Math 6880/7875: Advanced Optimization
(Numerical) Linear Algebra

Akil Narayan\textsuperscript{1}

\textsuperscript{1}Department of Mathematics, and Scientific Computing and Imaging (SCI) Institute
University of Utah

January 20, 2022
Why linear algebra?

Linear algebraic operations are foundational tools for many optimization problems. Some optimization problems are also explicitly solvable using linear algebra.

We’ll focus on a subset of tasks in numerical linear algebra, revolving around the factorizations,

- **Singular value decomposition**: writing a matrix as a conic sum of rank-1 pairwise orthogonal matrices
- **$QR$ decomposition**: Orthogonalizing vectors via Gram-Schmidt-like approaches
- **$LU$ decomposition**: Gaussian elimination
Why linear algebra?

Linear algebraic operations are foundational tools for many optimization problems. Some optimization problems are also explicitly solvable using linear algebra.

We’ll focus on a subset of tasks in numerical linear algebra, revolving around the factorizations,

- **Singular value decomposition**: writing a matrix as a conic sum of rank-1 pairwise orthogonal matrices
- **QR decomposition**: Orthogonalizing vectors via Gram-Schmidt-like approaches
- **LU decomposition**: Gaussian elimination
Vector metrics

The “size” of a vector can be measured via a norm.

Several vectors norms are “common”:

- $\ell^p$ norms, $p \geq 1$: $\|v\|_p = \sum_{j=1}^{n} |v_j|^p$.
- $\|Ax\|_2$ is a norm for any invertible (hence, square) matrix $A$

Without context, typically $\| \cdot \|$ refers to the 2-norm $\| \cdot \|_2$.

Norms are convex functions....

"triangle inequality"
Matrix metrics

Let $A \in \mathbb{R}^{m \times n}$. Matrix norms are quite a bit more complicated.

Two norms that are perhaps the most common are the *induced* 2-norm,

$$\|A\|_2 = \sup_{x \neq 0} \frac{\|Ax\|_2}{\|x\|_2},$$

and the *Frobenius* norm,

$$\|A\|_F^2 = \sum_{i \in [m], j \in [n]} |A_{i,j}|^2$$

Without context, frequently $\| \cdot \|$ refers to the *spectral* or induced 2-norm $\| \cdot \|_2$. 
Norm equivalence

For finite-dimensional vectors and matrices, any two norms are equivalent.

I.e., if \( \| \cdot \|_a \) and \( \| \cdot \|_b \) are (any!) vectors norms on \( n \)-dimensional space, then \( \exists \) a constant \( C = C(n) \) such that,

\[
\| v \|_a \leq C(n) \| v \|_b, \quad \forall v \in \mathbb{R}^n
\]

The same is true for matrix norms, but \( C \) may depend on both \( m \) and \( n \).
Eigenvalues and eigenvectors

Let $A \in \mathbb{R}^{n \times n}$. An **eigenvalue** of $A$ is any complex number satisfying,

$$Av = \lambda v,$$

$$v \in \mathbb{C}^{n \setminus \{0\}},$$

and any (nonzero) vector $v$ in the equality above is an **eigenvector**.

All square matrices have exactly $n$ eigenvalues, $(\lambda_1, \ldots, \lambda_n)$, possibly repeated.

$$Av_1 = \lambda_1 v_1, \quad Av_2 = \lambda_2 v_2, \ldots \quad Av_n = \lambda_n v_n.$$  

**Non-defective** matrices have a full set of linearly independent eigenvectors:

$$\text{span}\{v_1, \ldots, v_n\} = \mathbb{C}^n.$$  

**Non-defective** matrices are, equivalently, **diagonalizable**, that is,

$$V^{-1}AV = \Lambda,$$

$$V = (v_1, \ldots, v_n), \quad \Lambda = (\lambda_1, \ldots, \lambda_n).$$
Eigenvalues and eigenvectors

Let $A \in \mathbb{R}^{n \times n}$. An **eigenvalue** of $A$ is any complex number satisfying,

$$Av = \lambda v, \quad v \in \mathbb{C}^n \setminus \{0\},$$

and any (nonzero) vector $v$ in the equality above is an **eigenvector**.

