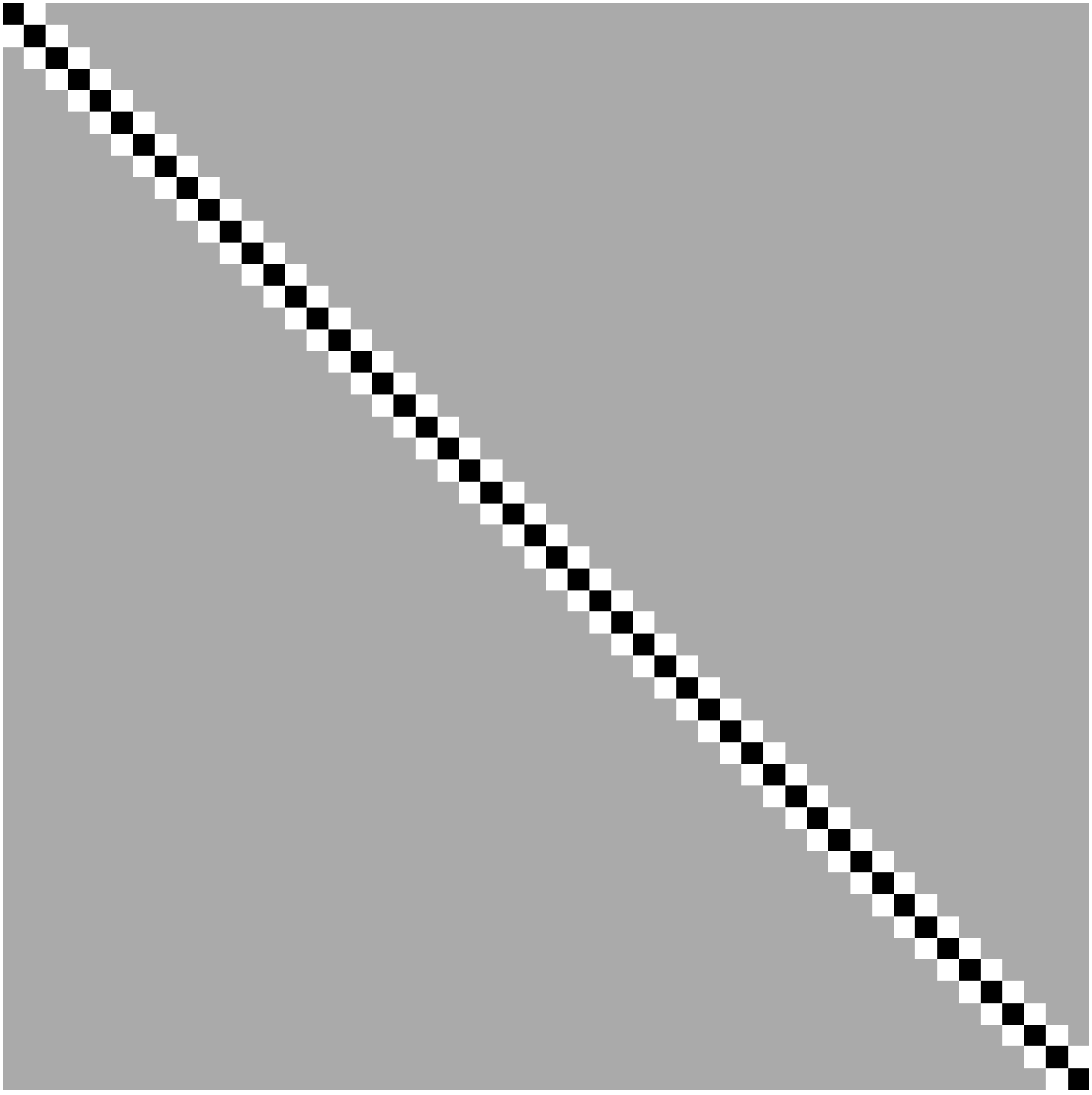
MATH 307

Applied Linear Algebra



January 6, 2021 v0.1

Contents

Preface

Learning Goals

- Summarize properties and constructions of matrix decompositions LU, QR and SVD
- Perform matrix computations using mathematical software Python, SciPy and Jupyter
- Compute solutions of large systems of linear equations using matrix decompositions
- Compute least squares approximations of large linear systems using matrix decompositions
- Compute eigenvalues of large matrices using iterative methods
- Analyze digital signals using the discrete Fourier transform
- Create and analyze mathematical models of real-world phenomenon

Prerequisites

We assume the reader has completed an introductory undergraduate course in linear algebra:

- linear systems of equations, row operations, elementary matrices and Gaussian elimination
- linear independence, span, subspaces, dimension and rank
- linear transformations, null space, row space and and column space of a matrix
- matrix multiplication, inverses and determinants
- eigenvalues and eigenvectors, characteristic polynomial and diagonalization
- dot product, length, orthogonality, orthogonal projection and Gram–Schmidt orthogonalization

See MATH 221, MATH 223 and MATH 152 at UBC Math.

Under Construction

This is a work in progress and the following sections are still under construction:

- Computed Tomography
- Eigenvalue Problems: Exercises on pseudoinverse, SVD expansion, image deblurring, computed tomography
- Convolution Theorem and Filtering

Chapter 1

Linear Systems of Equations

1.1 Review: Linear Systems

Big Idea. Solve a linear system of equations $A\boldsymbol{x} = \boldsymbol{b}$ by reducing the augmented matrix $[\,A\ \boldsymbol{b}\,]$ to row-echelon form via Gaussian elimination.

Definition. A **linear system of equations** [KN, p.1] is a collection of equations of the form

$$\begin{array}{ccccccccc} a_{1,1}x_1 & + & a_{1,2}x_2 & + & \cdots & + & a_{1,n}x_n & = & b_1 \\ a_{2,1}x_1 & + & a_{2,2}x_2 & + & \cdots & + & a_{2,n}x_n & = & b_2 \\ & & & & & & & \vdots & \\ a_{m,1}x_1 & + & a_{m,2}x_2 & + & \cdots & + & a_{m,n}x_n & = & b_m \end{array}$$

where the coefficients $a_{i,j}$, b_i are known constants. In matrix notation, we have $A\boldsymbol{x} = \boldsymbol{b}$ where

$$A = \begin{bmatrix} a_{1,1} & a_{1,2} & \cdots & a_{1,n} \\ a_{2,1} & a_{2,2} & \cdots & a_{2,n} \\ \vdots & \vdots & \ddots & \vdots \\ a_{m,1} & a_{m,2} & \cdots & a_{m,n} \end{bmatrix} \qquad \boldsymbol{x} = \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{bmatrix} \qquad \boldsymbol{b} = \begin{bmatrix} b_1 \\ b_2 \\ \vdots \\ b_m \end{bmatrix}$$

Definition. The following are called **elementary row operations** [KN, p.5]:

1. Interchange two rows.
2. Multiply one row by a nonzero number.
3. Add a multiple of one row to a different row.

Definition. A matrix is in **row-echelon form** [KN, p.10] if:

1. All zero rows are at the bottom.
2. The first nonzero entry in a row (from the left) is 1.
3. The first nonzero entry in a row (from the left) is located to the right of the first nonzero entry in every row above.

For example:

$$\begin{bmatrix} 1 & * & * \\ 0 & 1 & * \\ 0 & 0 & 1 \end{bmatrix} \quad \begin{bmatrix} 1 & * & * & * & * \\ 0 & 1 & * & * & * \\ 0 & 0 & 1 & * & * \\ 0 & 0 & 0 & 0 & 0 \end{bmatrix} \quad \begin{bmatrix} 1 & * & * & * & * \\ 0 & 0 & 1 & * & * \\ 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 \end{bmatrix} \quad \begin{bmatrix} 1 & * \\ 0 & 0 \end{bmatrix}$$

Theorem. Every matrix can be transformed to row-echelon form by a sequence of elementary row operations via the Gaussian elimination algorithm [KN, p.11].

Example. Find all solutions of $A\boldsymbol{x} = \boldsymbol{b}$ for:

1. $A = \begin{bmatrix} 1 & 1 & 1 \\ 1 & 0 & 1 \\ 2 & 5 & 2 \end{bmatrix} \quad \boldsymbol{b} = \begin{bmatrix} 2 \\ 1 \\ 7 \end{bmatrix}$

2. $A = \begin{bmatrix} 1 & -1 & 1 & -2 \\ -1 & 1 & 1 & 1 \\ -1 & 2 & 3 & 1 \\ 1 & -1 & 2 & 1 \end{bmatrix} \quad \boldsymbol{b} = \begin{bmatrix} 1 \\ -1 \\ 2 \\ 1 \end{bmatrix}$

3. $A = \begin{bmatrix} 1 & 1 & 2 & 1 \\ 1 & 0 & 1 & 1 \\ 0 & 1 & 1 & 0 \end{bmatrix} \quad \boldsymbol{b} = \begin{bmatrix} 1 \\ 1 \\ 2 \end{bmatrix}$

Implement Gaussian elimination for each example:

1. $\left[\begin{array}{ccc|c} 1 & 1 & 1 & 2 \\ 1 & 0 & 1 & 1 \\ 2 & 5 & 2 & 7 \end{array}\right] \longrightarrow \left[\begin{array}{ccc|c} 1 & 1 & 1 & 2 \\ 0 & 1 & 0 & 1 \\ 0 & 0 & 0 & 0 \end{array}\right] \Rightarrow \quad \boldsymbol{x} = \begin{bmatrix} 1-t \\ 1 \\ t \end{bmatrix}, \quad t \in \mathbb{R}$

2. $\left[\begin{array}{cccc|c} 1 & -1 & 1 & -2 & 1 \\ -1 & 1 & 1 & 1 & -1 \\ -1 & 2 & 3 & 1 & 2 \\ 1 & -1 & 2 & 1 & 1 \end{array}\right] \longrightarrow \left[\begin{array}{cccc|c} 1 & -1 & 1 & -2 & 1 \\ 0 & 1 & 4 & -1 & 3 \\ 0 & 0 & 1 & 3 & 0 \\ 0 & 0 & 0 & 1 & 0 \end{array}\right] \Rightarrow \quad \boldsymbol{x} = \begin{bmatrix} 4 \\ 3 \\ 0 \\ 0 \end{bmatrix}$

3. $\left[\begin{array}{cccc|c} 1 & 1 & 2 & 1 & 1 \\ 1 & 0 & 1 & 1 & 1 \\ 0 & 1 & 1 & 0 & 2 \end{array}\right] \longrightarrow \left[\begin{array}{cccc|c} 1 & 1 & 2 & 1 & 1 \\ 0 & 1 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 \end{array}\right] \Rightarrow$ No solution

Definition. The **rank** of a matrix A [KN, p.16] is the number of nonzero rows in the row-echelon form of A.

Example. Let A be $m \times n$ matrix with $\text{rank}(A) = r$. Describe when the linear system $A\boldsymbol{x} = \boldsymbol{b}$ has: a unique solution, infinitely many solutions, or no solutions.

The system $A\boldsymbol{x} = \boldsymbol{b}$ is inconsistent (ie. no solution) if $\text{rank}(A) < \text{rank}([\, A \;\; \boldsymbol{b}\,])$. That is, the row-echelon form of the augmented matrix $[\, A \;\; \boldsymbol{b}\,]$ has a row of the form

$$\left[\begin{array}{ccccc} 0 & 0 & \cdots & 0 & 1 \end{array}\right]$$

which implies $0 = 1$. The system has a unique solution when $\text{rank}(A) = \text{rank}([\, A \;\; \boldsymbol{b}\,])$ and

$\text{rank}(A) = n$. That is, the rank is equal to the number of variables in the system. Finally, the system has infinitely many solutions when $\text{rank}(A) = \text{rank}([\, A \; \boldsymbol{b} \,])$ and $\text{rank}(A) < n$.

1.2 LU and Cholesky Decompositions

Big Idea. The LU decomposition of a matrix A (if it exists) records the row operations of Gaussian elimination in a matrix factorization $A = LU$ where L is a unit lower triangular matrix and U is an upper triangular matrix. If A is symmetric positive definite, then the Cholesky decomposition $A = LL^T$ always exists where L is a lower triangular matrix (not unit lower in general).

Definition. An **elementary matrix** [KN, p.95] is a matrix E obtained from the identity matrix I by an elementary row operation. There are 3 types:

1. Switch rows i and j. For example:

$$\begin{bmatrix} 1 & 0 & 0 \\ 0 & 0 & 1 \\ 0 & 1 & 0 \end{bmatrix} \quad \begin{bmatrix} 0 & 0 & 1 \\ 0 & 1 & 0 \\ 1 & 0 & 0 \end{bmatrix} \quad \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 \end{bmatrix} \quad \begin{bmatrix} 0 & 0 & 0 & 0 & 1 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 1 & 0 & 0 & 0 & 0 \end{bmatrix}$$

2. Multiply row i by c. For example:

$$\begin{bmatrix} 1 & 0 & 0 \\ 0 & 2 & 0 \\ 0 & 0 & 1 \end{bmatrix} \quad \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 3 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad \begin{bmatrix} 1 & 0 & 0 & 0 & 0 \\ 0 & -5 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 \end{bmatrix}$$

3. Add c times row i to row j. For example:

$$\begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 4 & 0 & 1 \end{bmatrix} \quad \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 8 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad \begin{bmatrix} 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & -2 & 0 & 1 \end{bmatrix}$$

Proposition. Let E be an elementary matrix. Then matrix multiplication EA applies the corresponding row operation to A [KN, p.96].

