture we seen that the SVD can be found from the eigendecomposition of A^TA or AA^T.
- In this lecture we will see a more stable method using the eigendecomposition of

$$H = \begin{bmatrix} 0 & A^T \\ A & 0 \end{bmatrix}.$$

- We will also prove the existence of the SVD, discuss stability, and discuss how to compute the SVD efficiently.

Alternative Formulation

- Assume A is square, i.e. $A \in \mathbb{R}^{n \times n}$.
- Consider the $2n \times 2n$ symmetric matrix

$$H = \begin{bmatrix} 0 & A^T \\ A & 0 \end{bmatrix}.$$

- By computing the eigendecomposition of $H = Q \Lambda Q^T$ we can extract the singular values and vectors.
- We have that $\sigma_A = |\lambda_H|$, and U, V can be recovered from the eigenvectors.
- Let us see why all of this holds.

Alternative Formulation

- Write $A = U\Sigma V^T$, then we have $AV = U\Sigma$ because V is orthogonal.
- Likewise,

$$\begin{aligned} A^T &= (U\Sigma V^T)^T \\ &= V\Sigma^T U^T \\ &= V\Sigma U^T, \text{ because } \Sigma \text{ is diagonal.} \end{aligned}$$

- Thus $A^T U = V\Sigma$ because U is orthogonal.

Alternative Formulation

- Hence we have

$$\underbrace{\begin{bmatrix} 0 & A^T \\ A & 0 \end{bmatrix}}_{H} \underbrace{\begin{bmatrix} V & V \\ U & -U \end{bmatrix}}_{Q} = \begin{bmatrix} A^T U & -A^T U \\ AV & AV \end{bmatrix}$$

$$= \begin{bmatrix} V\Sigma & -V\Sigma \\ U\Sigma & U\Sigma \end{bmatrix}$$

$$= \underbrace{\begin{bmatrix} V & V \\ U & -U \end{bmatrix}}_{Q} \underbrace{\begin{bmatrix} \Sigma & 0 \\ 0 & -\Sigma \end{bmatrix}}_{\Lambda}$$

- Therefore, $HQ = Q\Lambda$, equivalently $H = Q\Lambda Q^T$, gives an **eigendecomposition** of H.
- Note, we need to normalize the columns of Q, to make Q an orthogonal matrix.
- See the next slide for the explanation of why the columns of Q constructed as above are orthogonal.

Alternative Formulation

Explanation Of Why Q's Columns Are Orthogonal, Given U, V Are Orthogonal Matrices

- Let $1 \leq i < j \leq n$ be arbitrary, with $j \neq n + i$.
- Then we have

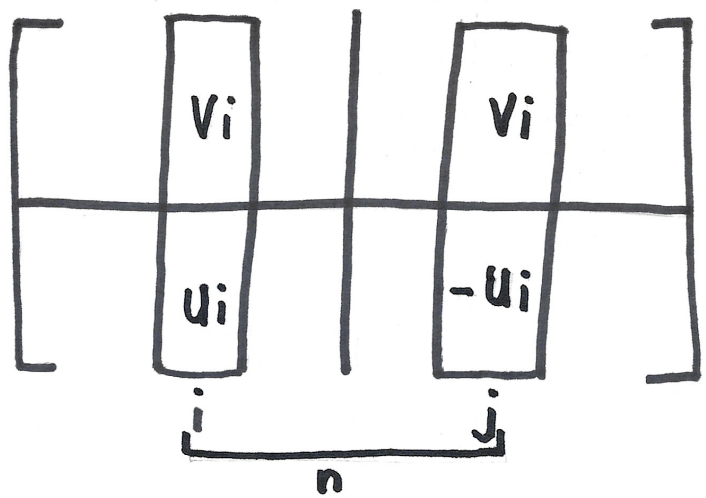
$$
\begin{aligned}
& \begin{bmatrix} v_i \\ u_i \end{bmatrix}^T \begin{bmatrix} v_j \\ \pm u_j \end{bmatrix} \\
= \quad & \begin{bmatrix} v_i^T & u_i^T \end{bmatrix} \begin{bmatrix} v_j \\ \pm u_j \end{bmatrix} \\
= \quad & v_i^T v_j \pm u_i^T u_j \\
\underbrace{=}_{U,V \text{ are orthogonal, } j \neq n+i} \quad & 0 \pm 0 \\
= \quad & 0.
\end{aligned}
$$