All square matrices have exactly $n$ eigenvalues, $(\lambda_1, \ldots, \lambda_n)$, possibly repeated.

$$Av_1 = \lambda_1 v_1, \quad Av_2 = \lambda_2 v_2, \ldots \quad Av_n = \lambda_n v_n.$$

**Non-defective** matrices have a full set of linearly independent eigenvectors:

$$\text{span}\{v_1, \ldots, v_n\} = \mathbb{C}^n.$$

Non-defective matrices are, equivalently, **diagonalizable**, that is,

$$V^{-1}AV = \Lambda, \quad V = (v_1, \ldots, v_n), \quad \Lambda = (\lambda_1, \ldots, \lambda_n).$$
Eigenvalues and eigenvectors

Let \( A \in \mathbb{R}^{n \times n} \). An **eigenvalue** of \( A \) is any complex number satisfying,

\[
Av = \lambda v, \quad v \in \mathbb{C}^n \setminus \{0\},
\]

and any (nonzero) vector \( v \) in the equality above is an **eigenvector**.

All square matrices have exactly \( n \) eigenvalues, \((\lambda_1, \ldots, \lambda_n)\), possibly repeated.

\[
Av_1 = \lambda_1 v_1, \quad Av_2 = \lambda_2 v_2, \ldots \quad Av_n = \lambda_n v_n.
\]

**Non-defective** matrices have a full set of linearly independent eigenvectors:

\[
\text{span}\{v_1, \ldots, v_n\} = \mathbb{C}^n.
\]

Non-defective matrices are, equivalently, **diagonalizable**, that is,

\[
V^{-1}AV = \Lambda, \quad V = (v_1, \ldots, v_n), \quad \Lambda = (\lambda_1, \ldots, \lambda_n).
\]

\[
AV = V \Lambda \quad \Rightarrow \quad \begin{pmatrix} Av_1 & \cdots & Av_n \end{pmatrix} = \begin{pmatrix} \lambda_1 v_1 & \cdots & \lambda_n v_n \end{pmatrix}
\]
Diagonalization

Diagonalizable matrices are, under an appropriate linear transformation, equal to a diagonal scaling operation.

“Most” matrices are diagonalizable, but many are not:

\[
A = \begin{pmatrix} 1 & 1 \\ 0 & 1 \end{pmatrix}.
\]

A matrix that is diagonalizable is “nice” in some limited sense, but there are “nicer” matrices.

The spectral radius of \( A \) is the maximum eigenvalue modulus:

\[
\rho(A) = \max_{j \in [n]} |\lambda_j|.
\]

Q: Eigenvalues seem to measure “size”. How does \( \rho(A) \) compare to, say, \( \|A\|_2 \)?
Diagonalization

Diagonalizable matrices are, under an appropriate linear transformation, equal to a diagonal scaling operation.

“Most” matrices are diagonalizable, but many are not:

\[
A = \begin{pmatrix} 1 & 1 \\ 0 & 1 \end{pmatrix}.
\]

A matrix that is diagonalizable is “nice” in some limited sense, but there are “nicer” matrices.

The spectral radius of \( A \) is the maximum eigenvalue modulus:

\[
\rho(A) = \max_{j \in [n]} |\lambda_j|.
\]

Q: Eigenvalues seem to measure “size”. How does \( \rho(A) \) compare to, say, \( \|A\|_2 \)?

True: \( \rho(A) \leq \|A\|_2 \) \( \Rightarrow \) \( \sup_{x \neq 0} \frac{\|A_k\|_2}{\|x\|_2} \geq \max_{i=1..n} \frac{\|A v_i\|_2}{\|v_i\|_2} = \rho(A) \)
But: \( A = \begin{pmatrix} 0 & R \\ 0 & 0 \end{pmatrix} \), \( R > 0 \)

\( \lambda(A) = \pm 1 \) \( \forall R \implies \rho(A) = 1 \)

But: \( \frac{\|A(0)\|_2}{\|0\|_2} = R \implies \|A\|_2 \geq R \)

But: Suppose \( A \) is diagonalizable, \( A = V \Lambda V^{-1} \), and that \( V \) is orthogonal (\( V^TV = I \)).

Then: \( V^{-1} = V^T \). 2-norm invariant under orthogonal \( k \)-forms

\( \| Ax \|_2 = \| V \Lambda V^{-1} x \|_2 = \| \Lambda V^{-1} x \|_2 \leq \rho(A) \| V^{-1} x \|_2 \leq \rho(A) \| V^T x \|_2 = \rho(A) \| x \|_2 \)

\( \implies \frac{\| Ax \|_2}{\| x \|_2} \leq \rho(A) \)

\( \implies \rho(A) = \| A \|_2 \)
Unitary diagonalization

A more well-behaved eigenvalue decomposition would be one where the eigenvalue matrix is unitary. (Recall $U \in \mathbb{R}^{n \times n}$ is orthogonal or unitary if $U^T U = I$, implying $U^T = U^{-1}$.)

I.e., a “nice” square matrix $A$ would be one satisfying,

$$A = V \Lambda V^{-1}, \quad V^T V = I.$$  

Such matrices are unitarily diagonalizable.