Definition. A **unit lower triangular matrix** (see Wikipedia: Triangular matrix) is a square matrix with ones on the diagonal and zeros above diagonal. For example:

$$\begin{bmatrix} 1 & & & \\ * & 1 & & \\ * & * & 1 & \\ * & * & * & 1 \end{bmatrix}$$

An **atomic lower triangular matrix** has nonzero entries below the diagonal in only one column:

$$L_k = \begin{bmatrix} 1 & & & & & \\ & \ddots & & & & \\ & & 1 & & & \\ & & \ell_{k+1,k} & 1 & & \\ & & \vdots & & \ddots & \\ & & \ell_{m,k} & & & 1 \end{bmatrix}$$

Proposition.

1. Let A be a m by n matrix and let L_k be an atomic lower triangular matrix with nonzero entries in column k. Multiplication $L_k A$ adds $\ell_{i,k}$ times row k of A to row i for each $i = k+1, \ldots, m$.

2. The inverse of an atomic lower triangular matrix:

$$L_k = \begin{bmatrix} 1 & & & & & \\ & \ddots & & & & \\ & & 1 & & & \\ & & \ell_{k+1,k} & 1 & & \\ & & \vdots & & \ddots & \\ & & \ell_{m,k} & & & 1 \end{bmatrix} \quad \Rightarrow \quad L_k^{-1} = \begin{bmatrix} 1 & & & & & \\ & \ddots & & & & \\ & & 1 & & & \\ & & -\ell_{k+1,k} & 1 & & \\ & & \vdots & & \ddots & \\ & & -\ell_{m,k} & & & 1 \end{bmatrix}$$

3. Multiplication of atomic lower triangular matrices:

$$L_1 L_2 \cdots L_{m-1} = \begin{bmatrix} 1 & & & & \\ \ell_{2,1} & 1 & & & \\ \ell_{3,1} & & 1 & & \\ \vdots & & & \ddots & \\ \ell_{m,1} & & & & 1 \end{bmatrix} \begin{bmatrix} 1 & & & & \\ & 1 & & & \\ & \ell_{3,2} & 1 & & \\ & \vdots & & \ddots & \\ & \ell_{m,2} & & & 1 \end{bmatrix} \cdots \begin{bmatrix} 1 & & & & \\ & 1 & & & \\ & & 1 & & \\ & & & \ddots & \\ & & & \ell_{m,m-1} & 1 \end{bmatrix}$$

$$= \begin{bmatrix} 1 & & & & \\ \ell_{2,1} & 1 & & & \\ \ell_{3,1} & \ell_{3,2} & 1 & & \\ \vdots & \vdots & \ddots & \ddots & \\ \ell_{m,1} & \ell_{m,2} & \cdots & \ell_{m,m-1} & 1 \end{bmatrix}$$

Theorem. If A can be reduced by Gaussian elimination to row echelon form without pivoting (that is, without interchanging rows), then

$$A = LU$$

where L is a unit lower triangular matrix and U is an upper triangular matrix. This is called the **LU decomposition** of A [MH, p.68]. The algorithm is:

Let m be the number of rows of A and let n be the number of columns.

Let $A_1 = A$ and use notation $A_k = \left[a_{i,j}^{(k)}\right]$.

for k from 1 to $n-1$:

Let L_k such that $\ell_{i,k} = -\dfrac{a_{i,k}^{(k)}}{a_{k,k}^{(k)}}$ and compute $L_k A_k = A_{k+1}$.

end

$A = LU$ where $L = L_1^{-1} \cdots L_{n-1}^{-1}$ is unit lower triangular and $U = A_n$ is upper triangular

Note. In each step of the LU decomposition algorithm, the operation $L_k A_k$ eliminates entries in A_k below the diagonal in column k. The result is $L_{n-1} \cdots L_1 A = U$ where $U = A_n$ is upper triangular and therefore $A = LU$ where $L = L_1^{-1} \cdots L_{n-1}^{-1}$ is unit lower triangular.

Example. Compute the LU decomposition of

$$A = \begin{bmatrix} 2 & 1 & 1 \\ 2 & 0 & 2 \\ 4 & 3 & 4 \end{bmatrix}$$

Start with $A_1 = A$ and compute

$$L_1 A_1 = \begin{bmatrix} 1 & 0 & 0 \\ -1 & 1 & 0 \\ -2 & 0 & 1 \end{bmatrix} \begin{bmatrix} 2 & 1 & 1 \\ 2 & 0 & 2 \\ 4 & 3 & 4 \end{bmatrix} = \begin{bmatrix} 2 & 1 & 1 \\ 0 & -1 & 1 \\ 0 & 1 & 2 \end{bmatrix}$$

$$L_2 A_2 = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 1 & 1 \end{bmatrix} \begin{bmatrix} 2 & 1 & 1 \\ 0 & -1 & 1 \\ 0 & 1 & 2 \end{bmatrix} = \begin{bmatrix} 2 & 1 & 1 \\ 0 & -1 & 1 \\ 0 & 0 & 3 \end{bmatrix}$$

Therefore $A = LU$ where

$$L = L_1^{-1}L_2^{-1} = \begin{bmatrix} 1 & 0 & 0 \\ 1 & 1 & 0 \\ 2 & -1 & 1 \end{bmatrix} \qquad U = \begin{bmatrix} 2 & 1 & 1 \\ 0 & -1 & 1 \\ 0 & 0 & 3 \end{bmatrix}$$

Note. Suppose A has an LU decomposition $A = LU$. Applications of LU are:

1. Solve the system of equations $A\boldsymbol{x} = \boldsymbol{b}$ by:
   - Solve $L\boldsymbol{y} = \boldsymbol{b}$ by forward substitution
   - Solve $U\boldsymbol{x} = \boldsymbol{y}$ by backward substitution
2. $\text{rank}(A) = \text{rank}(U)$
3. $\det(A) = \det(U) =$ product of diagonal entries of U
4. Compute A^{-1}: use LU decomposition to solve $A\boldsymbol{x}_1 = \boldsymbol{e}_1, \ldots, A\boldsymbol{x}_n = \boldsymbol{e}_n$ where $\boldsymbol{e}_k$ is the kth column of the identity I and then $A^{-1} = [\boldsymbol{x}_1 \cdots \boldsymbol{x}_n]$ (that is, the columns of A^{-1} are given by $\boldsymbol{x}_1, \ldots, \boldsymbol{x}_n$)

Example. Compute the LU decomposition of

$$A = \begin{bmatrix} 1 & 0 & 2 & 1 \\ 2 & 1 & 5 & 3 \\ 1 & 0 & 0 & 2 \\ 0 & -1 & 1 & 1 \end{bmatrix}$$

Start with $A_1 = A$ and compute

$$L_1A_1 = \begin{bmatrix} 1 & 0 & 0 & 0 \\ -2 & 1 & 0 & 0 \\ -1 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 2 & 1 \\ 2 & 1 & 5 & 3 \\ 1 & 0 & 0 & 2 \\ 0 & -1 & 1 & 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 2 & 1 \\ 0 & 1 & 1 & 1 \\ 0 & 0 & -2 & 1 \\ 0 & -1 & 1 & 1 \end{bmatrix}$$

$$L_2A_2 = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 2 & 1 \\ 0 & 1 & 1 & 1 \\ 0 & 0 & -2 & 1 \\ 0 & -1 & 1 & 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 2 & 1 \\ 0 & 1 & 1 & 1 \\ 0 & 0 & -2 & 1 \\ 0 & 0 & 2 & 2 \end{bmatrix}$$

$$L_3A_3 = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 1 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 2 & 1 \\ 0 & 1 & 1 & 1 \\ 0 & 0 & -2 & 1 \\ 0 & 0 & 2 & 2 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 2 & 1 \\ 0 & 1 & 1 & 1 \\ 0 & 0 & -2 & 1 \\ 0 & 0 & 0 & 3 \end{bmatrix}$$

And we compute $A = LU$ using properties of atomic lower triangular matrices

$$L = L_1^{-1}L_2^{-1}L_3^{-1} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 2 & 1 & 0 & 0 \\ 1 & 0 & 1 & 0 \\ 0 & -1 & -1 & 1 \end{bmatrix} \qquad U = \begin{bmatrix} 1 & 0 & 2 & 1 \\ 0 & 1 & 1 & 1 \\ 0 & 0 & -2 & 1 \\ 0 & 0 & 0 & 3 \end{bmatrix}$$

Theorem. Let A be an m by n matrix. If $\text{rank}(A) = m$ and A has a LU decomposition $A = LU$ where L is unit lower triangular and U is upper triangular, then the matrices L and U are unique [KN, p.124].

Definition. Let A be a square matrix.

1. A is **symmetric** if $A^T = A$.
2. A is **positive definite** if $\boldsymbol{x}^T A\boldsymbol{x} > 0$ for all $\boldsymbol{x} \neq 0$.

See [MH, p.84].

Example. If A is invertible, then both AA^T and A^TA are symmetric positive definite.

Theorem. Let A be a symmetric positive definite matrix. There exists a lower triangular matrix L with positive diagonal entries (not necessarily unit lower triangular) such that

$$A = LL^T$$

This is called the **Cholesky decomposition** of A [MH, p.84].

Proposition. Let A be a symmetric positive definite matrix. There exists a *unit* lower triangular matrix L and a diagonal matrix D with *positive* entries such that

$$A = LDL^T$$

Furthermore, let $\sqrt{D}$ be the diagonal matrix where the entries are the square roots of the entries in D (that is, $\sqrt{D}^2 = D$). Then

$$A = (L\sqrt{D})(\sqrt{D}L^T)$$

is the Cholesky decomposition of A.

Example. Computing the Cholesky decomposition requires about half the computations of the LU factorization since we need only compute L or U by Gaussian elimination. Compute the Cholesky decomposition of A where $A = M^T M$ for

$$M = \begin{bmatrix} 1 & 1 & -1 \\ 0 & -1 & -1 \\ 1 & 1 & 1 \end{bmatrix} \quad \Rightarrow \quad A = \begin{bmatrix} 2 & 2 & 0 \\ 2 & 3 & 1 \\ 0 & 1 & 3 \end{bmatrix}$$

We know that A is symmetric positive definite by construction and so we proceed by Gaussian elimination (without scaling or switching rows)

$$\begin{bmatrix} 2 & 2 & 0 \\ 2 & 3 & 1 \\ 0 & 1 & 3 \end{bmatrix} \to \begin{bmatrix} 2 & 2 & 0 \\ 0 & 1 & 1 \\ 0 & 1 & 3 \end{bmatrix} \to \begin{bmatrix} 2 & 2 & 0 \\ 0 & 1 & 1 \\ 0 & 0 & 2 \end{bmatrix}$$

Factor out the diagonal

$$\begin{bmatrix} 2 & 2 & 0 \\ 0 & 1 & 1 \\ 0 & 0 & 2 \end{bmatrix} = \begin{bmatrix} 2 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 2 \end{bmatrix} \begin{bmatrix} 1 & 1 & 0 \\ 0 & 1 & 1 \\ 0 & 0 & 1 \end{bmatrix}$$

Therefore the Cholesky decomposition is $A = LL^T$ where

$$L = \begin{bmatrix} 1 & 0 & 0 \\ 1 & 1 & 0 \\ 0 & 1 & 1 \end{bmatrix} \begin{bmatrix} \sqrt{2} & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & \sqrt{2} \end{bmatrix} = \begin{bmatrix} \sqrt{2} & 0 & 0 \\ \sqrt{2} & 1 & 0 \\ 0 & 1 & \sqrt{2} \end{bmatrix}$$

1.3 LU Decomposition with Partial Pivoting

Big Idea. For any matrix A, Gaussian elimination with partial pivoting computes a decomposition $A = PLU$ where P is a permutation matrix, L is unit lower triangular and U is upper triangular.

Note. At step k in the LU decomposition in the previous section, we form the atomic lower triangular matrix L_k with entries

$$\ell_{i,k} = -\frac{a_{i,k}^{(k)}}{a_{k,k}^{(k)}}$$

This is a problem if $a_{k,k}^{(k)}$ very small or zero! Therefore in general we need to use pivoting. In other words, we interchange rows as necessary at each step. This leads to

$$L_{n-1}P_{n-1}\cdots L_2 P_2 L_1 P_1 A = U \quad \Rightarrow \quad A = P_1^{-1}L_1^{-1}P_2^{-1}L_2^{-1}\cdots P_{n-1}^{-1}L_{n-1}^{-1}U$$

We need to figure out how to rearrange the matrices into the form $A = PLU$.

