Some NLA practice

Daniil Merkulov

Numerical Linear Algebra. Skoltech

Lectures 7-8 recap

Matrix decompositions and linear systems

In a least-squares, or linear regression, problem, we have measurements $X \in \mathbb{R}^{m \times n}$ and $y \in \mathbb{R}^m$ and seek a vector $\theta \in \mathbb{R}^n$ such that $X\theta$ is close to y. Closeness is defined as the sum of the squared differences:

$$\sum_{i=1}^{m} (x_i^\top \theta - y_i)^2 \qquad \|X\theta - y\|_2^2 \to \min_{\theta \in \mathbb{R}^n} \qquad X\theta^* = y$$

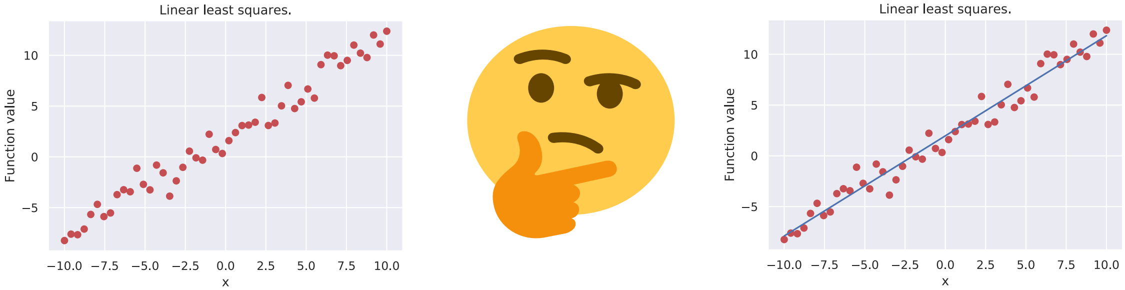


Figure 1: Illustration of linear system aka least squares

Matrix decompositions and linear systems. Approaches

Moore–Penrose inverse

If the matrix X is relatively small, we can write down and calculate exact solution:

$$\theta^* = (X^\top X)^{-1} X^\top y = X^\dagger y,$$

Matrix decompositions and linear systems. Approaches

Moore–Penrose inverse

If the matrix X is relatively small, we can write down and calculate exact solution:

$$\theta^* = (X^\top X)^{-1} X^\top y = X^\dagger y,$$

where $X^\dagger$ is called pseudo-inverse matrix. However, this approach squares the condition number of the problem, which could be an obstacle in case of ill-conditioned huge scale problem.

Matrix decompositions and linear systems. Approaches

Moore–Penrose inverse

If the matrix X is relatively small, we can write down and calculate exact solution:

$$\theta^* = (X^\top X)^{-1} X^\top y = X^\dagger y,$$

where $X^\dagger$ is called pseudo-inverse matrix. However, this approach squares the condition number of the problem, which could be an obstacle in case of ill-conditioned huge scale problem.

QR decomposition

For any matrix $X \in \mathbb{R}^{m \times n}$ there is exists QR decomposition:

$$X = Q \cdot R,$$

Matrix decompositions and linear systems. Approaches

Moore–Penrose inverse

If the matrix X is relatively small, we can write down and calculate exact solution:

$$\theta^* = (X^\top X)^{-1} X^\top y = X^\dagger y,$$

where $X^\dagger$ is called pseudo-inverse matrix. However, this approach squares the condition number of the problem, which could be an obstacle in case of ill-conditioned huge scale problem.

QR decomposition

For any matrix $X \in \mathbb{R}^{m \times n}$ there is exists QR decomposition:

$$X = Q \cdot R,$$

where Q is an orthogonal matrix (its columns are orthogonal unit vectors) meaning $Q^\top Q = QQ^\top = I$ and R is an upper triangular matrix. It is important to notice, that since $Q^{-1} = Q^\top$, we have:

$$QR\theta = y \quad \longrightarrow \quad R\theta = Q^\top y$$

Now, process of finding theta consists of two steps:

1. Find the QR decomposition of X.

Matrix decompositions and linear systems. Approaches

Moore–Penrose inverse

If the matrix X is relatively small, we can write down and calculate exact solution:

$$\theta^* = (X^\top X)^{-1} X^\top y = X^\dagger y,$$

where $X^\dagger$ is called pseudo-inverse matrix. However, this approach squares the condition number of the problem, which could be an obstacle in case of ill-conditioned huge scale problem.

QR decomposition

For any matrix $X \in \mathbb{R}^{m \times n}$ there is exists QR decomposition:

$$X = Q \cdot R,$$

where Q is an orthogonal matrix (its columns are orthogonal unit vectors) meaning $Q^\top Q = QQ^\top = I$ and R is an upper triangular matrix. It is important to notice, that since $Q^{-1} = Q^\top$, we have:

$$QR\theta = y \quad \longrightarrow \quad R\theta = Q^\top y$$

Now, process of finding theta consists of two steps:

1. Find the QR decomposition of X.
2. Solve triangular system $R\theta = Q^\top y$, which is triangular and, therefore, easy to solve.