**Theorem**

A matrix $A$ is unitarily diagonalizable if and only if it is a normal matrix.

(A matrix $A$ is normal if $AA^T = A^T A$.)

Note that symmetric and skew-symmetric matrices are normal matrices.
Unitary diagonalization

A more well-behaved eigenvalue decomposition would be one where the eigenvalue matrix is unitary. (Recall $U \in \mathbb{R}^{n \times n}$ is orthogonal or unitary if $U^T U = I$, implying $U^T = U^{-1}$.)

I.e., a “nice” square matrix $A$ would be one satisfying,

$$A = V \Lambda V^{-1}, \quad V^T V = I.$$ 

Such matrices are **unitarily diagonalizable**.

**Theorem**

A matrix $A$ is unitarily diagonalizable if and only if it is a normal matrix.

(A matrix $A$ is normal if $AA^T = A^T A$.)

Note that symmetric and skew-symmetric matrices are normal matrices.
Unitary diagonalization

A more well-behaved eigenvalue decomposition would be one where the eigenvalue matrix is unitary. (Recall $U \in \mathbb{R}^{n \times n}$ is orthogonal or unitary if $U^T U = I$, implying $U^T = U^{-1}$.)

I.e., a “nice” square matrix $A$ would be one satisfying,

$$A = V \Lambda V^{-1}, \quad V^T V = I.$$

Such matrices are unitarily diagonalizable.

**Theorem**

A matrix $A$ is unitarily diagonalizable if and only if it is a normal matrix.

(A matrix $A$ is normal if $AA^T = A^T A$.)

Note that symmetric and skew-symmetric matrices are normal matrices.
The spectral theorem

The facts discussed above are typically summarized and extended through the **Spectral Theorem**.

**Theorem**

Assume \( A \in \mathbb{C}^{n \times n} \) is normal. Then \( A \) is unitarily diagonalizable. Furthermore:

- If \( A \) is Hermitian/symmetric, then all its eigenvalues are real-valued.
- If \( A \) is skew-Hermitian/skew-symmetric, then all its eigenvalues are purely imaginary.

Unfortunately, “most” matrices are not normal.

However a decomposition, similar to unitary diagonalization, exists for general, even rectangular, matrices.
The spectral theorem

The facts discussed above are typically summarized and extended through the \textbf{Spectral Theorem}.

\begin{itemize}
  \item Assume $A \in \mathbb{C}^{n \times n}$ is normal. Then $A$ is unitarily diagonalizable. \\
  \textbf{Furthermore:}
  \begin{itemize}
    \item If $A$ is Hermitian/symmetric, then all its eigenvalues are real-valued.
    \item If $A$ is skew-Hermitian/skew-symmetric, then all its eigenvalues are purely imaginary.
  \end{itemize}
\end{itemize}

Unfortunately, “most” matrices are not normal.

However a decomposition, similar to unitary diagonalization, exists for general, even rectangular, matrices.
The singular value decomposition

Let \( A \in \mathbb{R}^{m \times n} \). Then, the **singular value decomposition** (SVD) of \( A \) is,

\[
A = U \Sigma V^T,
\]

where

- \( U \in \mathbb{R}^{m \times m} \) is unitary. \( U = (u_1, \ldots, u_m) \).
- \( V \in \mathbb{R}^{n \times n} \) is unitary. \( V = (v_1, \ldots, v_n) \).
- \( \Sigma \in \mathbb{R}^{m \times n} \) is diagonal with non-negative entries on the diagonal. \( \Sigma = \text{diag}(\sigma_1, \ldots, \sigma_p) \), with \( p = \min\{m, n\} \).

By convention, the singular values are listed in decreasing order,

\[
\sigma_1 \geq \sigma_2 \geq \cdots \geq \sigma_p.
\]

\( \sigma_j \) : “singular values”

\( u_j, v_k \) : “singular vectors”
SVD properties

\[
A = U \Sigma V^T, \quad U = (u_1, \ldots, u_m), \quad V = (v_1, \ldots, v_n)
\]

- \( \|A\|_2 = \max_{j \in [p]} \sigma_j = \sigma_1. \)
- \( \|A\|_F^2 = \sum_{j \in [p]} \sigma_j^2 \)
- With \( r = \text{rank}(A) \), \( \sigma_j > 0 \) for \( 1 \leq j \leq r \) and \( \sigma_j = 0 \) for \( j > r \).
- \( \text{range}(A) = \text{span}\{u_1, \ldots, u_r\} \)
- \( \text{ker}(A) = \text{span}\{v_{r+1}, \ldots, v_n\} \)
- \( \{\sigma_1^2, \ldots, \sigma_r^2\} \subseteq \lambda(AA^T), \lambda(A^TA). \)
Rank-1 summations

\[
\begin{pmatrix}
  u_1 & \cdots & u_m
\end{pmatrix}
\begin{pmatrix}
  \sigma_1 & & \\
  & \ddots & \\
  & & \sigma_p
\end{pmatrix}
\begin{pmatrix}
  v_1^T \\
  \vdots \\
  v_p^T
\end{pmatrix}
\]