Proposition. Let L_k be an atomic lower triangular matrix with nonzero entries below the diagonal in column k. Let P be an elementary permutation matrix which switches rows i and j for $i > k$ and $j > k$. Then $PL_kP = \tilde{L}_k$ where $\tilde{L}_k$ is simply L_k but with entries $\ell_{i,k}$ and $\ell_{j,k}$ switched.

Proof. Write L_k in block form

$$L_k = \begin{bmatrix} 1 & & & & \\ & \ddots & & & \\ & & 1 & & \\ & & \ell_{k+1,k} & 1 & \\ & & \vdots & & \ddots \\ & & \ell_{m,k} & & 1 \end{bmatrix} = \left[\begin{array}{c|c} I_k & 0 \\ \hline L_* & I_{m-k} \end{array}\right] \qquad L_* = \begin{bmatrix} 0 & \cdots & 0 & \ell_{k+1,k} \\ \vdots & & \vdots & \vdots \\ 0 & \cdots & 0 & \ell_{m,k} \end{bmatrix}$$

where I_k and I_{m-k} are the identity matrices of size k and $m-k$ respectively. Since P be an elementary permutation matrix which switches rows i and j for $i > k$ and $j > k$, write in block form

$$\left[\begin{array}{c|c} I_k & 0 \\ \hline 0 & P_* \end{array}\right]$$

Since P is an elementary permutation matrix which only switches 2 rows, we have $P^2 = I_m$ and also $P_*^2 = I_{m-k}$. Finally, using block matrices, we compute

$$\begin{aligned} PL_kP &= \left[\begin{array}{c|c} I_k & 0 \\ \hline 0 & P_* \end{array}\right] \left[\begin{array}{c|c} I_k & 0 \\ \hline L_* & I_{m-k} \end{array}\right] \left[\begin{array}{c|c} I_k & 0 \\ \hline 0 & P_* \end{array}\right] \\ &= \left[\begin{array}{c|c} I_k & 0 \\ \hline 0 & P_* \end{array}\right] \left[\begin{array}{c|c} I_k & 0 \\ \hline L_* & P_* \end{array}\right] \\ &= \left[\begin{array}{c|c} I_k & 0 \\ \hline P_*L_* & P_*^2 \end{array}\right] \\ &= \left[\begin{array}{c|c} I_k & 0 \\ \hline P_*L_* & I_{m-k} \end{array}\right] \end{aligned}$$

The result is $PL_kP = \tilde{L}_k$ where $\tilde{L}_k$ is simply L_k but with $\ell_{i,k}$ and $\ell_{j,k}$ switched. □

Example. Consider the matrices

$$P = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 \end{bmatrix} \qquad L = \begin{bmatrix} 1 & 0 & 0 & 0 \\ a & 1 & 0 & 0 \\ b & 0 & 1 & 0 \\ c & 0 & 0 & 1 \end{bmatrix}$$

Since P switches rows 2 and 4, we find

$$PLP = \begin{bmatrix} 1 & 0 & 0 & 0 \\ c & 1 & 0 & 0 \\ b & 0 & 1 & 0 \\ a & 0 & 0 & 1 \end{bmatrix}$$

Theorem. For any matrix A, there exists a permutation matrix P, unit lower triangular matrix L and upper triangular matrix U such that

$$A = PLU$$

This is the **LU decomposition with partial pivoting** [MH, p.72]. The algorithm is:

Let m be the number of rows of A and let n be the number of columns.

Let $A_1 = A$ and use notation $A_k = \left[a_{i,j}^{(k)}\right]$ and $\tilde{A}_k = \left[\tilde{a}_{i,j}^{(k)}\right]$.

for k from 1 to $n-1$:

Find index p with maximum value $\left|a_{p,k}^{(k)}\right|$ in column k of A_k below diagonal

(or if all entries in column k below diagonal are zero, move to the next column).

Let P_k such that $P_k A_k = \tilde{A}_k$ switches rows p and k.

Let L_k such that $\ell_{i,k} = -\dfrac{\tilde{a}_{i,k}^{(k)}}{\tilde{a}_{k,k}^{(k)}}$ and compute $L_k \tilde{A}_k = A_{k+1}$.

end

$A = PLU$ where $P = P_1 \cdots P_{n-1}$, $U = A_n$ and $L = \tilde{L}_1^{-1} \cdots \tilde{L}_{n-1}^{-1}$ where each $\tilde{L}_k^{-1}$ is given by L_k^{-1} with entries permuted in order by $P_{k'}$ for each $k' > k$.

Note. Pivoting ensures all the entries of L below the diagonal satisfy $|\ell_{i,j}| \leq 1$ $(i > j)$.

Example. Compute the LU decomposition with partial pivoting of

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

In the first column, we see the maximum value $|a_{i,1}|$ is 1 and we do not need to pivot therefore

$P_1 = I$ and $\tilde{A}_1 = A_1 = A$. Compute:

$$L_1\tilde{A}_1 = \begin{bmatrix} 1 & 0 & 0 \\ -1 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 2 \\ 0 & 1 & 1 \end{bmatrix} = \begin{bmatrix} 1 & 1 & 1 \\ 0 & 0 & 1 \\ 0 & 1 & 1 \end{bmatrix}$$

In the second column, we see $a^{(2)}_{2,2} = 0$ and $a^{(2)}_{3,2} = 1$ therefore we must swap rows

$$P_2A_2 = \begin{bmatrix} 1 & 1 & 1 \\ 0 & 1 & 1 \\ 0 & 0 & 1 \end{bmatrix}$$

This is already upper triangular and so $L_2 = I$ in the last step. Therefore

$$L_2P_2L_1P_1A = U \quad \Rightarrow \quad A = P_1^{-1}L_1^{-1}P_2^{-1}L_2^{-1}U = L_1^{-1}P_2^{-1}U$$

Using the proposition above we find $L_1^{-1}P_2^{-1} = P_2^{-1}\tilde{L}_1^{-1}$ where

$$\tilde{L}_1^{-1} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 1 & 0 & 1 \end{bmatrix}$$

Finally, we have $A = PLU$ where

$$P = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 0 & 1 \\ 0 & 1 & 0 \end{bmatrix} \qquad L = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 1 & 0 & 1 \end{bmatrix} \qquad U = \begin{bmatrix} 1 & 1 & 1 \\ 0 & 1 & 1 \\ 0 & 0 & 1 \end{bmatrix}$$

Example. Suppose the LU decomposition with partial pivoting applied to A yields matrices

$$P_1 = \begin{bmatrix} 0 & 0 & 0 & 1 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 1 & 0 & 0 & 0 \end{bmatrix} \qquad P_2 = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \qquad P_3 = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{bmatrix}$$

$$L_1 = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0.2 & 1 & 0 & 0 \\ -0.1 & 0 & 1 & 0 \\ 0.5 & 0 & 0 & 1 \end{bmatrix} \qquad L_2 = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0.7 & 1 & 0 \\ 0 & -0.1 & 0 & 1 \end{bmatrix} \qquad L_3 = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0.1 & 1 \end{bmatrix}$$

We rearrange the matrices

$$\begin{aligned} &P_1^{-1}L_1^{-1}P_2^{-1}L_2^{-1}P_3^{-1}L_3^{-1} \\ &= P_1 \begin{bmatrix} 1 & 0 & 0 & 0 \\ -0.2 & 1 & 0 & 0 \\ 0.1 & 0 & 1 & 0 \\ -0.5 & 0 & 0 & 1 \end{bmatrix} P_2 \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & -0.7 & 1 & 0 \\ 0 & 0.1 & 0 & 1 \end{bmatrix} P_3 \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & -0.1 & 1 \end{bmatrix} \end{aligned}$$

$$= P_1P_2 \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0.1 & 1 & 0 & 0 \\ -0.2 & 0 & 1 & 0 \\ -0.5 & 0 & 0 & 1 \end{bmatrix} P_3 \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0.1 & 1 & 0 \\ 0 & -0.7 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & -0.1 & 1 \end{bmatrix}$$

$$= P_1P_2P_3 \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0.1 & 1 & 0 & 0 \\ -0.5 & 0 & 1 & 0 \\ -0.2 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0.1 & 1 & 0 \\ 0 & -0.7 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & -0.1 & 1 \end{bmatrix}$$

$$= P_1P_2P_3 \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0.1 & 1 & 0 & 0 \\ -0.5 & 0.1 & 1 & 0 \\ -0.2 & -0.7 & -0.1 & 1 \end{bmatrix}$$

Therefore we have found P and L in $A = PLU$ where

$$P = P_1P_2P_3 = \begin{bmatrix} 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 1 & 0 & 0 \\ 1 & 0 & 0 & 0 \end{bmatrix} \qquad L = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0.1 & 1 & 0 & 0 \\ -0.5 & 0.1 & 1 & 0 \\ -0.2 & -0.7 & -0.1 & 1 \end{bmatrix}$$

Note. Mathematical software such as MATLAB `linsolve` (see MATLAB documentation) and SciPy `scipy.linalg.solve` (see SciPy documentation) compute solutions of linear systems by LU decomposition with partial pivoting.

1.4 Matrix Norms and the Condition Number

Big Idea. Given a linear system $A\boldsymbol{x} = \boldsymbol{b}$, the condition number of A quantifies how sensitive the solution $\boldsymbol{x}$ is relative to perturbations in $\boldsymbol{b}$.

Definition. A **norm** on $\mathbb{R}^n$ [MH, p.53] is a function $\|\cdot\|$ such that:

1. $\|\boldsymbol{x}\| \geq 0$ for all $\boldsymbol{x} \in \mathbb{R}^n$
2. $\|\boldsymbol{x}\| = 0$ if and only if $\boldsymbol{x} = \boldsymbol{0}$
3. $\|c\boldsymbol{x}\| = |c|\|\boldsymbol{x}\|$ for any $c \in \mathbb{R}$ and $\boldsymbol{x} \in \mathbb{R}^n$
4. $\|\boldsymbol{x} + \boldsymbol{y}\| \leq \|\boldsymbol{x}\| + \|\boldsymbol{y}\|$ for all $\boldsymbol{x}, \boldsymbol{y} \in \mathbb{R}^n$ (the **triangle inequality**)

Example. Let $\boldsymbol{x} = [x_1 \ \cdots \ x_n]^T \in \mathbb{R}^n$.

1. The 2-norm is given by the familiar formula

$$\|\boldsymbol{x}\|_2 = \sqrt{|x_1|^2 + \cdots + |x_n|^2} = \sqrt{\sum_{k=1}^{n} |x_k|^2}$$

2. More generally, the p-norm is given by

$$\|\boldsymbol{x}\|_p = \left(\sum_{k=1}^{n} |x_k|^p \right)^{1/p}$$

   For example, a commonly used norm is the 1-norm

$$\|\boldsymbol{x}\|_1 = |x_1| + \cdots + |x_n| = \sum_{k=1}^{n} |x_k|$$

3. The ∞-norm is given by

$$\|\boldsymbol{x}\|_\infty = \max_k |x_k|$$

Example.

1. Prove that the ∞-norm satisfies the required properties of a norm.
2. Sketch the "unit ball" in $\mathbb{R}^2$ for each norm:

$$B_1 = \{\mathbf{x} \in \mathbb{R}^2 : \|\boldsymbol{x}\|_1 = 1\}$$
$$B_2 = \{\mathbf{x} \in \mathbb{R}^2 : \|\boldsymbol{x}\|_2 = 1\}$$
$$B_\infty = \{\mathbf{x} \in \mathbb{R}^2 : \|\boldsymbol{x}\|_\infty = 1\}$$

3. Which set of inequalities is always true? Explain.

$$\|\boldsymbol{x}\|_1 \leq \|\boldsymbol{x}\|_2 \leq \|\boldsymbol{x}\|_\infty \quad \text{or} \quad \|\boldsymbol{x}\|_1 \geq \|\boldsymbol{x}\|_2 \geq \|\boldsymbol{x}\|_\infty$$

Definition. Choose a vector norm $\|\cdot\|$. The corresponding **matrix norm** (or **operator norm**) [MH, p.54] is

$$\|A\| = \max_{\boldsymbol{x} \neq \mathbf{0}} \frac{\|A\boldsymbol{x}\|}{\|\boldsymbol{x}\|}$$

Note that $\|A\boldsymbol{x}\|/\|\boldsymbol{x}\| = \|A(\boldsymbol{x}/\|\boldsymbol{x}\|)\|$ therefore

$$\|A\| = \max_{\|\boldsymbol{x}\|=1} \|A\boldsymbol{x}\|$$

In other words, the matrix norm is the maximum stretch of a unit vector under the linear transformation A.

Proposition. A matrix norm(corresponding to a vector norm as defined above) satisfies the properties:

1. $\|A\| > 0$ for all $A \neq 0$
2. $\|A\| = 0$ if an only $A = 0$
3. $\|cA\| = |c|\|A\|$ for any $c \in \mathbb{R}$
4. $\|A + B\| \leq \|A\| + \|B\|$
5. $\|AB\| \leq \|A\|\|B\|$
6. $\|A\boldsymbol{x}\| \leq \|A\|\|\boldsymbol{x}\|$ for any $\boldsymbol{x} \in \mathbb{R}^n$

See [MH, p.54].