Matrix decompositions and linear systems. Approaches

Cholesky decomposition

For any positive definite matrix $A \in \mathbb{R}^{n \times n}$ there is exists Cholesky decomposition:

$$X^\top X = A = L^\top \cdot L,$$

where L is an lower triangular matrix. We have:

$$L^\top L\theta = y \quad \longrightarrow \quad L^\top z_\theta = y$$

Now, process of finding theta consists of two steps:

1. Find the Cholesky decomposition of $X^\top X$.

Note, that in this case the error stil proportional to the squared condition number.

Matrix decompositions and linear systems. Approaches

Cholesky decomposition

For any positive definite matrix $A \in \mathbb{R}^{n \times n}$ there is exists Cholesky decomposition:

$$X^\top X = A = L^\top \cdot L,$$

where L is an lower triangular matrix. We have:

$$L^\top L\theta = y \quad \longrightarrow \quad L^\top z_\theta = y$$

Now, process of finding theta consists of two steps:

1. Find the Cholesky decomposition of $X^\top X$.
2. Find the $z_\theta = L\theta$ by solving triangular system $L^\top z_\theta = y$

Note, that in this case the error stil proportional to the squared condition number.

Matrix decompositions and linear systems. Approaches

Cholesky decomposition

For any positive definite matrix $A \in \mathbb{R}^{n \times n}$ there is exists Cholesky decomposition:

$$X^\top X = A = L^\top \cdot L,$$

where L is an lower triangular matrix. We have:

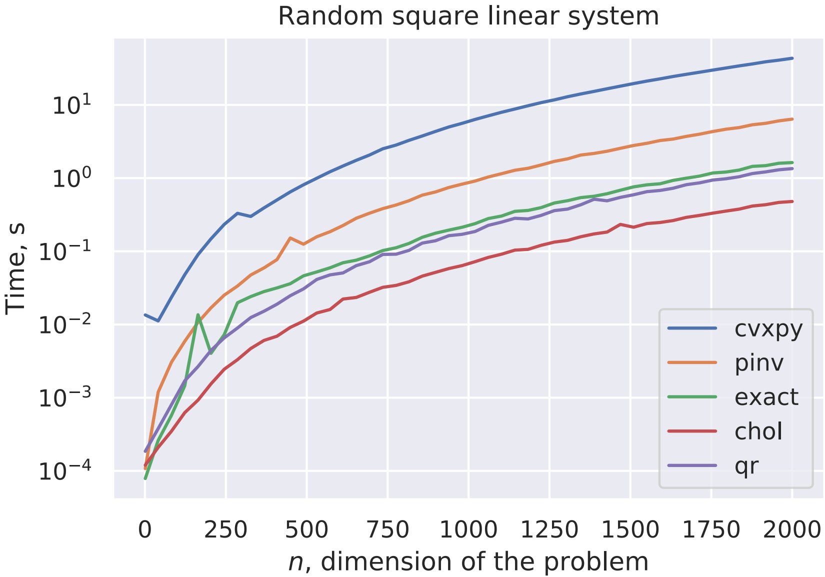
$$L^\top L\theta = y \quad \longrightarrow \quad L^\top z_\theta = y$$

Now, process of finding theta consists of two steps:

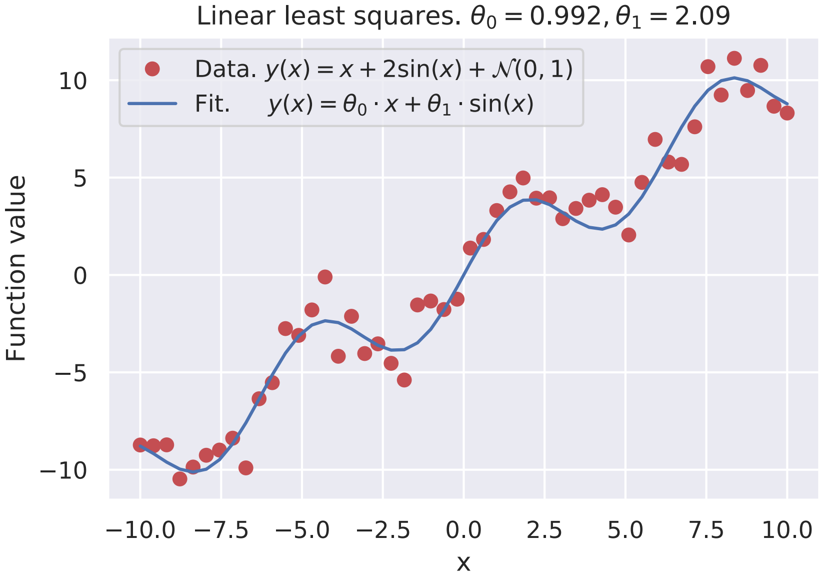
1. Find the Cholesky decomposition of $X^\top X$.
2. Find the $z_\theta = L\theta$ by solving triangular system $L^\top z_\theta = y$
3. Find the θ by solving triangular system $L\theta = z_\theta$

Note, that in this case the error stil proportional to the squared condition number.

Matrix decompositions and linear systems. Approaches



Matrix decompositions and linear systems. Non-linear data



Gram–Schmidt process

Input: n linearly independent vectors $u_0, \ldots, u_{n-1}$.