A direct algebraic computation with the SVD reveals,

\[
A = U\Sigma V^T = \sum_{j=1}^{p} \sigma_j (u_j v_j^T).
\]

\[
\langle u_j v_j^T, u_k v_k^T \rangle_F = \delta_{j,k}. \quad \text{Note: } u_j v_j^T \text{ has Frobenius norm/2-norm equal to 1 and }
\]

Thus, the SVD is a conic sum of unit-norm “orthogonal” matrices.

The SVD allows us to directly answer a particularly important optimization question:

\[
\arg \min_{B \in S} \|A - B\|_2 = ? \quad S = \left\{ C \in \mathbb{R}^{m \times n} \mid \text{rank}(C) \leq k \right\},
\]

where \(k\) is fixed and satisfies \(k \leq \text{rank}(A)\).
Rank-1 summations

A direct algebraic computation with the SVD reveals,

$$A = U \Sigma V^T = \sum_{j=1}^{p} \sigma_j (u_j v_j^T).$$

Note: $u_j v_j^T$ has Frobenius norm/2-norm equal to 1 and $(u_j v_j^T)^T (u_k v_k^T) = \delta_{j,k}$.

Thus, the SVD is a conic sum of unit-norm “orthogonal” matrices.

The SVD allows us to directly answer a particularly important optimization question:

$$\arg \min_{B \in S} \| A - B \|_2 = ?$$

$$S = \{ C \in \mathbb{R}^{m \times n} \mid \text{rank}(C) \leq k \},$$

where $k$ is fixed and satisfies $k \leq \text{rank}(A)$.

$$\begin{pmatrix} \ast & \ast \\ \ast & 0 \end{pmatrix} \quad \begin{pmatrix} 0 & \ast \\ 0 & \ast \end{pmatrix}$$
SVD solves some optimization problems

Direct manipulation of the SVD of a matrix solves certain optimization problems.

We will see this for:
- low-rank approximation
- Procrustes analysis
Optimal low-rank approximation

With the SVD decomposition,

\[ A = U \Sigma V^T = \sum_{j=1}^{p} \sigma_j (u_j v_j^T), \]

define \( A_k := \sum_{j=1}^{k} \sigma_j (u_j v_j^T) \) as a truncation of this sum.

**Theorem (Schmidt-Eckart-Young-Mirsky)**

\[ A_k = \arg \min_{\text{rank}(B) \leq k} \| A - B \|_*, \]

where \( \| \cdot \|_* \) is either the induced 2-norm, or the Frobenius norm. **Furthermore we have an accuracy certificate,**

\[ \min_{\text{rank}(B) \leq k} \| A - B \|_2 = \| A - A_k \|_2 = \sigma_{k+1}, \]
\[ \min_{\text{rank}(B) \leq k} \| A - B \|_F^2 = \| A - A_k \|_F^2 = \sum_{j=k+1}^{p} \sigma_j^2. \]

This is a result about low-rank matrix approximation.
Compression and dimension reduction

Optimal low-rank approximations are often used in compressing data representations.

Let $A \in \mathbb{R}^{M \times n}$ be given, with $M \gg 1$. SVD-based (optimal) compression of $A$ amounts to replacing $A$ with its rank-$k$ approximation,

$$A \approx A_k = \sum_{j=1}^{k} \sigma_j (u_j v_j^T)$$

Storage of $A \sim Mn$ numbers
Storage of $A_k \sim (M + n)k \ll Mn$ numbers
Procrustes analysis
Procrustes analysis

Procrustes analysis: “benignly” modify data set to match reference.

Image registration registration, shape analysis, uniformizing disparately scaled data
The orthogonal Procrustes problem

Reference data: collect landmark points as columns of a matrix $R$. 
$R \in \mathbb{R}^{m \times n}$: $n$ points in $m$-dimensional space.