Definition. The **condition number** (with respect to the matrix norm $\|\cdot\|$) [MH, p.55] of a nonsingular matrix A is

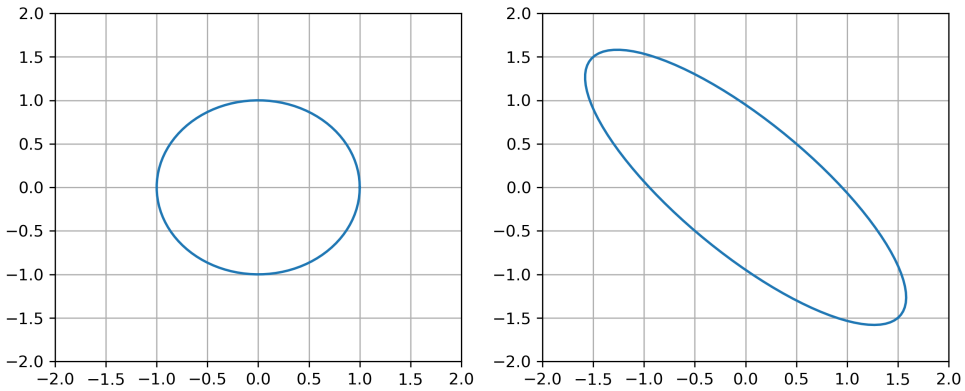
$$\text{cond}(A) = \|A\|\|A^{-1}\|$$

By convention, we define $\text{cond}(A) = \infty$ if $\det(A) = 0$.

Note. If A is nonsingular, we have

$$\begin{aligned}
\text{cond}(A) = \|A\|\|A^{-1}\| &= \max_{\boldsymbol{x}\neq 0} \frac{\|A\boldsymbol{x}\|}{\|\boldsymbol{x}\|} \cdot \max_{\boldsymbol{x}\neq 0} \frac{\|A^{-1}\boldsymbol{x}\|}{\|\boldsymbol{x}\|} \\
&= \max_{\boldsymbol{x}\neq 0} \frac{\|A\boldsymbol{x}\|}{\|\boldsymbol{x}\|} \cdot \max_{\boldsymbol{x}\neq 0} \frac{\|\boldsymbol{x}\|}{\|A\boldsymbol{x}\|} \\
&= \max_{\|\boldsymbol{x}\|=1} \|A\boldsymbol{x}\| \cdot \max_{\|\boldsymbol{x}\|=1} \frac{1}{\|A\boldsymbol{x}\|} \\
&= \frac{\max\limits_{\|\boldsymbol{x}\|=1} \|A\boldsymbol{x}\|}{\min\limits_{\|\boldsymbol{x}\|=1} \|A\boldsymbol{x}\|} = \frac{\text{maximum stretch of a unit vector}}{\text{minimum stretch of a unit vector}}
\end{aligned}$$

Example. The image below shows the unit circle and its image under the linear transformation defined by a 2×2 matrix A. Determine $\|A\|$, $\|A^{-1}\|$ and $\text{cond}(A)$ (with respect to the 2-norm).



Observe the maximum stretch of a unit vector is $\|A\| = 3\sqrt{2}/2$, the minimum stretch $\|A^{-1}\| = \sqrt{2}/2$ and the condition number is $\text{cond}(A) = 3$.

Proposition. Let A be a nonsingular matrix and consider the linear system $A\boldsymbol{x} = \boldsymbol{b}$. If a small change $\Delta\boldsymbol{b}$ corresponds to a change $\Delta\boldsymbol{x}$ in the sense that $A(\boldsymbol{x} + \Delta\boldsymbol{x}) = \boldsymbol{b} + \Delta\boldsymbol{b}$, then

$$\frac{\|\Delta\boldsymbol{x}\|}{\|\boldsymbol{x}\|} \leq \text{cond}(A)\frac{\|\Delta\boldsymbol{b}\|}{\|\boldsymbol{b}\|}$$

See [MH, p.58].

Proof. Since $A\boldsymbol{x} = \boldsymbol{b}$, we have $\Delta x = A^{-1}\Delta\boldsymbol{b}$. Computing norms we find

$$\begin{aligned}
\|\boldsymbol{b}\| &= \|A\boldsymbol{x}\| \\
\|\Delta\boldsymbol{x}\|\|\boldsymbol{b}\| &= \|A^{-1}\Delta\boldsymbol{b}\|\|A\boldsymbol{x}\| \\
\|\Delta\boldsymbol{x}\|\|\boldsymbol{b}\| &\leq \|A^{-1}\|\|\Delta\boldsymbol{b}\|\|A\|\|\boldsymbol{x}\| \\
\frac{\|\Delta\boldsymbol{x}\|}{\|\boldsymbol{x}\|} &\leq \|A\|\|A^{-1}\|\frac{\|\Delta\boldsymbol{b}\|}{\|\boldsymbol{b}\|}
\end{aligned}$$

□

Definition. Given a vector $\boldsymbol{b}$ and small perturbation $\Delta\boldsymbol{b}$, the **relative change** (or **relative error**) is

$$\frac{\|\Delta\boldsymbol{b}\|}{\|\boldsymbol{b}\|}$$

Note. The error bound

$$\frac{\|\Delta\boldsymbol{x}\|}{\|\boldsymbol{x}\|} \leq \text{cond}(A)\frac{\|\Delta\boldsymbol{b}\|}{\|\boldsymbol{b}\|}$$

implies that if A has a large condition number, then small changes in $\boldsymbol{b}$ may result in *very* large

changes in the solution $\boldsymbol{x}$. In other words, the solution is sensitive to errors and is .

1.5 Polynomial Interpolation

Big Idea. Given points $(t_0, y_0), \ldots, (t_d, y_d)$, there exists a unique polynomial $p(t)$ of degree (at most) d such that $p(t_k) = y_k$ for each $k = 0, \ldots, d$.

Definition. Given $d+1$ points $(t_0, y_0), \ldots, (t_d, y_d)$, polynomial interpolation with respect to the **monomial basis** [MH, p.312] seeks a polynomial of the form

$$p(t) = c_0 + c_1 t + \cdots + c_d t^d$$

such that $p(t_k) = y_k$ for each $k = 0, \ldots, d$. The elements $1, t, t^2, \ldots, t^d$ form the monomial basis of the vector space $\mathbb{P}_d$ of polynomials of degree less than or equal to d. Note that each point defines an equation

$$\begin{aligned} c_0 + c_1 t_0 + \cdots + c_d t_0^d &= y_0 \\ c_0 + c_1 t_1 + \cdots + c_d t_1^d &= y_1 \\ &\vdots \\ c_0 + c_1 t_d + \cdots + c_d t_d^d &= y_d \end{aligned}$$

This yields a system of equations $A\boldsymbol{c} = \boldsymbol{y}$

$$\begin{bmatrix} 1 & t_0 & \cdots & t_0^d \\ 1 & t_1 & \cdots & t_1^d \\ \vdots & \vdots & \ddots & \vdots \\ 1 & t_d & \cdots & t_d^d \end{bmatrix} \begin{bmatrix} c_0 \\ c_1 \\ \vdots \\ c_d \end{bmatrix} = \begin{bmatrix} y_0 \\ y_1 \\ \vdots \\ y_d \end{bmatrix}$$

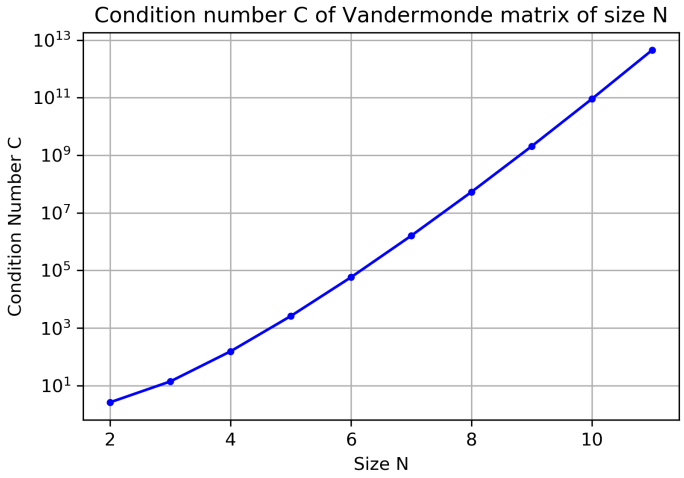
The matrix

$$A = \begin{bmatrix} 1 & t_0 & \cdots & t_0^d \\ 1 & t_1 & \cdots & t_1^d \\ \vdots & \vdots & \ddots & \vdots \\ 1 & t_d & \cdots & t_d^d \end{bmatrix}$$

is called a **Vandermonde matrix**. Solve the system $A\boldsymbol{c} = \boldsymbol{y}$ to find the coefficients

$$\boldsymbol{c} = \begin{bmatrix} c_0 \\ c_1 \\ \vdots \\ c_d \end{bmatrix}$$

Note. The condition number of a Vandermonde matrix gets very large as the size of the matrix increases. This means that interpolation by the monomial basis is very sensitive to changes in the data for polynomials of large degree. For example, for 11 equally spaced points $t_0 = 0, \ldots, t_{10} = 10$, the Vandermonde matrix A is 11 by 11 and has condition number larger than 10^{12}. Yikes!



Proposition. Consider $d+1$ data points $(t_0, y_0), \ldots, (t_d, y_d)$ (such that $t_i \neq t_j$ for $i \neq j$). There exists a unique polynomial $p(t)$ of degree (at most) d such that $p(t_k) = y_k$ for each $k = 0, \ldots, d$.

Proof. The Vandermonde matrix is invertible when the values t_k are distinct therefore there is a unique solution of the system $A\boldsymbol{c} = \boldsymbol{y}$. □

Definition. Given $d+1$ points $(t_0, y_0), \ldots, (t_d, y_d)$, **Lagrange interpolation** [MH, p.315] seeks a polynomial of the form

$$p(t) = c_0\ell_0(t) + c_1\ell_1(t) + \cdots + c_d\ell_d(t)$$

where the **Lagrange basis** $\ell_0(t), \ldots, \ell_d(t)$ is given by

$$\ell_k(t) = \frac{\prod_{j=0, j\neq k}^{d}(t - t_j)}{\prod_{j=0, j\neq k}^{d}(t_k - t_j)}$$

The essential property of these polynomials is

$$\ell_k(t_j) = \begin{cases} 1 & \text{if } k = j \\ 0 & \text{if } k \neq j \end{cases}, \quad k, j = 0, \ldots, d$$

Clearly $c_k = y_k$ for $k = 0, \ldots, d$ and so

$$p(t) = y_0\ell_0(t) + y_1\ell_1(t) + \cdots + y_d\ell_d(t)$$

Definition. Given $d+1$ points $(t_0, y_0), \ldots, (t_d, y_d)$, **Newton interpolation** [MH, p.317] seeks a polynomial of the form

$$p(t) = c_0p_0(t) + c_1p_1(t) + \cdots + c_dp_d(t)$$

where the **Newton basis** $p_0(t), \ldots, p_d(t)$ is given by

$$\begin{aligned} p_0(t) &= 1 \\ p_1(t) &= t - t_0 \\ p_2(t) &= (t - t_0)(t - t_1) \\ &\vdots \\ p_{d-1}(t) &= (t - t_0)(t - t_1)(t - t_2)\cdots(t - t_{d-1}) \end{aligned}$$

Note. Recall that there is a unique polynomial $p(t)$ of degree (at most) d which interpolates $d + 1$ points $(t_0, y_0), \ldots, (t_d, y_d)$ if the values $t_0, \ldots, t_d$ are different. Therefore the monomial, Lagrange and Newton bases all produce the *same* result but computed and represented differently.