Alternative Formulation

Explanation Of Why Q's Columns Are Orthogonal, Given U, V Are Orthogonal Matrices

- The only case not covered by the above argument is $1 \leq i \leq n$, with $j = n + i$.
- Then we get

$$
\begin{aligned}
& v_i^T v_j \pm u_i^T u_j \\
= \; & v_i^T v_i \underbrace{-}_{\substack{2^{nd} \; u_i \text{ comes from } -U \text{ block}}} u_i^T u_i \\
= \; & 1 - 1 \\
= \; & 0.
\end{aligned}
$$



Alternative Formulation

To summarize, we have the following steps:

1. Form $H = \begin{bmatrix} 0 & A^T \\ A & 0 \end{bmatrix}$,
2. Compute eigendecomposition $HQ = Q\Lambda$,
3. Set $\sigma_A = |\lambda_H|$,
4. Extract U, V from Q (normalizing for orthogonality).

A Few Words About The Non-Square Case

- Let A be $m \times n$, where $m > n$.
- Then a reduced SVD for A has the form

$$\underbrace{A}_{m\times n} = \underbrace{\hat{U}}_{m\times n}\underbrace{\Sigma}_{n\times n}\underbrace{V^T}_{n\times n}.$$

- Then defining H as above makes H $(m+n) \times (m+n)$.
- Since H is still square, therefore the same setup works from this point onwards.

Alternative Formulation

- This algorithm is preferable with respect to stability (see the more detailed section below).
- The error in the singular values satisfies $|\tilde{\sigma}_k - \sigma_k| = O(\epsilon \|A\|)$, compared to $O\left(\epsilon \|A\|^2 / \sigma_k\right)$ for the algorithm using $A^T A$.
- This approach can be extended to non-square matrices too.
- Practical algorithms are based on this premise, but without explicitly forming the (large) matrix H.

Alternative Formulation - Alternate Approach Example

$$A = \begin{bmatrix} 0 & -\frac{1}{2} \\ 3 & 0 \end{bmatrix}$$

Therefore

$$H = \begin{bmatrix} 0 & A^T \\ A & 0 \end{bmatrix} = \begin{bmatrix} 0 & 0 & 0 & 3 \\ 0 & 0 & -\frac{1}{2} & 0 \\ 0 & -\frac{1}{2} & 0 & 0 \\ 3 & 0 & 0 & 0 \end{bmatrix}$$

Alternative Formulation - Alternate Approach Example

MATLAB eigendecomposition gives

$$Q = \frac{1}{\sqrt{2}} \begin{bmatrix} 1 & 0 & 0 & 1 \\ 0 & 1 & 1 & 0 \\ 0 & 1 & -1 & 0 \\ -1 & 0 & 0 & 1 \end{bmatrix}, \Lambda = \begin{bmatrix} -3 & & & \\ & -\frac{1}{2} & & \\ & & \boxed{\frac{1}{2}} & \\ & & & \boxed{3} \end{bmatrix}$$

Order may be different (of cols) so read off desired cols, for positive Σ entries.

Alternative Formulation - Alternate Approach Example

One can verify that the following eigendecomposition (permuting the columns of the previous one, to change the order of the eigenvalues) is also correct, and perfectly fits the shape required of our setup:

$$Q = \frac{1}{\sqrt{2}} \begin{bmatrix} 1 & 0 & 1 & 0 \\ 0 & 1 & 0 & 1 \\ 0 & -1 & 0 & 1 \\ 1 & 0 & -1 & 0 \end{bmatrix}, \Lambda = \begin{bmatrix} \boxed{3} & & & \\ & \boxed{\frac{1}{2}} & & \\ & & -3 & \\ & & & -\frac{1}{2} \end{bmatrix}$$

Even better, this modified eigendecomposition fits in perfectly with the remainder of the computation.