Object data: $A \in \mathbb{R}^{m \times n}$ the corresponding landmarks on source object

The orthogonal Procrustes problem

\[ R \in \mathbb{R}^{m \times n}, \quad A \in \mathbb{R}^{m \times n}. \]

Goal: “align” \( A \) to best fit \( R \). Types of allowed alignments:

- translations
- rotations
- reflections

Written in math: find an orthogonal matrix \( Q \) over \( m \)-dimensional space so that \( QA \approx R \).

\[
\min_{Q \in \mathbb{R}^{m \times m}} \|QA - R\|_F^2 \quad \text{subject to} \quad Q^TQ = QQ^T = I_m
\]

Is this problem convex?
The orthogonal Procrustes problem

\[ R \in \mathbb{R}^{m \times n}, \quad A \in \mathbb{R}^{m \times n}. \]

Goal: “align” \( A \) to best fit \( R \). Types of allowed alignments:

- translations
- rotations
- reflections

Written in math: find an orthogonal matrix \( Q \) over \( m \)-dimensional space so that \( QA \approx R \).

\[
\min_{Q \in \mathbb{R}^{m \times m}} \| QA - R \|_F^2 \quad \text{subject to} \quad Q^T Q = Q Q^T = I_m
\]

Is this problem convex?

\( \text{No} \quad \text{\( \sqrt{\text{No}} \) } \)

\( m = 1 \): \( Q_1 = +1 \)

\( m = 2 \): \( Q_2 = -1 \)
\[ R = \begin{bmatrix} r_1 & \cdots & r_3 \end{bmatrix} \]
\[ A = \begin{bmatrix} a_1 & \cdots & a_3 \end{bmatrix} \]
\[ \text{target landmarks} \quad (R) \]
\[ \text{result of QA} \quad \text{v} \]
\[ \| QA - R \|_F^2 = \sum_{j=1}^{3} \| r_j - QA_j \|_2^2 \]
\[ \min_{Q} \| QA - R \|_F^2 \quad \text{s.t.} \quad QTQ = I = QQ^T \]

**Property**: Given \( C, D \in \mathbb{R}^{M \times N} \)
\[ \| C \|_F^2 = \text{Tr} (C^T C) \]

**Inner product**: \( \langle C, D \rangle_F = \text{Tr} (D^T C) \)

\[ \min_{Q} \| QA - R \|_F^2 = \min_{Q} \langle QA - R, QA - R \rangle_F \]
\[ = \min_{Q} \langle QA, QA \rangle_F + \langle R, R \rangle_F - 2 \langle QA, R \rangle_F \]
\[ \text{Tr}(ATQ^TQA) + \| R \|_F^2 - \text{Tr}(QAR) \]
\[
\text{Tr}(A^TA) = \begin{aligned}
\min_{Q} & \quad \|A^T - R\|^2_F + \|R\|^2_F - 2 \langle Q, RA^T \rangle_F \\
\max_{Q} & \quad 2 \langle Q, RA^T \rangle_F \\
& \quad \overset{\text{square SVD}}{=} R A^T = U \Sigma V^T \\
& \quad \overset{\text{m} \times \text{m matrices}}{\geq} \\
\max_{Q} & \quad 2 \langle Q, U \Sigma V^T \rangle_F \\
& \quad \overset{\text{m} \times \text{m}}{\geq} \text{unitary, } W \\
& \quad w = V \Sigma U^T \\
& \quad \max_{w} 2 \langle w, \Sigma \rangle_F = \max_{w} \sum_{j=1}^{m} \sigma_j w_{j,j} \\
& \quad \text{achieved by } w_{j,j} = 1 \text{ and } w_{j,k} = 0 \forall k \neq j \\
& \quad \Rightarrow w = I = V \Sigma U^T
\end{aligned}
\]
The Procrustes solution

\[ \min_{Q \in \mathbb{R}^{m \times m}} \|QA - R\|_F^2 \quad \text{subject to} \quad Q^TQ = QQ^T = I_m \]

Solution:

- Compute the SVD of \( RA^T = U\Sigma V^T \)
- Solution: \( Q = UV^T \).

A related problem: the “closest” unitary matrix to a given \( A \in \mathbb{R}^{m \times m} \),

\[ \min_{Q \in \mathbb{R}^{m \times m}} \|Q - A\|_F^2 \quad \text{subject to} \quad Q^TQ = QQ^T = I_m \]

Solution: \( Q = UV^T \), where \( A = U\Sigma V^T \) is the SVD of \( A \).
The Procrustes solution

\[
\min_{Q \in \mathbb{R}^{m \times m}} \|QA - R\|_F^2 \quad \text{subject to} \quad Q^T Q = QQ^T = I_m
\]

Solution:

- Compute the SVD of \( RA^T = U\Sigma V^T \)
- Solution: \( Q = UV^T \).