Example. Find the interpolating polynomial for $(-1, 1), (0, 0), (1, 1)$ using each of the monomial, Lagrange and Newton bases. We know the result is $p(t) = t^2$. Begin with monomial interpolation and setup the Vandermonde matrix and solve the linear system $A\boldsymbol{c} = \boldsymbol{y}$

$$\begin{bmatrix} 1 & -1 & 1 \\ 1 & 0 & 0 \\ 1 & 1 & 1 \end{bmatrix} \begin{bmatrix} c_0 \\ c_1 \\ c_2 \end{bmatrix} = \begin{bmatrix} 1 \\ 0 \\ 1 \end{bmatrix} \quad \Rightarrow \quad \boldsymbol{c} = \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix}$$

and therefore $c_0 = c_1 = 0$ and $c_2 = 1$ and so $p(t) = t^2$. Now construct the Lagrange basis

$$\begin{aligned} \ell_0(t) &= \frac{(t - 0)(t - 1)}{(-1 - 0)(-1 - 1)} = \frac{t(t - 1)}{2} \\ \ell_1(t) &= \frac{(t - (-1))(t - 1)}{(0 - (-1))(0 - 1)} = 1 - t^2 \\ \ell_2(t) &= \frac{(t - (-1))(t - 0)}{(1 - (-1))(1 - 0)} = \frac{t(t + 1)}{2} \end{aligned}$$

and the interpolating polynomial

$$p(t) = y_0\ell_0(t) + y_1\ell_1(t) + y_2\ell_2(t) = (1)\frac{t(t-1)}{2} + (0)(1-t^2) + (1)\frac{t(t+1)}{2} = t^2$$

Now construct the Newton basis

$$\begin{aligned} p_0(t) &= 1 \\ p_1(t) &= t - (-1) = t + 1 \\ p_2(t) &= (t-(-1))(t-0) = t^2 + t \end{aligned}$$

Each point yields an equation $p(t_k) = y_k$ for $k = 0, 1, 2$ and so we solve the linear system

$$\begin{bmatrix} 1 & 0 & 0 \\ 1 & t_1 - t_0 & 0 \\ 1 & t_2 - t_0 & (t_2 - t_0)(t_2 - t_1) \end{bmatrix} \begin{bmatrix} c_0 \\ c_1 \\ c_2 \end{bmatrix} = \begin{bmatrix} 1 \\ 0 \\ 1 \end{bmatrix}$$

$$\begin{bmatrix} 1 & 0 & 0 \\ 1 & 1 & 0 \\ 1 & 2 & 2 \end{bmatrix} \begin{bmatrix} c_0 \\ c_1 \\ c_2 \end{bmatrix} = \begin{bmatrix} 1 \\ 0 \\ 1 \end{bmatrix} \quad \Rightarrow \quad \begin{bmatrix} c_0 \\ c_1 \\ c_2 \end{bmatrix} = \begin{bmatrix} 1 \\ -1 \\ 1 \end{bmatrix}$$

The interpolating polynomial is again

$$p(t) = c_0p_0(t) + c_1p_1(t) + c_2p_2(t) = 1 - (t+1) + t^2 + t = t^2$$

1.6 Cubic Spline Interpolation

Big Idea. Given $N+1$ points $(t_0, y_0), \ldots, (t_N, y_N)$, a cubic spline is a piecewise cubic polynomial defined by a different polynomial $p_k(t)$ for each subinterval $[t_{k-1}, t_k]$, $k = 1, \ldots, N$.

Definition. Consider $N+1$ points $(t_0, y_0), \ldots, (t_N, y_N)$. A **cubic spline** [MH, p.326] interpolating the data is a function $p(t)$ defined piecewise by N cubic polynomials $p_1(t), \ldots, p_N(t)$ where

$$p_k(t) = a_k(t - t_{k-1})^3 + b_k(t - t_{k-1})^2 + c_k(t - t_{k-1}) + d_k \quad , \quad t \in [t_{k-1}, t_k]$$

such that $p(t)$, $p'(t)$ and $p''(t)$ are continuous functions.

Note. Each polynomial $p_k(t)$ has 4 coefficients a_k, b_k, c_k, d_k therefore we require $4N$ equations to specify the $4N$ unknowns:

1. Interpolation at left endpoints: $p_k(t_{k-1}) = y_{k-1}$ for $k = 1, \ldots, N$ yields N equations.
2. Interpolation at right endpoints: $p_k(t_k) = y_k$ for $k = 1, \ldots, N$ yields N equations.
3. Continuity of $p'(t)$: $p'_k(t_k) = p'_{k+1}(t_k)$ for $k = 1, \ldots, N-1$ yields $N-1$ equations.

4. Continuity of $p''(t)$: $p_k''(t_k) = p_{k+1}''(t_k)$ for $k = 1, \ldots, N-1$ yields $N-1$ equations.

The conditions impose only $4N-2$ equations therefore we need 2 more to determine the cubic spline uniquely. There are different choices such as the natural spline and the "not-a-knot" condition. See [MH, p.327].

Definition. The **natural cubic spline** [MH, p.327] requires $p_1''(t_0) = p_N''(t_N) = 0$.

Definition. Represent a cubic spline $p(t)$ by the **coefficient matrix**

$$C = \begin{bmatrix} a_1 & a_2 & \cdots & a_N \\ b_1 & b_2 & \cdots & b_N \\ c_1 & c_2 & \cdots & c_N \\ d_1 & d_2 & \cdots & d_N \end{bmatrix}$$

where the kth column of C consists of the coefficients for the kth cubic polynomial in the spline

$$p_k(t) = a_k(t - t_{k-1})^3 + b_k(t - t_{k-1})^2 + c_k(t - t_{k-1}) + d_k \quad , \quad t \in [t_{k-1}, t_k]$$

Proposition. Consider $N+1$ points $(t_0, y_0), \ldots, (t_N, y_N)$ (with $t_i \neq t_j$ for $i \neq j$). The unique natural cubic spline $p(t)$ which interpolates the points is given by the coefficient matrix

$$C = \begin{bmatrix} a_1 & a_2 & \cdots & a_N \\ b_1 & b_2 & \cdots & b_N \\ c_1 & c_2 & \cdots & c_N \\ d_1 & d_2 & \cdots & d_N \end{bmatrix}$$

where $d_k = y_{k-1}$ for $k = 1, \ldots, N$ and the coefficients $a_1, b_1, c_1, \ldots, a_N, b_N, c_N$ are the solution of the linear system

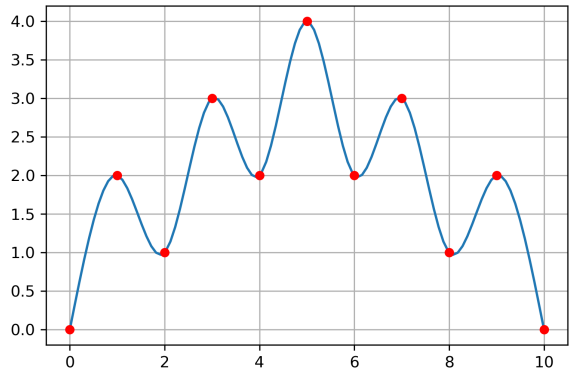
$$\left[\begin{array}{c|c|c|c|c} A(L_1) & B & & & \\ \hline & A(L_2) & B & & \\ \hline & & \ddots & \ddots & \\ \hline & & & A(L_{N-1}) & B \\ \hline T & & & & V \end{array}\right] \begin{bmatrix} a_1 \\ b_1 \\ c_1 \\ \vdots \\ a_N \\ b_N \\ c_N \end{bmatrix} = \begin{bmatrix} y_1 - y_0 \\ 0 \\ 0 \\ \vdots \\ y_N - y_{N-1} \\ 0 \\ 0 \end{bmatrix}$$

where $L_k = t_k - t_{k-1}$ is the length of the subinterval $[t_{k-1}, t_k]$ and

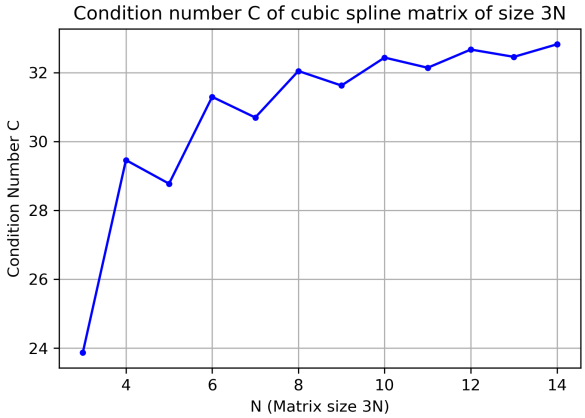
$$A(L) = \begin{bmatrix} L^3 & L^2 & L \\ 3L^2 & 2L & 1 \\ 6L & 2 & 0 \end{bmatrix} \quad B = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & -1 \\ 0 & -2 & 0 \end{bmatrix} \quad T = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 2 & 0 \\ 0 & 0 & 0 \end{bmatrix} \quad V = \begin{bmatrix} L_N^3 & L_N^2 & L_N \\ 0 & 0 & 0 \\ 6L_N & 2 & 0 \end{bmatrix}$$

Example. Construct the natural cubic spline interpolating points

$$(0,0), (1,2), (2,1), (3,3), (4,2), (5,4), (6,2), (7,3), (8,1), (9,2), (10,0)$$



Note. The condition number of the matrix for constructing the natural cubic spline does not increase as drastically with the number of points $N + 1$ as compared with the Vandrmonde matrix. For example, for 11 equally spaced points $t_0 = 0, \ldots, t_{10} = 10$, the Vandermonde matrix is 11 by 11 and has $\text{cond}(A) \approx 10^{12}$ whereas the cubic spline matrix is 30 by 30 and the condition number is only around 33.



1.7 Finite Difference Method

Big Idea. Most differential equations are impossible to solve exactly and so we use numerical methods such as the finite difference method to approximate solutions. The finite difference method applied to a linear differential equations yields a linear system of equations $A\boldsymbol{y} = \boldsymbol{b}$.

Definition. An **ordinary differential equation** is an equation involving an unknown function $y(t)$ and its derivatives. The **order** of a differential equation is the highest order derivative appearing in the equation. There are many kinds of differential equations. In this section, we consider only **second order linear ordinary differential equations**

$$y'' + p(t)y' + q(t)y = r(t)$$

Boundary conditions are equations imposed on the solution at the boundary points t_0 and t_f. For example, specify values of the solution $y(t)$ at the endpoints

$$y(t_0) = \alpha \quad y(t_f) = \beta$$

or specify a value of the solution $y(t)$ at one endpoint and a value of the derivative $y'(t)$ at the other endpoint

$$y'(t_0) = \alpha \quad y(t_f) = \beta$$

Definition. The **Taylor series** of a smooth function $f(x)$ centered at $x = a$ is

$$f(x) = \sum_{n=0}^{\infty} \frac{f^{(n)}(a)}{n!}(x-a)^n = f(a) + f'(a)(x-a) + \frac{f''(a)}{2}(x-a)^2 + \frac{f'''(a)}{6}(x-a)^3 + \cdots$$

Definition. **Finite difference formulas** are derived from Taylor series. Let $y(t)$ be a smooth function and consider the Taylor series

$$y(t+h) = y(t) + y'(t)h + \frac{y''(t)}{2}h^2 + \cdots$$
$$y(t-h) = y(t) - y'(t)h + \frac{y''(t)}{2}h^2 + \cdots$$

Truncate and rearrange the first series for the **forward difference formula**

$$y'(t) \approx \frac{y(t+h) - y(t)}{h}$$

Truncate and rearrange the second series for the **backward difference formula**

$$y'(t) \approx \frac{y(t) - y(t-h)}{h}$$

Subtract $y(t+h) - y(t-h)$ and truncate to get the (first order) **central difference formula**

$$y'(t) \approx \frac{y(t+h) - y(t-h)}{2h}$$

Add $y(t+h) + y(t-h)$ and truncate to get the (second order) **central difference formula**

$$y''(t) \approx \frac{y(t+h) - 2y(t) + y(t-h)}{h^2}$$

Definition. The **finite difference method** applied to a second order linear ordinary differential equation with boundary conditions

$$y'' + p(t)y' + q(t)y = r(t) \quad , \quad t \in [t_0, t_f]$$

1. Discretize the domain: choose N, let $h = \dfrac{t_f - t_0}{N+1}$ and define $t_k = t_0 + kh$.
2. Let $y_k \approx y(t_k)$ denote the approximation of the solution at t_k.
3. Substitute finite difference formulas into the equation to define an equation at each t_k.
4. Rearrange the system of equations into a linear system $A\boldsymbol{y} = \boldsymbol{b}$ and solve for

$$\boldsymbol{y} = \begin{bmatrix} y_1 & y_2 & \cdots & y_N \end{bmatrix}^T$$

Example. Consider a second order linear ordinary differential equation with boundary conditions of the form

$$y'' = r(t) \quad , \quad y(t_0) = \alpha \ , \quad y(t_f) = \beta$$

Choose N and let $h = \dfrac{t_f - t_0}{N+1}$ and define $t_k = t_0 + kh$. Let y_k denote an approximation of $y(t_k)$. Note that the boundary conditions give us $y_0 = \alpha$ and $y_{N+1} = \beta$ and let

$$\boldsymbol{y} = \begin{bmatrix} y_1 & y_2 & \cdots & y_N \end{bmatrix}^T$$

Let $r_k = r(t_k)$ and substitute the central difference formula at t_k into the differential equation

$$\frac{y_{k+1} - 2y_k + y_{k-1}}{h^2} = r_k$$

Therefore we have N equations and N unknowns y_k for $k = 1, \ldots, N$. Use the boundary

conditions $y_0 = \alpha$ and $y_{N+1} = \beta$ and rearrange the equations

$$\begin{array}{rcrcrcrcrcl}
-2y_1 & + & y_2 & & & & & & & = & h^2 r_1 - \alpha \\
y_1 & - & 2y_2 & + & y_3 & & & & & = & h^2 r_2 \\
 & & y_2 & - & 2y_3 & + & y_4 & & & = & h^2 r_3 \\
 & & & & & & \ddots & & & \vdots & \\
 & & & & & & y_{N-2} & - & 2y_{N-1} + y_N & = & h^2 r_{N-1} \\
 & & & & & & & & y_{N-1} - 2y_N & = & h^2 r_N - \beta
\end{array}$$

Rewrite in matrix form $A\boldsymbol{y} = \boldsymbol{b}$ where

$$A = \begin{bmatrix} -2 & 1 & & & \\ 1 & -2 & 1 & & \\ & & \ddots & & \\ & & 1 & -2 & 1 \\ & & & 1 & -2 \end{bmatrix} \qquad \boldsymbol{b} = \begin{bmatrix} h^2 r_1 - \alpha \\ h^2 r_2 \\ \vdots \\ h^2 r_{N-1} \\ h^2 r_N - \beta \end{bmatrix}$$

Example. Setup a linear system $A\boldsymbol{y} = \boldsymbol{b}$ for the equation with boundary conditions

$$y'' = -2 \quad , \quad y(0) = 0 \ , \quad y(1) = 0$$

using step size $h = 0.2$.