Output: n linearly independent vectors, which are pairwise orthogonal $d_0, \ldots, d_{n-1}$.

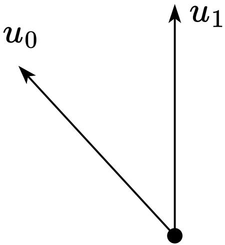


Figure 4: Illustration of Gram-Schmidt orthogonalization process

Gram–Schmidt process

Input: n linearly independent vectors $u_0, \ldots, u_{n-1}$.

Output: n linearly independent vectors, which are pairwise orthogonal $d_0, \ldots, d_{n-1}$.

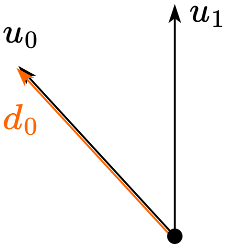


Figure 5: Illustration of Gram-Schmidt orthogonalization process

Gram–Schmidt process

Input: n linearly independent vectors $u_0, \ldots, u_{n-1}$.

Output: n linearly independent vectors, which are pairwise orthogonal $d_0, \ldots, d_{n-1}$.

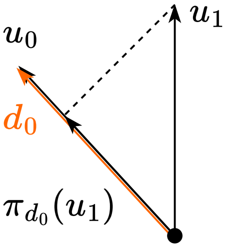


Figure 6: Illustration of Gram-Schmidt orthogonalization process

Gram–Schmidt process

Input: n linearly independent vectors $u_0, \ldots, u_{n-1}$.

Output: n linearly independent vectors, which are pairwise orthogonal $d_0, \ldots, d_{n-1}$.

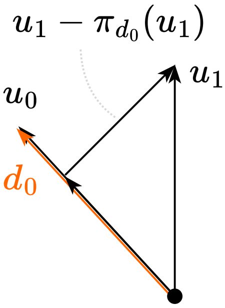


Figure 7: Illustration of Gram-Schmidt orthogonalization process

Gram–Schmidt process

Input: n linearly independent vectors $u_0, \ldots, u_{n-1}$.

Output: n linearly independent vectors, which are pairwise orthogonal $d_0, \ldots, d_{n-1}$.

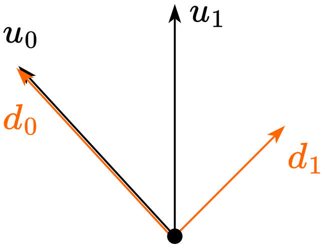
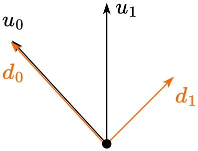
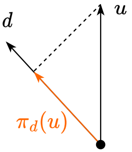


Figure 8: Illustration of Gram-Schmidt orthogonalization process

Gram–Schmidt process

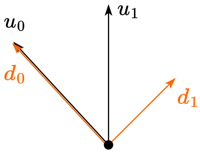
Input: n linearly independent vectors $u_0, \ldots, u_{n-1}$.

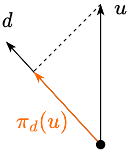




$$\pi_d(u) = \frac{\langle d, u \rangle}{\|d\|_2^2} d$$

Gram–Schmidt process



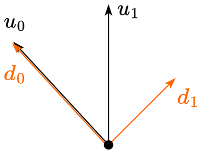


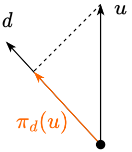
$$\pi_d(u) = \frac{\langle d, u \rangle}{\|d\|_2^2} d$$

Input: n linearly independent vectors $u_0, \ldots, u_{n-1}$.
Output: n linearly independent vectors, which are pairwise orthogonal $d_0, \ldots, d_{n-1}$.

$$d_0 = u_0$$

Gram–Schmidt process



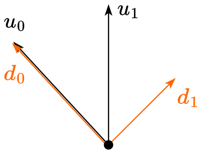


$$\pi_d(u) = \frac{\langle d, u \rangle}{\|d\|_2^2} d$$

Input: n linearly independent vectors $u_0, \ldots, u_{n-1}$.
Output: n linearly independent vectors, which are pairwise orthogonal $d_0, \ldots, d_{n-1}$.

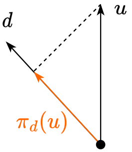
$$d_0 = u_0$$
$$d_1 = u_1 - \pi_{d_0}(u_1)$$

Gram–Schmidt process



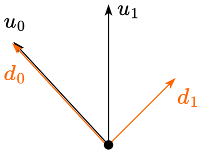
Input: n linearly independent vectors $u_0, \ldots, u_{n-1}$.
Output: n linearly independent vectors, which are pairwise orthogonal $d_0, \ldots, d_{n-1}$.