A related problem: the “closest” unitary matrix to a given \( A \in \mathbb{R}^{m \times m} \),

\[
\min_{Q \in \mathbb{R}^{m \times m}} \|Q - A\|_F^2 \quad \text{subject to} \quad Q^T Q = QQ^T = I_m
\]

Solution: \( Q = UV^T \), where \( A = U\Sigma V^T \) is the SVD of \( A \).

Caveat: “generalized” Procrustes problems typically don’t have such nice solutions.
Orthogonalization

Our second factorization: $QR$

Idea: Given vectors $a_1, \ldots, a_n \in \mathbb{R}^m$, orthogonalize them:

\[
\{a_1, \ldots, a_n\} \longrightarrow \{q_1, \ldots, q_n\} \subset \mathbb{R}^m
\]

Such that $\langle q_k, q_j \rangle = q_j^T q_k = \delta_{k,j}$.

The conceptually simple strategy to accomplish this: Gram-Schmidt orthogonalization:

\[
\begin{align*}
    r_{1,1} &= \|u_1\|_2 \\
    r_{1,2} &= \langle a_2, q_1 \rangle \\
    r_{2,2} &= \|u_2\|_2, \\
    q_1 &= \frac{a_1}{r_{1,1}} \\
    q_2 &= \frac{u_2}{r_{2,2}} \\
    u_1 &= a_1 \\
    u_2 &= a_2 - r_{1,2} q_1, \\
    u_j &= a_j - \sum_{k<j} r_{k,j} q_k, \\
    u_j &= a_j - \sum_{k<j} r_{k,j} q_k,
\end{align*}
\]

\[
\begin{align*}
    r_{j,j} &= \|u_j\|_2 \\
    q_j &= \frac{u_j}{r_{j,j}}
\end{align*}
\]
Orthogonalization

Our second factorization: $QR$

Idea: Given vectors $a_1, \ldots, a_n \in \mathbb{R}^m$, orthogonalize them:

$$\{a_1, \ldots, a_n\} \longrightarrow \{q_1, \ldots, q_n\} \subset \mathbb{R}^m$$

such that $\langle q_k, q_j \rangle = q_j^T q_k = \delta_{k,j}$.

The conceptually simple strategy to accomplish this: Gram-Schmidt orthogonalization:

$$\begin{align*}
    r_{1,2} &= \langle a_2, q_1 \rangle \\
    u_1 &= a_1 \\
    u_2 &= a_2 - r_{1,2} q_1, \\
    r_{1,1} &= \|u_1\|_2 \\
    q_1 &= \frac{a_1}{r_{1,1}} \\
    r_{2,2} &= \|u_2\|_2, \\
    q_2 &= \frac{u_2}{r_{2,2}} \\
    \cdots \\
    r_{k,j} &= \langle a_j, q_k \rangle, \ (k < j) \\
    u_j &= a_j - \sum_{k<j} r_{k,j} q_k \\
    r_{j,j} &= \|u_j\|_2 \\
    q_j &= \frac{u_j}{r_{j,j}}
\end{align*}$$
The QR decomposition

Collect all these vectors into matrices:

\[
A = \begin{pmatrix}
a_1 & a_2 & \cdots & a_n \\
\end{pmatrix} \quad Q = \begin{pmatrix}
a_1 & a_2 & \cdots & a_n \\
\end{pmatrix}
\]

If one maintains a diary of orthogonalization operations, this is the QR decomposition:

\[
A = QR \approx \begin{pmatrix}
Q
\end{pmatrix} \begin{pmatrix}
\end{pmatrix}
\]

- \( Q \) is an orthogonal matrix: \( Q^T Q = I \).
- \( R \) is an upper triangular matrix.
Pivoting

A more powerful version of this algorithm is a pivoted one:

At step \( j \), the standard factorization computes:

\[
\begin{align*}
    r_{j,j} &= \left\| a_j - \sum_{k<j} \langle a_j, q_k \rangle q_k \right\|_2 \\
    &= \left\| a_j - PQ_{j-1} a_j \right\|_2, \quad Q_{j-1} = \text{span}\{q_1, \ldots, q_{k-1}\}
\end{align*}
\]

The pivoted QR decomposition first performs the permutation:

\[
\begin{align*}
    a_j, a_{j+1}, \ldots, a_{s-1}, a_s, a_{s+1}, \ldots, a_{n-1}, a_n \\
    \downarrow \quad a_s, a_{j+1}, \ldots, a_{s-1}, a_j, a_{s+1}, \ldots, a_{n-1}, a_n,
\end{align*}
\]

where \( s \) is chosen according to the rule,

\[
s = \arg \max_{k=j, \ldots, n} \left\| a_k - PQ_{j-1} a_k \right\|_2.
\]
Pivoting