The step size h corresponds to $N = 4$ in our formulation, and $r(t) = -2$, $\alpha = \beta = 0$ therefore

$$\begin{bmatrix} -2 & 1 & & \\ 1 & -2 & 1 & \\ & 1 & -2 & 1 \\ & & 1 & -2 \end{bmatrix} \begin{bmatrix} y_1 \\ y_2 \\ y_3 \\ y_4 \end{bmatrix} = \begin{bmatrix} -0.08 \\ -0.08 \\ -0.08 \\ -0.08 \end{bmatrix}$$

Solve the system to find

$$\begin{bmatrix} y_1 \\ y_2 \\ y_3 \\ y_4 \end{bmatrix} = \begin{bmatrix} 0.16 \\ 0.24 \\ 0.24 \\ 0.16 \end{bmatrix}$$

The equation is very simple and we can solve exactly by integrating twice

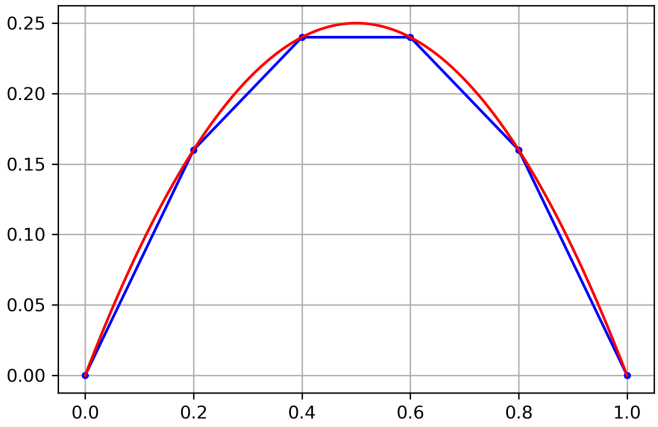
$$y(t) = -t^2 + C_1 t + C_2$$

The boundary conditions imply $C_1 = 1$ and $C_2 = 0$ and therefore the exact solution is

$$y(t) = t - t^2$$

Notice that our finite difference approximation found the exact values $y(0.2) = 0.16$, $y(0.4) = 0.24$, $y(0.6) = 0.24$, $y(0.8) = 0.16$. This is because our equation is very simple and the solution

is a polynomial of degree 2. The finite difference method does not compute exact values in general.



Example. Setup a linear system $A\boldsymbol{y} = \boldsymbol{b}$ for the equation with boundary conditions

$$y'' = \cos(t) \quad , \quad y(0) = 0 \ , \quad y(2\pi) = 1$$

using 7 equally spaced points from $t_0 = 0$ to $t_f = 2\pi$.

The value $N = 5$ corresponds to 7 equally spaced points in our formulation with step size

$$h = \frac{t_f - t_0}{N+1} = \frac{2\pi - 0}{5+1} = \frac{\pi}{3}$$

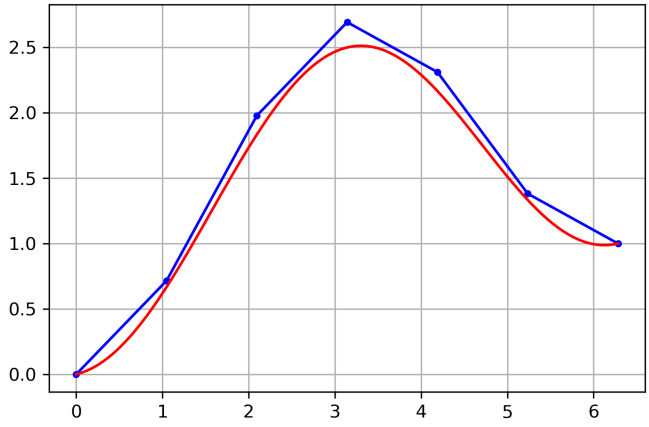
We have $r(t) = \cos(t)$ and $\alpha = 0$ and $\beta = 1$. Note that $r_k = \cos(k\pi/3)$ therefore

$$\begin{bmatrix} -2 & 1 & & & \\ 1 & -2 & 1 & & \\ & 1 & -2 & 1 & \\ & & 1 & -2 & 1 \\ & & & 1 & -2 \end{bmatrix} \begin{bmatrix} y_1 \\ y_2 \\ y_3 \\ y_4 \\ y_5 \end{bmatrix} = \begin{bmatrix} \pi^2/18 \\ -\pi^2/18 \\ -1 \\ -\pi^2/18 \\ \pi^2/18 - 1 \end{bmatrix}$$

Use `scipy.linalg.solve` to compute the solution. The equation is elementary and we can solve exactly by integrating twice

$$y(t) = 1 - \cos(t) + \frac{t}{2\pi}$$

Plot the exact solution together with our approximation



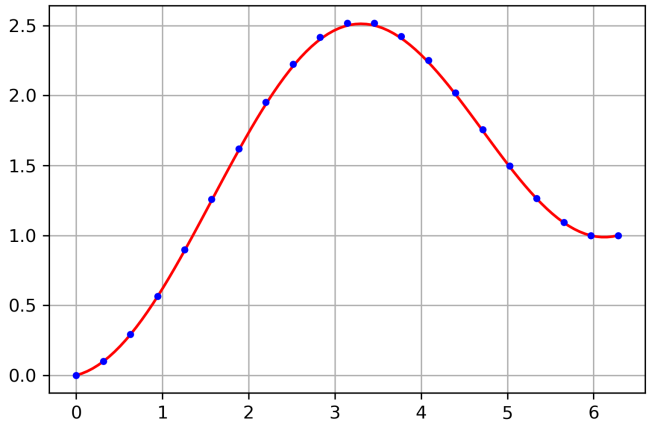
Note. Increasing the number of points in the discretization (equivalently, decreasing the step size h) decreases the error but increases the number of computations. This is a general principle in numerical computing: *higher accuracy requires more computations.* For example, consider the same equation as the previous example

$$y'' = \cos(t) \quad , \quad y(0) = 0 \ , \quad y(2\pi) = 1$$

but now use 21 equally spaced points from $t_0 = 0$ to $t_f = 2\pi$. Then $N = 19$ and

$$h = \frac{t_f - t_0}{N+1} = \frac{2\pi - 0}{19+1} = \frac{\pi}{10}$$

and the finite difference method produces a much better solution



Example. Consider the general form of a second order linear ordinary differential equation with boundary conditions

$$y'' + p(t)y' + q(t)y = r(t) \quad , \quad y(t_0) = \alpha \ , \quad y(t_f) = \beta$$

Choose N and let $h = \dfrac{t_f - t_0}{N+1}$ and define $t_k = t_0 + kh$. Let y_k denote an approximation of $y(t_k)$. Note that the boundary conditions give us $y_0 = \alpha$ and $y_{N+1} = \beta$ and let

$$\boldsymbol{y} = \begin{bmatrix} y_1 & y_2 & \cdots & y_N \end{bmatrix}^T$$

Let $p_k = p(t_k)$, $q_k = q(t_k)$ and $r_k = r(t_k)$, and substitute the central difference formulas for both y'' and y' at t_k into the differential equation

$$\frac{y_{k+1} - 2y_k + y_{k-1}}{h^2} + p_k \frac{y_{k+1} - y_{k-1}}{2h} + q_k y_k = r_k$$

Rearrange the equation

$$y_{k+1} - 2y_k + y_{k-1} + \frac{hp_k}{2}\left(y_{k+1} - y_{k-1}\right) + h^2 q_k y_k = h^2 r_k$$
$$\left(1 - \frac{hp_k}{2}\right) y_{k-1} + (h^2 q_k - 2) y_k + \left(1 + \frac{hp_k}{2}\right) y_{k+1} = h^2 r_k$$

Introduce the notation

$$a_k = 1 - \frac{hp_k}{2} \qquad b_k = h^2 q_k - 2 \qquad c_k = 1 + \frac{hp_k}{2}$$

Use the boundary conditions $y_0 = \alpha$ and $y_{N+1} = \beta$ and rearrange the equations

$$\begin{array}{rcrcrcrcl}
b_1 y_1 & + & c_1 y_2 & & & & & = & h^2 r_1 - (1 - hp_1/2)\,\alpha \\
a_2 y_1 & + & b_2 y_2 & + & c_2 y_3 & & & = & h^2 r_2 \\
 & & & & \ddots & & & \vdots & \\
 & & & & a_{N-1} y_{N-2} & + & b_{N-1} y_{N-1} + c_{N-1} y_N & = & h^2 r_{N-1} \\
 & & & & & & a_N y_{N-1} + b_N y_N & = & h^2 r_N - (1 + hp_N/2)\,\beta
\end{array}$$

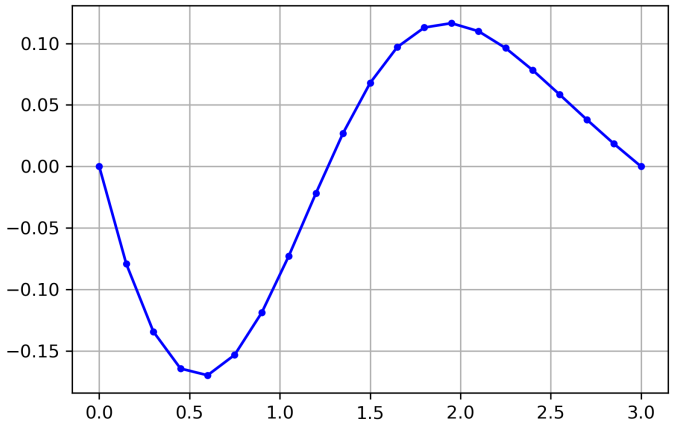
Rewrite in matrix form $A\boldsymbol{y} = \boldsymbol{b}$ where

$$A = \begin{bmatrix}
b_1 & c_1 & & & \\
a_2 & b_2 & c_2 & & \\
 & & \ddots & & \\
 & & a_{N-1} & b_{N-1} & c_{N-1} \\
 & & & a_N & b_N
\end{bmatrix} \qquad \boldsymbol{b} = \begin{bmatrix}
h^2 r_1 - (1 - hp_1/2)\,\alpha \\
h^2 r_2 \\
\vdots \\
h^2 r_{N-1} \\
h^2 r_N - (1 + hp_N/2)\,\beta
\end{bmatrix}$$

Example. Consider the differential equation with boundary conditions

$$y'' + t^2 y' + y = \cos(t) \quad , \quad y(0) = 0 \ , \quad y(3) = 0$$

Solving the linear system derived above with $N = 19$ produces the result



Example. Consider a second order linear ordinary differential equation with boundary conditions of the form

$$y'' = r(t) \quad , \quad y'(t_0) = \alpha \ , \quad y(t_f) = \beta$$

Note that the boundary condition at t_0 specifies the value of the derivative $y'(t_0) = \alpha$. Use the same notation as in the examples above and apply the central difference formula

$$y_{k-1} - 2y_k + y_{k+1} = h^2 r_k$$

Therefore we have N equations

$$\begin{array}{ccccccccccc}
y_0 & - & 2y_1 & + & y_2 & & & & & = & h^2 r_1 \\
 & & y_1 & - & 2y_2 & + & y_3 & & & = & h^2 r_2 \\
 & & & & & & \ddots & & & \vdots & \\
 & & & & & & y_{N-2} & - & 2y_{N-1} + y_N & = & h^2 r_{N-1} \\
 & & & & & & & & y_{N-1} - 2y_N + y_{N+1} & = & h^2 r_N
\end{array}$$

We can use $y_{N+1} = \beta$ and move the term to the right side in the last equation but y_0 in the first equation is unknown. Use the forward difference formula to approximate y_0

$$y'(t_0) \approx \frac{y(t_1) - y(t_0)}{h} \quad \Rightarrow \quad y_0 = y_1 - h\alpha$$

Therefore we can write the equations in matrix form $A\boldsymbol{y} = \boldsymbol{b}$ where

$$A = \begin{bmatrix} -1 & 1 & & & \\ 1 & -2 & 1 & & \\ & & \ddots & & \\ & & 1 & -2 & 1 \\ & & & 1 & -2 \end{bmatrix} \qquad \boldsymbol{b} = \begin{bmatrix} h^2 r_1 + h\alpha \\ h^2 r_2 \\ \vdots \\ h^2 r_{N-1} \\ h^2 r_N - \beta \end{bmatrix}$$

1.8 Exercises

1. Determine whether the statement is **True** or **False**.

   (a) Let A be a m by n matrix such that $m > n$ and $\text{rank}(A) = n$. There is a unique solution of $A\boldsymbol{x} = \boldsymbol{b}$ for any $\boldsymbol{b}$.