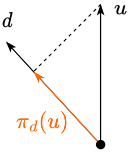
$$
\begin{aligned}
d_0 &= u_0 \\
d_1 &= u_1 - \pi_{d_0}(u_1) \\
d_2 &= u_2 - \pi_{d_0}(u_2) - \pi_{d_1}(u_2)
\end{aligned}
$$



$$\pi_d(u) = \frac{\langle d, u \rangle}{\|d\|_2^2} d$$

Gram–Schmidt process



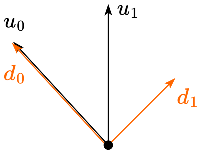


$$\pi_d(u) = \frac{\langle d, u\rangle}{\|d\|_2^2} d$$

Input: n linearly independent vectors $u_0, \ldots, u_{n-1}$.
Output: n linearly independent vectors, which are pairwise orthogonal $d_0, \ldots, d_{n-1}$.

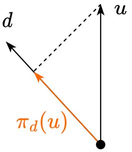
$$\begin{aligned}
d_0 &= u_0 \\
d_1 &= u_1 - \pi_{d_0}(u_1) \\
d_2 &= u_2 - \pi_{d_0}(u_2) - \pi_{d_1}(u_2) \\
&\vdots
\end{aligned}$$

Gram–Schmidt process



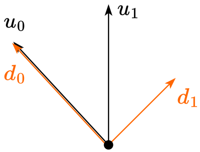
Input: n linearly independent vectors $u_0, \ldots, u_{n-1}$.
Output: n linearly independent vectors, which are pairwise orthogonal $d_0, \ldots, d_{n-1}$.

$$
\begin{aligned}
d_0 &= u_0 \\
d_1 &= u_1 - \pi_{d_0}(u_1) \\
d_2 &= u_2 - \pi_{d_0}(u_2) - \pi_{d_1}(u_2) \\
&\vdots \\
d_k &= u_k - \sum_{i=0}^{k-1} \pi_{d_i}(u_k)
\end{aligned}
$$



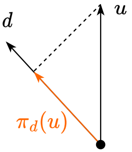
$$\pi_d(u) = \frac{\langle d, u \rangle}{\|d\|_2^2} d$$

Gram–Schmidt process



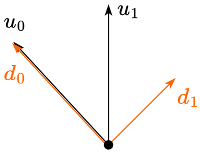
Input: n linearly independent vectors $u_0, \ldots, u_{n-1}$.
Output: n linearly independent vectors, which are pairwise orthogonal $d_0, \ldots, d_{n-1}$.

$$
\begin{aligned}
d_0 &= u_0 \\
d_1 &= u_1 - \pi_{d_0}(u_1) \\
d_2 &= u_2 - \pi_{d_0}(u_2) - \pi_{d_1}(u_2) \\
&\vdots \\
d_k &= u_k - \sum_{i=0}^{k-1} \pi_{d_i}(u_k)
\end{aligned}
$$



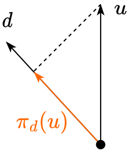
$$\pi_d(u) = \frac{\langle d, u \rangle}{\|d\|_2^2} d$$

Gram–Schmidt process



Input: n linearly independent vectors $u_0, \dots, u_{n-1}$.
Output: n linearly independent vectors, which are pairwise orthogonal $d_0, \dots, d_{n-1}$.

$$\begin{aligned}
d_0 &= u_0 \\
d_1 &= u_1 - \pi_{d_0}(u_1) \\
d_2 &= u_2 - \pi_{d_0}(u_2) - \pi_{d_1}(u_2) \\
&\vdots \\
d_k &= u_k - \sum_{i=0}^{k-1} \pi_{d_i}(u_k)
\end{aligned}$$



$$\pi_d(u) = \frac{\langle d, u \rangle}{\|d\|_2^2} d$$

$$d_k = u_k + \sum_{i=0}^{k-1} \beta_{ik} d_i \qquad \beta_{ik} = -\frac{\langle d_i, u_k \rangle}{\langle d_i, d_i \rangle} \tag{1}$$

Here's how you can structure the final slide to illustrate the **Gram-Schmidt process** in matrix form via QR decomposition:

Gram–Schmidt process in Matrix Form via QR Decomposition

Step-by-step process in matrix notation:

- Given a matrix A with columns $u_0, u_1, \dots, u_{n-1}$, the goal is to decompose A into:

$$A = QR$$

where:

Gram–Schmidt process in Matrix Form via QR Decomposition

Step-by-step process in matrix notation:

- Given a matrix A with columns $u_0, u_1, \ldots, u_{n-1}$, the goal is to decompose A into:

$$A = QR$$

  where:
  - Q: an orthogonal matrix whose columns are the orthonormal vectors $q_0, q_1, \ldots, q_{n-1}$.

Gram–Schmidt process in Matrix Form via QR Decomposition

Step-by-step process in matrix notation:

- Given a matrix A with columns $u_0, u_1, \ldots, u_{n-1}$, the goal is to decompose A into:

$$A = QR$$

where:
  - Q: an orthogonal matrix whose columns are the orthonormal vectors $q_0, q_1, \ldots, q_{n-1}$.
  - R: an upper triangular matrix.

Gram–Schmidt process in Matrix Form via QR Decomposition

Step-by-step process in matrix notation:

- Given a matrix A with columns $u_0, u_1, \ldots, u_{n-1}$, the goal is to decompose A into:

$$A = QR$$

  where:
  - Q: an orthogonal matrix whose columns are the orthonormal vectors $q_0, q_1, \ldots, q_{n-1}$.
  - R: an upper triangular matrix.