A more powerful version of this algorithm is a \textit{pivoted} one:

At step $j$, the standard factorization computes:

\[
\begin{align*}
    r_{j,j} &= \left\| a_j - \sum_{k<j} \langle a_j, q_k \rangle q_k \right\|_2 \\
    &= \left\| a_j - P Q_{j-1} a_j \right\|_2, \quad Q_{j-1} = \text{span}\{q_1, \ldots, q_{k-1}\}
\end{align*}
\]

The \textit{pivoted} $QR$ decomposition first performs the permutation:

\[
\begin{align*}
    a_j, a_{j+1}, \ldots, a_{s-1}, a_s, a_{s+1}, \ldots, a_{n-1}, a_n \\
    \quad a_s, a_{j+1}, \ldots, a_{s-1}, a_j, a_{s+1}, \ldots, a_{n-1}, a_n
\end{align*}
\]

where $s$ is chosen according to the rule,

\[
s = \arg\max_{k=j,\ldots,n} \left\| a_k - P Q_{j-1} a_k \right\|_2.
\]
The *pivoted* $QR$ decomposition

I.e., this corresponds to a permutation of the column indices $\{1, \ldots, n\}$.

Then there is a permutation matrix\(^1\) $P \in \mathbb{R}^{n \times n}$, such that

$$AP = QR,$$

\(^1\) A permutation matrix $P$ has the form $P = [e_{\pi(1)}, \ldots, e_{\pi(n)}]$ for some permutation map $\pi$ of $[n]$. 
Combinatorial optimization

Many optimization problems take the form,

$$\max_{p_1, \ldots, p_N \in \Omega} f_N(p_1, \ldots, p_N),$$

where $f_N$ is an objective function of $N$ arguments, with $\Omega$ a feasible set of options. (I.e., an optimization problem with $N$ choices.)

- $f_N$ is the traveling salesman problem path length, with $N$ stops.
- The knapsack problem: identify $N$ items, where each has specifics weights and payoffs
- The assignment problem: Divide $N$ agents among many tasks so that the task payoff is maximized while minimizing the agent cost

These problems are typically hard: require global optimize over $N$ objects simultaneously
Combinatorial optimization

Many optimization problems take the form,

$$\max_{p_1, \ldots, p_N \in \Omega} f_N(p_1, \ldots, p_N),$$

where $f_N$ is an objective function of $N$ arguments, with $\Omega$ a feasible set of options. (I.e., an optimization problem with $N$ choices.)

- $f_N$ is the traveling salesman problem path length, with $N$ stops.
- The knapsack problem: identify $N$ items, where each has specifics weights and payoffs
- The assignment problem: Divide $N$ agents among many tasks so that the task payoff is maximized while minimizing the agent cost

These problems are typically hard: require global optimize over $N$ objects simultaneously
Greedy algorithms

One strategy to \textit{approximately} solve combinatorial optimization problems: \textit{Greedy} methods.

\[
\max_{p_1, \ldots, p_N \in \Omega} f_N(p_1, \ldots, p_N),
\]

In our language, a greedy algorithm to approximate the solution above is:

- Choose \( p_1 = \arg \max_{p \in S} f_1(p) \)
- For \( j = 2, \ldots, N \): choose \( p_j = \arg \max_{p \in S} f_j(p_1, \ldots, p_{j-1}, p) \)

Greedy algorithms (almost always) do not result in optimal solutions. But frequently they are \textit{close} to optimal.
Greedy algorithms

One strategy to *approximately* solve combinatorial optimization problems: *Greedy* methods.

\[
\max_{p_1, \ldots, p_N \in \Omega} f_N(p_1, \ldots, p_N),
\]

In our language, a greedy algorithm to approximate the solution above is:
- Choose \( p_1 = \arg \max_{p \in S} f_1(p) \)
- For \( j = 2, \ldots, N \): choose \( p_j = \arg \max_{p \in S} f_j(p_1, \ldots, p_{j-1}, p) \)

Greedy algorithms (almost always) do not result in optimal solutions.

But frequently they are close to optimal.
Pivoting and greedy algorithms

Consider the following (combinatorial) optimization problems:

\[ S = \arg \max_{S \subseteq [n], \ |S| = k} \max_{j \in [n]} \| a_j - P_{A_S} a_j \|_2, \]

\[ S = \arg \max_{S \subseteq [n], \ |S| = k} | \det A^T_S A_S | \]

Above, \( A_S \) is the submatrix of \( A \) formed by a subset of column indices \( S \). \( P_{A_S} \) is the orthogonal projection operator, projecting general vectors onto \( \text{range}(A_S) \).