Alternative Formulation - Alternate Approach Example

Therefore

$$
\begin{aligned}
\sigma_1 &= 3 \\
v_1 &= \begin{bmatrix} 1 \\ 0 \end{bmatrix} \\
u_1 &= \begin{bmatrix} 0 \\ 1 \end{bmatrix} \\
\sigma_2 &= \frac{1}{2} \\
v_2 &= \begin{bmatrix} 0 \\ 1 \end{bmatrix} \\
u_2 &= \begin{bmatrix} -1 \\ 0 \end{bmatrix}
\end{aligned}
$$

Alternative Formulation - Alternate Approach Example

So

$$
\begin{aligned}
U &= \begin{bmatrix} 0 & -1 \\ 1 & 0 \end{bmatrix} \\
V &= \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \\
\Sigma &= \begin{bmatrix} 3 & 0 \\ 0 & \frac{1}{2} \end{bmatrix}
\end{aligned}
$$

Alternative Formulation - Alternate Approach Example

Check:

$$
\begin{aligned}
U\Sigma V^T &= \begin{bmatrix} 0 & -1 \\ 1 & 0 \end{bmatrix} \begin{bmatrix} 3 & 0 \\ 0 & \frac{1}{2} \end{bmatrix} \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \\
&= \begin{bmatrix} 0 & -\frac{1}{2} \\ 3 & 0 \end{bmatrix} \\
&= A
\end{aligned}
$$

Proof of Existence of SVD

- We claimed in Lecture 18 that every matrix $A \in \mathbb{R}^{m \times n}$ has a singular value decomposition.
- We will now prove this result.

Proof:

- Let A be an arbitrary $m \times n$ matrix.
- The proof is by induction on $n \geq 1$.
- Recall that the induced matrix norm is defined as

$$\|A\| := \max_{\|x\|=1} \|Ax\| .$$

- Let $\sigma_1 = \|A\|_2$.
- Let v_1 have $\|v_1\|_2 = 1$ and a direction such that $\|Av_1\|_2 = \|A\|_2 = \sigma_1$.
- Also, let $u_1 = \frac{Av_1}{\sigma_1}$, so that $Av_1 = \sigma_1 u_1$.
- Note, $\sigma_1 > 0$, so that there is no possibility of a "dvide by zero" problem when defining u_1 this way.

Proof of Existence of SVD

- Consider any extensions of vectors $u_1 \in \mathbb{R}^m$ and $v_1 \in \mathbb{R}^n$ to orthonormal bases U_1 and V_1:

$$\begin{aligned} U_1 &= \left[u_1 \middle| \cdots\right] \text{ is } m \times m, \\ V_1 &= \left[v_1 \middle| \cdots\right] \text{ is } n \times n. \end{aligned}$$

- Then we have

$$U_1^T A V_1 \underbrace{=}_{\text{definition}} S = \underbrace{\begin{bmatrix} \sigma_1 & w^T \\ 0 & B \end{bmatrix}}_{\text{See below for explanation of column 1}},$$

where 0 is the $m-1$ column vector, w^T is a $n-1$ row vector, and B has dimensions $(m-1) \times (n-1)$.