Gram–Schmidt process in Matrix Form via QR Decomposition

Step-by-step process in matrix notation:

- Given a matrix A with columns $u_0, u_1, \dots, u_{n-1}$, the goal is to decompose A into:

$$A = QR$$

  where:
  - Q: an orthogonal matrix whose columns are the orthonormal vectors $q_0, q_1, \dots, q_{n-1}$.
  - R: an upper triangular matrix.

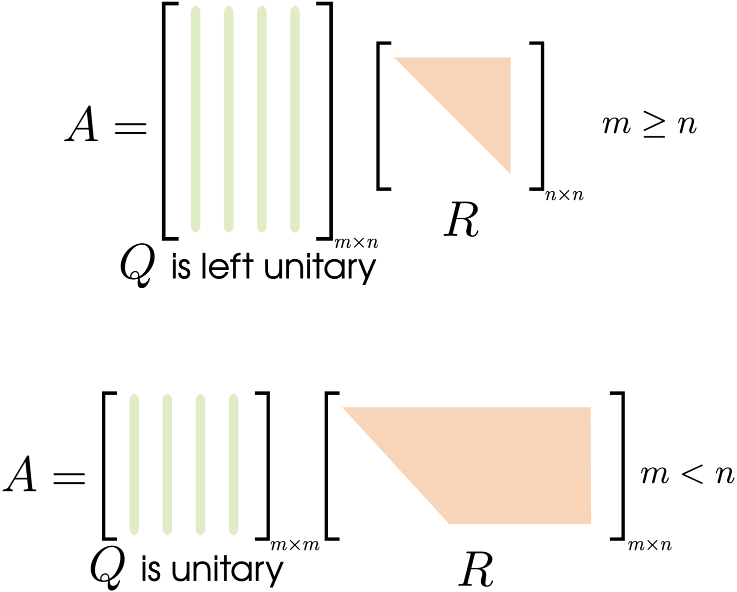
Illustration:

$$v_k = u_k - \sum_{i=0}^{k-1} \langle u_k, q_i \rangle q_i \qquad q_k = \frac{v_k}{\|v_k\|} \qquad R_{ij} = \langle u_j, q_i \rangle \qquad \text{for } i \leq j$$

$$\text{For } A = \begin{bmatrix} | & | & & | \\ u_0 & u_1 & \cdots & u_{n-1} \\ | & | & & | \end{bmatrix} \to Q = \begin{bmatrix} | & | & & | \\ q_0 & q_1 & \cdots & q_{n-1} \\ | & | & & | \end{bmatrix}, \quad R = \begin{bmatrix} r_{00} & r_{01} & \cdots & r_{0(n-1)} \\ 0 & r_{11} & \cdots & r_{1(n-1)} \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & r_{(n-1)(n-1)} \end{bmatrix}$$

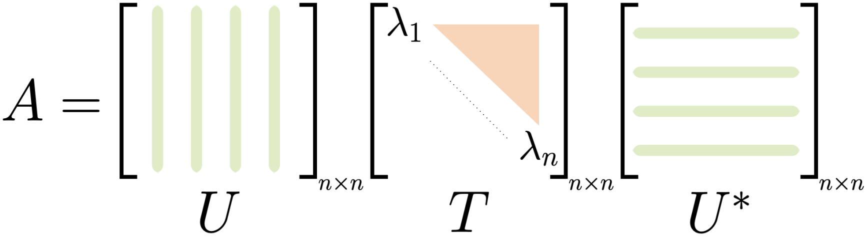
QR decomposition

QR



Schur form

Schur



- U is unitary
- $\lambda_1, \dots, \lambda_n$ are *eigenvalues*
- columns of U are *Schur vectors*

Figure 10: Decomposition

QR algorithm

- The QR algorithm was independently proposed in 1961 by Kublanovskaya and Francis.

QR algorithm

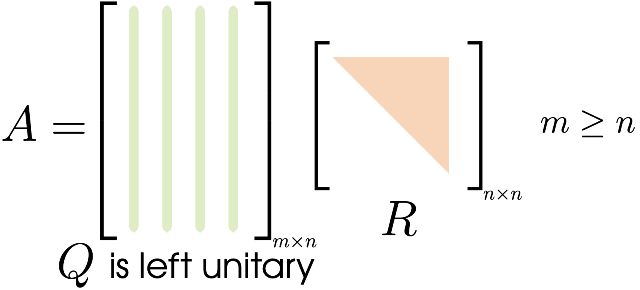
- The QR algorithm was independently proposed in 1961 by Kublanovskaya and Francis.
- Do not **mix** QR algorithm and QR decomposition!

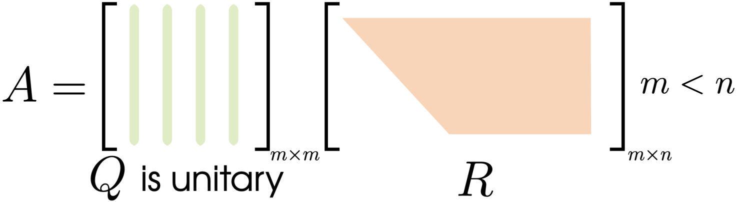
QR algorithm

- The QR algorithm was independently proposed in 1961 by Kublanovskaya and Francis.
- Do not **mix** QR algorithm and QR decomposition!
- QR decomposition is the representation of a matrix, whereas QR algorithm uses QR decomposition to compute the eigenvalues!