   (b) Let A be a m by n matrix such that $m < n$ and $\text{rank}(A) = m$. There are infinitely many solutions of $A\boldsymbol{x} = \boldsymbol{b}$ for any $\boldsymbol{b}$.

   (c) Let A be a m by n matrix such that $m > n$ and $\text{rank}(A) = n$. If the system $A\boldsymbol{x} = \boldsymbol{b}$ has one solution then there is only one solution.

   (d) Let A be a m by n matrix such that $m > n$ and $\text{rank}(A) < n$. If the system $A\boldsymbol{x} = \boldsymbol{b}$ has one solution then there are infinitely many solutions.

   (e) If P is a permutation matrix such that PA interchanges 2 rows of A, then $P^2 = I$.

   (f) If $A = PLU$ is the LU decomposition of A with partial pivoting, then $|\ell_{i,j}| \leq 1$ for all $i > j$ where $\ell_{i,j}$ denotes the entry of L at row i and index j.

   (g) If $A = LU$ is the LU decomposition of A without pivoting, then $|\ell_{i,j}| \leq 1$ for all $i > j$ where $\ell_{i,j}$ denotes the entry of L at row i and index j.

   (h) If $A = LU$ is the LU decomposition of A then $\det(L) \neq 0$.

   (i) If P is any permutation matrix and L is unit lower triangular, then $P^{-1}LP$ is also unit lower triangular.

   (j) If A is of the form

   $$A = \begin{bmatrix} * & * & 0 & 0 \\ * & * & * & 0 \\ 0 & * & * & * \\ 0 & 0 & * & * \end{bmatrix}$$

   and the LU decomposition $A = LU$ exists, then L and U are of the form

   $$L = \begin{bmatrix} 1 & 0 & 0 & 0 \\ * & 1 & 0 & 0 \\ 0 & * & 1 & 0 \\ 0 & 0 & * & 1 \end{bmatrix} \qquad U = \begin{bmatrix} * & * & 0 & 0 \\ 0 & * & * & 0 \\ 0 & 0 & * & * \\ 0 & 0 & 0 & * \end{bmatrix}$$

   (k) If $\boldsymbol{x} \in \mathbb{R}^n$ such that $\|\boldsymbol{x}\|_1 = \|\boldsymbol{x}\|_\infty = \lambda$ then $\|\boldsymbol{x}\|_p = \lambda$ for any $p > 1$.

   (l) If $\|A\| = 1$ then $A = I$.

   (m) Define $\|\boldsymbol{x}\|_0 = \sum_{k=1}^n x_k^2$ for any $\boldsymbol{x} = \begin{bmatrix} x_1 & \cdots & x_n \end{bmatrix}^T \in \mathbb{R}^n$. Then $\|\boldsymbol{x}\|_0$ is a norm.

   (n) Define $\|\boldsymbol{x}\|_{\min} = \min_k |x_k|$ for any $\boldsymbol{x} = \begin{bmatrix} x_1 & \cdots & x_n \end{bmatrix}^T \in \mathbb{R}^n$. Then $\|\boldsymbol{x}\|_{\min}$ is a norm.

   (o) Consider $N + 1$ data points $(t_0, y_0), \ldots, (t_N, y_N)$. Let $A_1\boldsymbol{c}_1 = \boldsymbol{b}_1$ be the linear system such that the solution $\boldsymbol{c}_1$ consists of the coefficients of the interpolating polynomial with respect to the monomial basis. Let $A_2\boldsymbol{c}_2 = \boldsymbol{b}_2$ be the linear system such that the solution $\boldsymbol{c}_2$ consists of the coefficients of the interpolating natural cubic spline. Then we expect $\text{cond}(A_1) < \text{cond}(A_2)$ for large values of N.

   (p) Consider d data points $(t_1, y_1), \ldots, (t_d, y_d)$ (such that $t_i \neq t_j$). There exists a unique polynomial of degree (at most) $d - 1$ which interpolates the data.

(q) Consider d data points $(t_1, y_1), \ldots, (t_d, y_d)$ (such that $t_i \neq t_j$). There exists a unique polynomial $p(t)$ of degree (at most) d which interpolates the data and also satisfies $p'(t_1) = 0$ and $p''(t_1) = 0$.

(r) Consider d data points $(t_1, y_1), \ldots, (t_d, y_d)$ (such that $t_i \neq t_j$). There exists a unique polynomial $p(t)$ of degree (at most) d which interpolates the data and $p'(t_1) = 0$.

(s) Consider $d+1$ data points $(t_0, y_0), (t_1, y_1), \ldots, (t_d, y_d)$. Let $p(t)$ be the interpolating polynomial constructed using the monomial basis. Let $q(t)$ be the interpolating polynomial constructed using the Lagrange basis. Let $r(t)$ be the interpolating polynomial constructed using the Newton basis. Then $p(t) = q(t) = r(t)$ for all t.

(t) The finite difference method applied to a linear second order differential equation with boundary conditions will compute exact values of the solution $y_k = y(t_k)$ if the step size h is chosen to be small enough.

(u) The finite difference method applied to a linear second order differential equation with boundary conditions will never compute exact values of the solution $y_k = y(t_k)$ for any differential equation and step size h.

2. Let I be the identity matrix of size n and let R be the n by n matrix with all zeros except for the nonzero scalar c at index (i, j) where $i \neq j$. That is, the entry of R in row i and column j is c and all other entries of R are 0. Let $E = I + R$ and let A be any n by n matrix.

   (a) Matrix multiplication EA is equivalent to which elementary row/column operation on A?

   (b) Matrix multiplication AE is equivalent to which elementary row/column operation on A?

3. Find a value c such that the system $A\boldsymbol{x} = \boldsymbol{b}$ has infinitely many solutions where

$$A = \begin{bmatrix} 3 & -1 & 2 \\ 1 & 1 & -1 \\ 2 & -2 & 3 \end{bmatrix} \qquad \boldsymbol{b} = \begin{bmatrix} 3 \\ 2 \\ c \end{bmatrix}$$

4. Suppose Gaussian elimination without pivoting applied to a matrix A produces the result

$$L_3 L_2 L_1 A = U$$

where

$$L_1 = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 \\ 3 & 0 & 1 & 0 \\ 2 & 0 & 0 & 1 \end{bmatrix} \qquad L_2 = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 2 & 1 & 0 \\ 0 & 5 & 0 & 1 \end{bmatrix} \qquad L_3 = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 4 & 1 \end{bmatrix}$$

Determine the matrix L in the decomposition $A = LU$.

5. Suppose we perform 2 iterations of Gaussian elimination with partial pivoting on a matrix A such that the result is

$$L_2 P_2 L_1 P_1 A = A_3 = \begin{bmatrix} * & * & * & * \\ 0 & * & * & * \\ 0 & 0 & 2 & 1 \\ 0 & 0 & 5 & 4 \end{bmatrix}$$

Determine the matrices P_3 and L_3 in the third step of the algorithm.

6. Let P be the permutation matrix which moves row 3 to row 4, row 4 to row 5 and row 5 to row 3, and let

$$L = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & a & 1 & 0 & 0 \\ 0 & b & 0 & 1 & 0 \\ 0 & c & 0 & 0 & 1 \end{bmatrix}$$

Determine PLP^{-1}.

7. Suppose A is a symmetric positive definite matrix such that $A = LU$ where L is unit lower triangular and

$$U = \begin{bmatrix} 4 & 1 & 0 & 3 \\ 0 & 1 & 2 & 1 \\ 0 & 0 & 9 & 5 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

Find the Cholesky decomposition of A.

8. Suppose Gaussian elimination with partial pivoting applied to a matrix A produces the result

$$L_2 P_2 L_1 P_1 A = U$$

where

$$L_1 = \begin{bmatrix} 1 & 0 & 0 \\ 0.5 & 1 & 0 \\ 0.7 & 0 & 1 \end{bmatrix} \quad P_1 = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 0 & 1 \\ 0 & 1 & 0 \end{bmatrix} \quad L_2 = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0.2 & 1 \end{bmatrix} \quad P_2 = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

Determine the matrices P and L in the decomposition $A = PLU$.

9. Suppose A is a square matrix with LU decomposition $A = LU$ (where L is unit lower triangular). Describe a method to compute $\det(A)$. Be specific.

10. Compute the Cholesky factorization of the matrix

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

11. Find the solution of the system $A\boldsymbol{x} = \boldsymbol{b}$ for

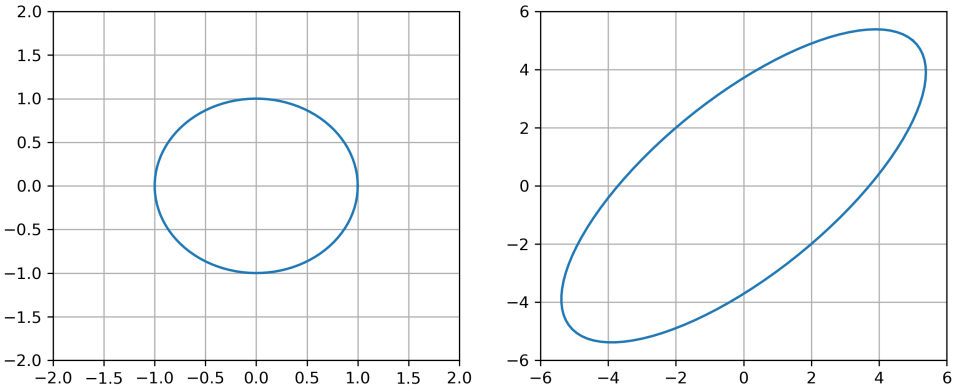
$$A = \begin{bmatrix} 3 & 0 & 1 \\ -3 & -1 & 0 \\ 3 & -1 & 3 \end{bmatrix} \quad \boldsymbol{b} = \begin{bmatrix} 1 \\ -1 \\ 1 \end{bmatrix}$$

given the LU factorization

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

12. Suppose we compute a decomposition $A = L_0U_0$ such that U_0 is *unit* upper triangular and L_0 is lower triangular. Describe a method to derive a decomposition $A = LU$ such that L is *unit* lower triangular and U is upper triangular.

13. Suppose A is a 2 by 2 matrix such that the image of the unit circle under the linear transformation given by A is:



Determine $\text{cond}_2(A)$ (the condition number with respect to the 2-norm).

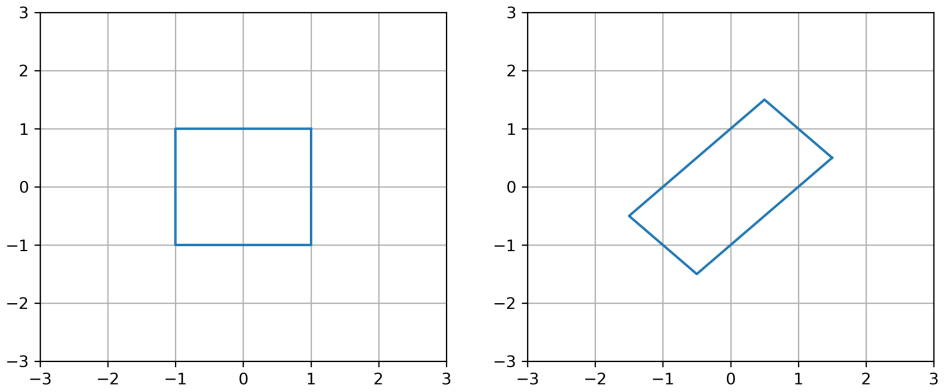
14. Suppose $\boldsymbol{x} = \begin{bmatrix}1 & 2 & 2 & c & 1 & 1\end{bmatrix}^T \in \mathbb{R}^6$ such that $\|\boldsymbol{x}\|_3 = 3$. Find all possible values c.

15. Suppose $\boldsymbol{x} = \begin{bmatrix}1 & 3 & 2 & 2 & c & 2 & 1 & 2\end{bmatrix}^T \in \mathbb{R}^8$ such that $\|\boldsymbol{x}\|_3 = 5$. Find all possible values c.

16. Suppose $\boldsymbol{x} = \begin{bmatrix}1 & 0 & 2 & c & -1\end{bmatrix}^T \in \mathbb{R}^5$ such that $\|\boldsymbol{v}\|_1 = 5$. Find all possible values c.

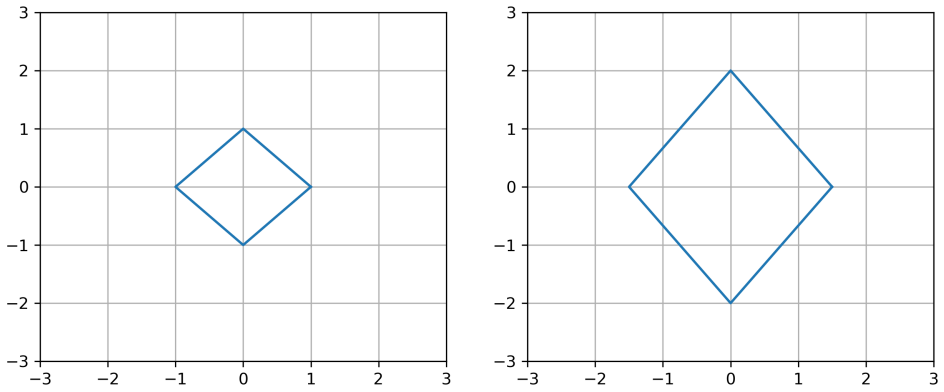
17. Suppose $\boldsymbol{x} = \begin{bmatrix}1 & 0 & 2 & c & -1\end{bmatrix}^T \in \mathbb{R}^5$ such that $\|\boldsymbol{x}\|_\infty = 5$. Find all possible values c.