Proof of Existence of SVD

- Note, the top-left comes from

$$\begin{aligned} Av_1 &= \sigma_1 u_1 \\ u_1^T Av_1 &= \sigma_1 \underbrace{u_1^T u_1}_{=1} \\ &= \sigma_1. \end{aligned}$$

- The bottom-left is zero because

$$\begin{aligned} u_i^T Av_1 &= u_i^T(\sigma_1 u_1), \\ &= \sigma_1 u_i^T u_1, \\ &= 0, \ \forall i > 1. \end{aligned}$$

Proof of Existence of SVD

- Now, we can show $w = 0$ as follows:
- Consider

$$
\begin{aligned}
& \left\| \begin{bmatrix} \sigma_1 & w^T \\ 0 & B \end{bmatrix} \begin{bmatrix} \sigma_1 \\ w \end{bmatrix} \right\|_2 \\
= & \left\| \begin{bmatrix} \sigma_1^2 + w^T w \\ Bw \end{bmatrix} \right\|_2 \\
= & \sqrt{\left(\sigma_1^2 + w^T w\right)^2 + (Bw)^T Bw}, \\
= & \sqrt{\left(\sigma_1^2 + w^T w\right)^2 + w^T \underbrace{B^T B}_{\text{symm, + semi-def}} w}, \\
\geq & \sigma_1^2 + w^T w \\
= & \sqrt{\sigma_1^2 + w^T w} \underbrace{\left\| \begin{bmatrix} \sigma_1 \\ w \end{bmatrix} \right\|_2}_{= \sqrt{\sigma_1^2 + w^T w}}.
\end{aligned}
$$

Proof of Existence of SVD

- By the induced matrix norm definition, this implies $\|S\|_2 \geq \sqrt{\sigma_1^2 + w^T w}$.
- However, since U_1 and V_1 are orthogonal, therefore $\|S\|_2 = \|A\|_2 = \sigma_1$.
- Thus we must have $w = 0$, and so

$$U_1^T A V_1 = \begin{bmatrix} \sigma_1 & 0 \\ 0 & B \end{bmatrix}.$$

Proof of Existence of SVD

- For $n = 1$ (base case) this completes the proof.
- For $n > 1$ (induction case), by the inductive hypothesis, the SVD of B exists. Write $B = U_2 \Sigma_2 V_2^T$.
- Then if we let

$$A = \underbrace{U_1 \begin{bmatrix} 1 & 0 \\ 0 & U_2 \end{bmatrix}}_{U} \underbrace{\begin{bmatrix} \sigma_1 & 0 \\ 0 & \Sigma_2 \end{bmatrix}}_{\Sigma} \underbrace{\begin{bmatrix} 1 & 0 \\ 0 & V_2^T \end{bmatrix} V_1^T}_{V^T},$$

  it is easy to verify that this is an SVD of A. (Some explanation is given on the next slide.)
- Therefore, in either case, the SVD of A always exists.

□

Proof of Existence of SVD - Additional Explanation

Earlier, we had

$$U_1^T A V_1 = \begin{bmatrix} \sigma_1 & 0 \\ 0 & B \end{bmatrix}.$$

Recall that U_1 and V_1 are orthogonal. Therefore left multiplying by U_1 and right multiplying by V_1 yields

$$A = U_1 \begin{bmatrix} \sigma_1 & 0 \\ 0 & B \end{bmatrix} V_1^T.$$

Thus it suffices to prove that

$$\begin{bmatrix} 1 & 0 \\ 0 & U_2 \end{bmatrix} \begin{bmatrix} \sigma_1 & 0 \\ 0 & \Sigma_2 \end{bmatrix} \begin{bmatrix} 1 & 0 \\ 0 & V_2^T \end{bmatrix} = \begin{bmatrix} \sigma_1 & 0 \\ 0 & B \end{bmatrix},$$

where $B = U_2 \Sigma_2 V_2^T$.

Stability Comparison

- This section gives a more detailed comparison of the stability of the two approaches to compute the SVD.
- Assume a **stable** algorithm is used for finding eigenvalues (e.g., QR iteration) such that

$$|\tilde{\lambda}_k - \lambda_k| = O\left(\epsilon_{\text{machine}}\|A\|\right),$$

  where $\tilde{\lambda}_k$ denotes the numerical approximation.
- This satisfies $\tilde{\lambda}_k = \lambda_k(A + \delta A)$, with $\frac{\|\delta A\|}{\|A\|} = O\left(\epsilon_{\text{machine}}\right)$.
- That is, we compute the exact eigenvalues for a slightly **perturbed** matrix, $A + \delta A$.