SVD

QR


$A =$
$m\times n$
Q is left unitary
R
$n\times n$
$m \geq n$


$A =$
$m\times m$
Q is unitary
R
$m\times n$
$m < n$

Singular value decomposition

Suppose $A \in \mathbb{R}^{m \times n}$ with rank $A = r$. Then A can be factored as

$$A = U\Sigma V^T$$

Singular value decomposition

Suppose $A \in \mathbb{R}^{m \times n}$ with rank $A = r$. Then A can be factored as

$$A = U \Sigma V^T$$

where $U \in \mathbb{R}^{m \times m}$ satisfies $U^T U = I$, $V \in \mathbb{R}^{n \times n}$ satisfies $V^T V = I$, and Σ is a matrix with non-zero elements on the main diagonal $\Sigma = \text{diag}(\sigma_1, ..., \sigma_r) \in \mathbb{R}^{m \times n}$, such that

Singular value decomposition

Suppose $A \in \mathbb{R}^{m \times n}$ with rank $A = r$. Then A can be factored as

$$A = U \Sigma V^T$$

where $U \in \mathbb{R}^{m \times m}$ satisfies $U^T U = I$, $V \in \mathbb{R}^{n \times n}$ satisfies $V^T V = I$, and Σ is a matrix with non-zero elements on the main diagonal $\Sigma = \text{diag}(\sigma_1, ..., \sigma_r) \in \mathbb{R}^{m \times n}$, such that

$$\sigma_1 \geq \sigma_2 \geq \ldots \geq \sigma_r > 0.$$

Singular value decomposition

Suppose $A \in \mathbb{R}^{m \times n}$ with rank $A = r$. Then A can be factored as

$$A = U \Sigma V^T$$

where $U \in \mathbb{R}^{m \times m}$ satisfies $U^T U = I$, $V \in \mathbb{R}^{n \times n}$ satisfies $V^T V = I$, and Σ is a matrix with non-zero elements on the main diagonal $\Sigma = \text{diag}(\sigma_1, ..., \sigma_r) \in \mathbb{R}^{m \times n}$, such that

$$\sigma_1 \geq \sigma_2 \geq \ldots \geq \sigma_r > 0.$$

This factorization is called the **singular value decomposition (SVD)** of A. The columns of U are called left singular vectors of A, the columns of V are right singular vectors, and the numbers σ_i are the singular values. The singular value decomposition can be written as

$$A = \sum_{i=1}^{r} \sigma_i u_i v_i^T,$$

where $u_i \in \mathbb{R}^m$ are the left singular vectors, and $v_i \in \mathbb{R}^n$ are the right singular vectors.

Singular value decomposition

Question

Suppose, matrix $A \in \mathbb{S}^n_{++}$. What can we say about the connection between its eigenvalues and singular values?

Singular value decomposition

> **Question**
>
> Suppose, matrix $A \in \mathbb{S}^n_{++}$. What can we say about the connection between its eigenvalues and singular values?

> **Question**
>
> How do the singular values of a matrix relate to its eigenvalues, especially for a symmetric matrix?

Skeleton decomposition

Simple, yet very interesting decomposition is Skeleton decomposition, which can be written in two forms:

$$A = UV^T \quad A = \hat{C}\hat{A}^{-1}\hat{R}$$

Skeleton decomposition

Simple, yet very interesting decomposition is Skeleton decomposition, which can be written in two forms:

$$A = UV^T \quad A = \hat{C}\hat{A}^{-1}\hat{R}$$

The latter expression refers to the fun fact: you can randomly choose r linearly independent columns of a matrix and any r linearly independent rows of a matrix and store only them with the ability to reconstruct the whole matrix exactly.

Skeleton decomposition

Simple, yet very interesting decomposition is Skeleton decomposition, which can be written in two forms:

$$A = UV^T \quad A = \hat{C}\hat{A}^{-1}\hat{R}$$

The latter expression refers to the fun fact: you can randomly choose r linearly independent columns of a matrix and any r linearly independent rows of a matrix and store only them with the ability to reconstruct the whole matrix exactly.
Use cases for Skeleton decomposition are:

- Model reduction, data compression, and speedup of computations in numerical analysis: given rank-r matrix with $r \ll n, m$ one needs to store $\mathcal{O}((n+m)r) \ll nm$ elements.