1. Problem 1: Compute the subset of columns of \( A \) that minimizes the projection error of projecting each column of \( A \) onto the subspace spanned by the column subset.

2. Problem 2: Choose a column subset \( S \) that maximizes the determinant of the Gram matrix of \( A_S \).
Pivoting and greedy algorithms

Consider the following (combinatorial) optimization problems:

\[
S = \arg \max_{S \subseteq [n], |S| = k} \max_{j \in [n]} \|a_j - P_{A_S} a_j\|_2,
\]

\[
S = \arg \max_{S \subseteq [n], |S| = k} |\det A_S^T A_S|
\]

Above, \(A_S\) is the submatrix of \(A\) formed by a subset of column indices \(S\). \(P_{A_S}\) is the orthogonal projection operator, projecting general vectors onto \(\text{range}(A_S)\).

1. Problem 1: Compute the subset of columns of \(A\) that minimizes the projection error of projecting each column of \(A\) onto the subspace spanned by the column subset.

2. Problem 2: Choose a column subset \(S\) that maximizes the determinant of the Gram matrix of \(A_S\).
Problem 1: Minimizing residuals

\[ S = \text{arg max}_{S \subseteq [n], |S| = k} \max_{j \in [n]} \|a_j - P_{AS}a_j\|_2, \]

The pivoted QR decomposition gives an approximate (but easily computable!) solution,

\[ AP = QR \]

Choosing \( S \) as the first \( k \) columns chosen by the permutation matrix \( P \) is equivalent to the following greedy procedure:

\[ s_j = \text{arg max}_{s \in [n]} \max_{j \in [n]} \|a_j - P_{AS_{j-1}}a_j\|_2, \quad S_k = \{s_1, \ldots, s_k\}. \]

This kind of problem appears exactly in

- “Structured” data reduction: approximation of large data sets by a small number of exemplars (data coresets, matrix skeletonization)
- Scientific model reduction: columns of \( A \) are PDE solutions
Problem 1: Minimizing residuals

\[ S = \arg \max_{S \subset [n]|S|=k} \max_{j \in [n]} \| a_j - P_{A_S} a_j \|_2, \]

The pivoted \( QR \) decomposition gives an approximate (but easily computable!) solution,

\[ AP = QR \]

Choosing \( S \) as the first \( k \) columns chosen by the permutation matrix \( P \) is equivalent to the following greedy procedure:

\[ s_j = \arg \max_{s \in [n]} \max_{j \in [n]} \| a_j - P_{A_{S_{j-1}}} a_j \|_2, \quad S_k = \{s_1, \ldots, s_k\}. \]

This kind of problem appears exactly in

- “Structured” data reduction: approximation of large data sets by a small number of exemplars (data coresets, matrix skeletonization)
- Scientific model reduction: columns of \( A \) are PDE solutions
Problem 2: Determinant maximization

\[ S = \arg \max_{S \subset [n] | S| = k} \det A_S^T A_S \]

The pivoted QR decomposition gives an approximate (but easily computable!) solution,

\[ AP = QR \]

Choosing \( S \) as the first \( k \) columns chosen by the permutation matrix \( P \) is equivalent to the following greedy procedure:

\[ s_j = \arg \max_{s \in [n]} \max_{j \in [n]} \left| \det A_{S_{j-1}^*}^T A_{S_{j-1}^*}^* \right| \quad S_k = \{s_1, \ldots, s_k \}, \quad S_k^* = S_k \cup \{s\}. \]

This kind of problem appears exactly in

- Optimal experimental design: A \( D \)-optimal design of experiments maximizes the determinant of the Fisher Information Matrix.
- Function approximation: Point configurations maximizing a determinant are Fekete points, and are excellent sites for collecting data.
Problem 2: Determinant maximization

\[ S = \arg \max_{S \subset [n], |S| = k} |\det A_S^T A_S| \]

The pivoted \textit{QR} decomposition gives an approximate (but easily computable!) solution,

\[ AP = QR \]

Choosing \( S \) as the first \( k \) columns chosen by the permutation matrix \( P \) is equivalent to the following greedy procedure:

\[ s_j = \arg \max_{s \in [n]} \max_{j \in [n]} |\det A_{S_{j-1}^*}^T A_{S_{j-1}^*}| \quad S_k = \{s_1, \ldots, s_k\}, \quad S_k^* = S_k \cup \{s\}. \]

This kind of problem appears exactly in

- Optimal experimental design: A \( D \)-optimal design of experiments maximizes the determinant of the Fisher Information Matrix.

- Function approximation: Point configurations maximizing a determinant are \textit{Fekete points}, and are excellent sites for collecting data.
References 1