Stability Comparison

Applying this to $H = \begin{bmatrix} 0 & A^T \\ A & 0 \end{bmatrix}$, we can get the singular values with

$$|\tilde{\sigma}_k - \sigma_k| = |\tilde{\lambda}_k - \lambda_k| = O\left(\epsilon_{\text{machine}} \|H\|\right) = O\left(\epsilon_{\text{machine}} \|A\|\right).$$

Explanation for Why $\|H\| = \|A\|$:

- See Lecture Notes.

Stability Comparison

- If we instead applied our eigenvalue routine to $A^T A$

$$|\tilde{\lambda}_k - \lambda_k| = O\left(\epsilon_{\text{machine}} \|A^T A\|\right) \approx O\left(\epsilon_{\text{machine}} \|A\|^2\right).$$

- Taking the square roots (i.e., divide by $\sqrt{\lambda_k}$) to get σ_k gives

$$|\tilde{\sigma}_k - \sigma_k| = O\left(\frac{|\tilde{\lambda}_k - \lambda_k|}{\sqrt{\lambda_k}}\right) = O\left(\frac{\epsilon_{\text{machine}} \|A\|^2}{\sigma_k}\right).$$

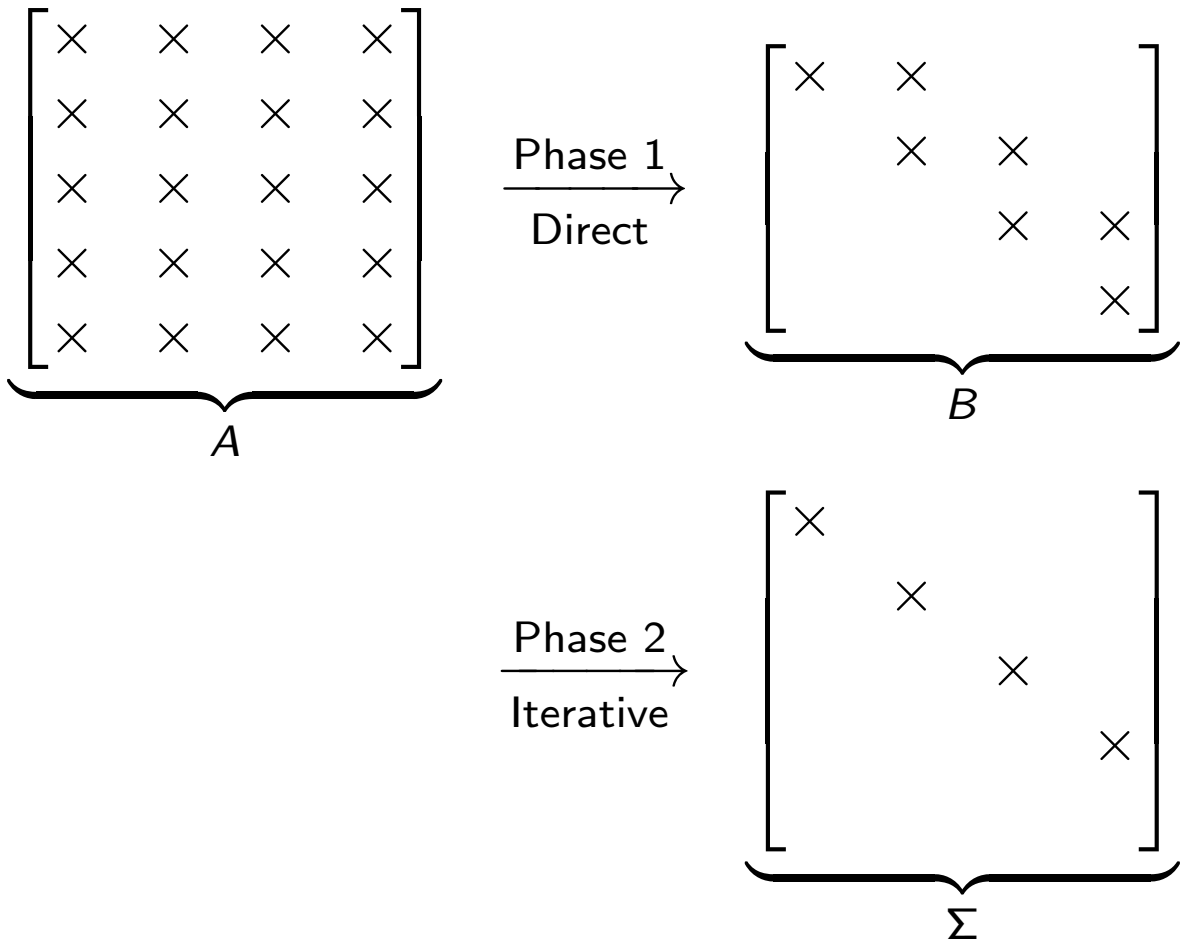
- This is quite inaccurate for $\sigma_k \ll \|A\|$.