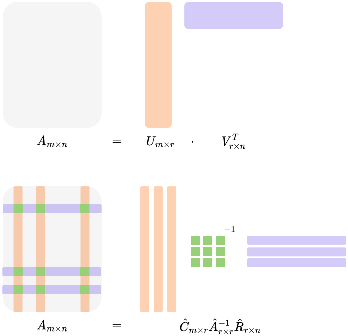


Figure 12: Illustration of Skeleton decomposition

Skeleton decomposition

Simple, yet very interesting decomposition is Skeleton decomposition, which can be written in two forms:

$$A = UV^T \quad A = \hat{C}\hat{A}^{-1}\hat{R}$$

The latter expression refers to the fun fact: you can randomly choose r linearly independent columns of a matrix and any r linearly independent rows of a matrix and store only them with the ability to reconstruct the whole matrix exactly.
Use cases for Skeleton decomposition are:

- Model reduction, data compression, and speedup of computations in numerical analysis: given rank-r matrix with $r \ll n, m$ one needs to store $\mathcal{O}((n+m)r) \ll nm$ elements.
- Feature extraction in machine learning, where it is also known as matrix factorization

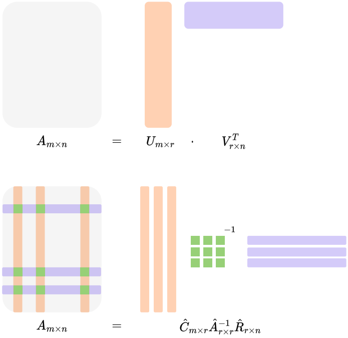


Figure 12: Illustration of Skeleton decomposition

Skeleton decomposition

Simple, yet very interesting decomposition is Skeleton decomposition, which can be written in two forms:

$$A = UV^T \quad A = \hat{C}\hat{A}^{-1}\hat{R}$$

The latter expression refers to the fun fact: you can randomly choose r linearly independent columns of a matrix and any r linearly independent rows of a matrix and store only them with the ability to reconstruct the whole matrix exactly.
Use cases for Skeleton decomposition are:

- Model reduction, data compression, and speedup of computations in numerical analysis: given rank-r matrix with $r \ll n, m$ one needs to store $\mathcal{O}((n+m)r) \ll nm$ elements.
- Feature extraction in machine learning, where it is also known as matrix factorization
- All applications where SVD applies, since Skeleton decomposition can be transformed into truncated SVD form.

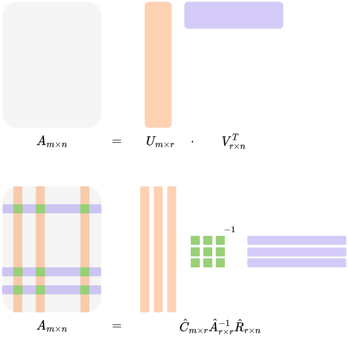


Figure 12: Illustration of Skeleton decomposition

Canonical tensor decomposition

One can consider the generalization of Skeleton decomposition to the higher order data structure, like tensors, which implies representing the tensor as a sum of r primitive tensors.

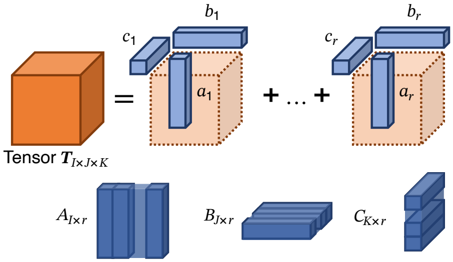


Figure 13: Illustration of Canonical Polyadic decomposition

> **Example**
>
> Note, that there are many tensor decompositions: Canonical, Tucker, Tensor Train (TT), Tensor Ring (TR), and others. In the tensor case, we do not have a straightforward definition of *rank* for all types of decompositions. For example, for TT decomposition rank is not a scalar, but a vector.

Problems

Example. Simple yet important idea on matrix computations.

Suppose, you have the following expression

$$
b = A_1 A_2 A_3 x,
$$

where the $A_1, A_2, A_3 \in \mathbb{R}^{3 \times 3}$ - random square dense matrices and $x \in \mathbb{R}^n$ - vector. You need to compute b.

Which one way is the best to do it?

1. $A_1 A_2 A_3 x$ (from left to right)

Check the simple code snippet after all.

Example. Simple yet important idea on matrix computations.

Suppose, you have the following expression

$$
b = A_1 A_2 A_3 x,
$$

where the $A_1, A_2, A_3 \in \mathbb{R}^{3 \times 3}$ - random square dense matrices and $x \in \mathbb{R}^n$ - vector. You need to compute b.

Which one way is the best to do it?

1. $A_1 A_2 A_3 x$ (from left to right)
2. $(A_1 (A_2 (A_3 x)))$ (from right to left)

Check the simple code snippet after all.

Example. Simple yet important idea on matrix computations.

Suppose, you have the following expression

$$
b = A_1 A_2 A_3 x,
$$

where the $A_1, A_2, A_3 \in \mathbb{R}^{3 \times 3}$ - random square dense matrices and $x \in \mathbb{R}^n$ - vector. You need to compute b.

Which one way is the best to do it?

1. $A_1 A_2 A_3 x$ (from left to right)
2. $\left(A_1 \left(A_2 \left(A_3 x\right)\right)\right)$ (from right to left)
3. It does not matter

Check the simple code snippet after all.

Example. Simple yet important idea on matrix computations.

Suppose, you have the following expression

$$
b = A_1 A_2 A_3 x,
$$

where the $A_1, A_2, A_3 \in \mathbb{R}^{3 \times 3}$ - random square dense matrices and $x \in \mathbb{R}^n$ - vector. You need to compute b.

Which one way is the best to do it?

1. $A_1 A_2 A_3 x$ (from left to right)
2. $\left(A_1 \left(A_2 \left(A_3 x\right)\right)\right)$ (from right to left)
3. It does not matter
4. The results of the first two options will not be the same.

Check the simple code snippet after all.

Problem 1

Find SVD of the following matrix:

$$A = \begin{bmatrix} 1 \\ 2 \\ 3 \end{bmatrix}$$

Problem 1

Find SVD of the following matrix:

$$A = \begin{bmatrix} 1 \\ 2 \\ 3 \end{bmatrix}$$

Solution

1. Compute A^TA:

$$A^TA = \begin{bmatrix} 1 & 2 & 3 \end{bmatrix} \begin{bmatrix} 1 \\ 2 \\ 3 \end{bmatrix} = 1^2 + 2^2 + 3^2 = 14.$$

   The singular values σ_i are the square roots of the eigenvalues of A^TA. Since A^TA is a 1×1 matrix with value 14, the singular value is $\sigma = \sqrt{14}$.

Problem 1

Find SVD of the following matrix:

$$A = \begin{bmatrix} 1 \\ 2 \\ 3 \end{bmatrix}$$

Solution

1. Compute $A^T A$:

$$A^T A = \begin{bmatrix} 1 & 2 & 3 \end{bmatrix} \begin{bmatrix} 1 \\ 2 \\ 3 \end{bmatrix} = 1^2 + 2^2 + 3^2 = 14.$$

   The singular values σ_i are the square roots of the eigenvalues of $A^T A$. Since $A^T A$ is a 1×1 matrix with value 14, the singular value is $\sigma = \sqrt{14}$.
2. Since V is an $n \times n$ orthogonal matrix (1×1 in this case), it can be $V = [1]$ (or $V = [-1]$). We choose $V = [1]$.

Problem 1

Find SVD of the following matrix:

$$A = \begin{bmatrix} 1 \\ 2 \\ 3 \end{bmatrix}$$

Solution

1. Compute $A^T A$:

$$A^T A = \begin{bmatrix} 1 & 2 & 3 \end{bmatrix} \begin{bmatrix} 1 \\ 2 \\ 3 \end{bmatrix} = 1^2 + 2^2 + 3^2 = 14.$$

   The singular values σ_i are the square roots of the eigenvalues of $A^T A$. Since $A^T A$ is a 1×1 matrix with value 14, the singular value is $\sigma = \sqrt{14}$.

2. Since V is an $n \times n$ orthogonal matrix (1×1 in this case), it can be $V = [1]$ (or $V = [-1]$). We choose $V = [1]$.

3. The simplest form of SVD allows us to write:

$$A = U \Sigma V^T = \begin{bmatrix} \frac{1}{\sqrt{14}} \\ \frac{2}{\sqrt{14}} \\ \frac{3}{\sqrt{14}} \end{bmatrix} \begin{bmatrix} \sqrt{14} \end{bmatrix} \begin{bmatrix} 1 \end{bmatrix}$$

Problem 1

Find SVD of the following matrix:

$$A = \begin{bmatrix} 1 \\ 2 \\ 3 \end{bmatrix}$$

Solution

1. Compute $A^T A$:

$$A^T A = \begin{bmatrix} 1 & 2 & 3 \end{bmatrix} \begin{bmatrix} 1 \\ 2 \\ 3 \end{bmatrix} = 1^2 + 2^2 + 3^2 = 14.$$

   The singular values σ_i are the square roots of the eigenvalues of $A^T A$. Since $A^T A$ is a 1×1 matrix with value 14, the singular value is $\sigma = \sqrt{14}$.

2. Since V is an $n \times n$ orthogonal matrix (1×1 in this case), it can be $V = [1]$ (or $V = [-1]$). We choose $V = [1]$.

3. The simplest form of SVD allows us to write:

$$A = U\Sigma V^T = \begin{bmatrix} \frac{1}{\sqrt{14}} \\ \frac{2}{\sqrt{14}} \\ \frac{3}{\sqrt{14}} \end{bmatrix} \begin{bmatrix} \sqrt{14} \end{bmatrix} \begin{bmatrix} 1 \end{bmatrix}$$

4. However, if you would like to use another form with square singular matrices:

$$A = U\Sigma V^T = \begin{bmatrix} \frac{1}{\sqrt{14}} & \frac{1}{\sqrt{3}} & \frac{-5}{\sqrt{42}} \\ \frac{2}{\sqrt{14}} & \frac{1}{\sqrt{3}} & \frac{4}{\sqrt{42}} \\ \frac{3}{\sqrt{14}} & \frac{-1}{\sqrt{3}} & \frac{-1}{\sqrt{42}} \end{bmatrix} \begin{bmatrix} \sqrt{14} \\ 0 \\ 0 \end{bmatrix} \begin{bmatrix} 1 \end{bmatrix}$$

Problem 2

Find SVD of the following matrix:

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

Problem 3

Find R matrix in QR decomposition for matrix $A = ab^T$, where $a = [1, 2, 1, 2, 1, 2, 1], b = [1, 2, 3, 4, 5, 6, 7, 8, 9]$

Solution