Golub-Kahan Bidiagonalization

- This section describes a way of accelerating the SVD computation.
- We apply another two-phase process, as we did for QR iteration.
- We **pre-process** the matrix A to reduce the total cost of computing the SVD.
- The idea is to first convert to a **bidiagonal** matrix and then extract the SVD!

Golub-Kahan Bidiagonalization

This process is depicted below.



Golub-Kahan Bidiagonalization

- Why bidiagonal?
- We do not have to maintain a similarity transformation for SVD (unlike for the eigendecomposition).
- Therefore, we can apply **different** Householder reflectors on left and right to introduce zeros:

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

A $\qquad$ $U_1^T A$ $\qquad$ $U_1^T A V_1$

$$\begin{bmatrix} \times & \times & 0 & 0 \\ 0 & \times & \times & \times \\ 0 & 0 & \times & \times \\ 0 & 0 & \times & \times \\ 0 & 0 & \times & \times \end{bmatrix} \longrightarrow \begin{bmatrix} \times & \times & 0 & 0 \\ 0 & \times & \times & 0 \\ 0 & 0 & \times & \times \\ 0 & 0 & \times & \times \\ 0 & 0 & \times & \times \end{bmatrix} \longrightarrow \text{etc...}$$

$U_2^T U_1^T A V_1$ $\qquad$ $U_2^T U_1^T A V_1 V_2$

Golub-Kahan Bidiagonalization

- Bidiagonalization ultimately uses n reflectors on the left, $n-2$ on the right.
- Therefore, the cost of bidiagonalization is

$$\text{flops(bidagonalization)} \approx 2 \times \text{flops(QR)} \approx 4mn^2 - \frac{4}{3}m^3.$$

- For the case of $m \gg n$, there exist faster algorithms (see Trefethen & Bau, Lecture 31 if curious).

Golub-Kahan Bidiagonalization

For computing the SVD the cost is as follows.

- In practice, the cost of bidiagonalization phase $\approx O(mn^2)$ dominates over the eigendecomposition phase $\approx O(n^2)$.
- This cost is typically more expensive than other factorizations of square matrices we have seen previously:
  1. LU (Lecture 02): $\frac{2}{3}n^3 + O(n^2)$
  2. QR via Gram-Schmidt (Lecture 11): $2n^3 + O(n^2)$
  3. Eigendecomposition via Simultaneous Iteration (Lecture 16): Depends on setup and choice of tolerance
- However, the SVD computation is more numerically stable (i.e., preferable for ill-conditioned/rank-deficient matrices).
  - Here, **rank-deficient** simply means not of full rank.