18. Suppose A is a 2 by 2 matrix such that the image of the unit square under the linear transformation given by A is:



Determine $\text{cond}_\infty(A)$ (the condition number with respect to the ∞-norm).

19. Suppose A is a 2 by 2 matrix such that the image of the unit "diamond" under the linear transformation given by A is:



Determine $\text{cond}_1(A)$ (the condition number with respect to the 1-norm).

20. Suppose we have 4 points $(0, y_0), (1, y_1), (2, y_2), (3, y_3)$ and we want to interpolate the data using a spline $p(t)$ constructed from 3 degree 2 polynomials p_1, p_2, p_3 where

$$p_k(t) = a_k(t - t_{k-1})^2 + b_k(t - t_{k-1}) + c_k \quad , \quad t \in [t_{k-1}, t_k]$$

We require that $p(t)$ and $p'(t)$ are continuous and $p''(t_0) = 0$. Setup a linear system $A\boldsymbol{x} = \boldsymbol{b}$ where the solution is

$$\boldsymbol{x} = \begin{bmatrix} a_1 & b_1 & a_2 & b_2 & a_3 & b_3 \end{bmatrix}^T$$

Note: the system depends on the unspecified values y_0, y_1, y_2, y_3.

21. Suppose a cubic spline $p(t)$ interpolates the data

$$(0.0, 0.0), (1.0, 2.0), (2.0, 1.0), (3.0, 3.0), (4.0, -1.0), (5.0, 1.0)$$

and $p(t)$ has coefficient matrix

$$\begin{bmatrix} 1.9 & 1.9 & a_3 & 3.1 & 3.1 \\ -7.2 & -1.5 & b_3 & -6.3 & 3.0 \\ 7.3 & -1.4 & c_3 & -0.8 & -4.1 \\ 0.0 & 2.0 & 1.0 & 3.0 & -1.0 \end{bmatrix}$$

Determine the coefficients a_3, b_3, c_3.

22. Suppose a cubic spline $p(t)$ interpolates the data

$$(0.0, 1.0), (1.0, 3.0), (2.0, 1.0), (3.0, 1.0), (4.0, 2.0), (5.0, 1.0)$$

and $p(t)$ has coefficient matrix (rounded to 2 decimal places)

$$\begin{bmatrix} -1.19 & 1.93 & a_3 & -0.75 & 0.55 \\ 0.00 & -3.56 & b_3 & 0.60 & -1.65 \\ 3.19 & -0.37 & c_3 & 1.15 & 0.10 \\ 1.00 & 3.00 & 1.00 & 1.00 & 2.00 \end{bmatrix}$$

Determine the coefficients a_3, b_3, c_3.

23. Suppose we discretize the domain $[0, 1]$ of a differential equation with boundary conditions

$$y'' + p(t)y' + q(t)y = r(t) \quad , \quad y(0) = \alpha \quad , \quad y(1) = \beta$$

with step size $h = 0.1$ and derive a linear system $A\boldsymbol{y} = \boldsymbol{b}$. How many unknown values y_k are we solving for in this case?

24. Setup a linear system $A\boldsymbol{y} = \boldsymbol{b}$ to approximate the solution of the equation with boundary conditions

$$y'' = 2^t \quad , \quad y'(0) = 1 \ , \quad y(1) = 0$$

using step size $h = 0.25$. Use the forward difference formula and the boundary condition $y'(0) = 0$ to approximate the boundary value y_0.

25. Derive the general form of the linear system $A\boldsymbol{y} = \boldsymbol{b}$ for an equation with boundary conditions

$$y'' + p(t)y' = r(t) \quad , \quad y(t_0) = \alpha \quad , \quad y(t_f) = \beta$$

using the forward difference formula to approximate y'. Use the notation as in the examples: choose N, let $h = (t_f - t_0)/(N + 1)$ and $t_k = t_0 + kh$, let y_k denote an approximation of $y(t_k)$ and note $y_0 = \alpha$ and $y_{N+1} = \beta$.

26. Explain why it is not possible to derive a linear system $A\boldsymbol{y} = \boldsymbol{b}$ for the equation

$$y' = \cos(y) \quad , \quad y(0) = 0 \quad , \quad y(1) = \frac{\pi}{4}$$

by applying finite difference formulas.

27. Suppose we compute the finite difference approximation of the equation

$$y'' = \frac{5}{1 + t^4} \quad , \quad y(0) = 0 \quad , \quad y(1) = 1$$

with 5 equally spaced points from $t_0 = 0$ to $t_4 = 1$ and find $y_1 = -0.19554177$ and $y_3 = 0.35872678$. Determine y_2.

28. Setup a linear system $A\boldsymbol{y} = \boldsymbol{b}$ for the finite difference approximation of

$$y'' + ty = 0 \quad , \quad y(1) = 1 \quad , \quad y(3) = -1$$

using 5 equally spaced points from $t_0 = 1$ to $t_4 = 3$.

29. Setup a linear system $A\boldsymbol{y} = \boldsymbol{b}$ for the finite difference approximation of

$$y'' + ty = 0 \quad , \quad y(1) = 1 \quad , \quad y'(3) = -1$$

using 5 equally spaced points from $t_0 = 1$ to $t_4 = 3$. (Hint: use the backwards difference formula to approximate y_4.)

30. Setup the linear system $A\boldsymbol{y} = \boldsymbol{b}$ corresponding to the finite difference method applied to the equation

$$y'' + y' = t^2 \quad , \quad y(-1) = y(1) = 0$$

using 9 equally spaced points on the domain $[-1, 1]$.

31. Setup the linear system $A\boldsymbol{y} = \boldsymbol{b}$ corresponding to the finite difference method applied to the equation

$$y'' + \cos(t)y = \sin(t) \quad , \quad y(0) = y(2\pi) = 0$$

using 9 equally spaced points on the domain $[0, 2\pi]$.

32. Setup the linear system $A\boldsymbol{y} = \boldsymbol{b}$ corresponding to the finite difference method applied to the equation

$$y'' + \sin(t)y = \cos^2(t) \quad , \quad y(0) = y(2\pi) = 0$$

using 9 equally spaced points on the domain $[0, 2\pi]$.

Chapter 2

Least Squares Approximation

2.1 Review: Orthogonality

Big Idea. Vectors $\boldsymbol{x}, \boldsymbol{y} \in \mathbb{R}^n$ are orthogonal if $\boldsymbol{x} \cdot \boldsymbol{y} = 0$.

Definition. The **dot product** (or **inner product**) [KN, p.282] of vectors $\boldsymbol{x}, \boldsymbol{y} \in \mathbb{R}^n$ is

$$\boldsymbol{x} \cdot \boldsymbol{y} = \sum_{k=1}^{n} x_k y_k = x_1 y_1 + \cdots + x_n y_n$$

Note.

- Write the dot product of column vectors as matrix multiplication

$$\boldsymbol{x} \cdot \boldsymbol{y} = \boldsymbol{x}^T \boldsymbol{y} = \begin{bmatrix} x_1 & \cdots & x_n \end{bmatrix} \begin{bmatrix} y_1 \\ \vdots \\ y_n \end{bmatrix}$$

- We can also write the dot product in terms of the angle between vectors

$$\boldsymbol{x} \cdot \boldsymbol{y} = \|\boldsymbol{x}\| \|\boldsymbol{y}\| \cos\theta \qquad 0 \leq \theta \leq \pi$$

- The square root of the dot product of a vector $\boldsymbol{x}$ with itself is equal to the norm

$$\sqrt{\boldsymbol{x} \cdot \boldsymbol{x}} = \|\boldsymbol{x}\|$$

Theorem. Let $\boldsymbol{x}, \boldsymbol{y} \in \mathbb{R}^n$. Then

$$|\boldsymbol{x} \cdot \boldsymbol{y}| \leq \|\boldsymbol{x}\| \|\boldsymbol{y}\|$$

This is called the **Cauchy-Schwartz inequality** [KN, p.284].

Theorem. Let $\boldsymbol{x}, \boldsymbol{y} \in \mathbb{R}^n$. Then

$$\|\boldsymbol{x} + \boldsymbol{y}\| \leq \|\boldsymbol{x}\| + \|\boldsymbol{y}\|$$

This is called the **triangle inequality** [KN, p.284].

Definition. Vectors $\boldsymbol{x}, \boldsymbol{y} \in \mathbb{R}^n$ are **orthogonal** [KN, p.285] if $\boldsymbol{x} \cdot \boldsymbol{y} = 0$. More generally, (nonzero) vectors $\boldsymbol{x}_1, \ldots, \boldsymbol{x}_m \in \mathbb{R}^n$ are **orthogonal** if $\boldsymbol{x}_i \cdot \boldsymbol{x}_j = 0$ for all $i \neq j$. In other words, each $\boldsymbol{x}_i$ is orthogonal to every other vector $\boldsymbol{x}_j$ in the set. Furthermore, vectors $\boldsymbol{x}_1, \ldots, \boldsymbol{x}_m \in \mathbb{R}^n$

are **orthonormal** if they are orthogonal and each is a unit vector, $\|\boldsymbol{x}_k\| = 1$, $k = 1, \ldots, m$.

Theorem. Let $\boldsymbol{x}_1, \ldots, \boldsymbol{x}_m \in \mathbb{R}^n$ be orthogonal. Then

$$\|\boldsymbol{x}_1 + \cdots + \boldsymbol{x}_m\| = \|\boldsymbol{x}_1\| + \cdots + \|\boldsymbol{x}_m\|$$

This is called the **Pythagoras theorem** [KN, p.286].

Definition. Let $S_1 \subset \mathbb{R}^n$ and $S_2 \subset \mathbb{R}^n$ be subspaces. Then S_1 and S_2 are **orthogonal** if $\boldsymbol{x}_1 \cdot \boldsymbol{x}_2 = 0$ for all $\boldsymbol{x}_1 \in S_1$ and $\boldsymbol{x}_2 \in S_2$. Notation: if S_1 and S_2 are orthogonal subspaces, we write $S_1 \perp S_2$.

Example. Let $S_1 \subset \mathbb{R}^3$ and $S_2 \subset \mathbb{R}^3$ be 2-dimensional subspaces (planes). Is it possible that $S_1 \perp S_2$? No!

Definition. Let $S \subset \mathbb{R}^n$ be a subspace. The **orthogonal complement of** S [KN, p.418] is the subspace

$$S^\perp = \{\boldsymbol{x} \in \mathbb{R}^n : \boldsymbol{x} \cdot \boldsymbol{y} = 0 \text{ for all } \boldsymbol{y} \in \mathbb{R}^n\}$$

Theorem. Let $S \subset \mathbb{R}^n$ be a subspace. Then

$$\dim(S) + \dim(S^\perp) = n$$

2.2 Orthogonal Projection

Big Idea. The point in a subspace $U \subset \mathbb{R}^n$ nearest to $\boldsymbol{x} \in \mathbb{R}^n$ is the orthogonal projection $\text{proj}_U(\boldsymbol{x})$ of $\boldsymbol{x}$ onto U.

Definition. The **projection** [KN, p.419] of a vector $\boldsymbol{x}$ onto a vector $\boldsymbol{u}$ is

$$\text{proj}_{\boldsymbol{u}}(\boldsymbol{x}) = \frac{\boldsymbol{x} \cdot \boldsymbol{u}}{\|\boldsymbol{u}\|^2}\boldsymbol{u}$$

Note. Projection onto $\boldsymbol{u}$ is given by matrix multiplication

$$\text{proj}_{\boldsymbol{u}}(\boldsymbol{x}) = P\boldsymbol{x} \quad \text{where} \quad P = \frac{1}{\|\boldsymbol{u}\|^2}\boldsymbol{u}\boldsymbol{u}^T$$

Note that $P^2 = P$, $P^T = P$ and $\text{rank}(P) = 1$.

Theorem. Let $\{\boldsymbol{u}_1, \ldots, \boldsymbol{u}_m\}$ be a basis of a subspace $U \subset \mathbb{R}^n$. The **Gram-Schmidt orthogonalization algorithm** [KN, p.417] constructs an orthogonal basis of U:

$$\begin{aligned}
\boldsymbol{v}_1 &= \boldsymbol{u}_1 \\
\boldsymbol{v}_2 &= \boldsymbol{u}_2 - \text{proj}_{\boldsymbol{v}_1}(\boldsymbol{u}_2) \\
\boldsymbol{v}_3 &= \boldsymbol{u}_3 - \text{proj}_{\boldsymbol{v}_1}(\boldsymbol{u}_3) - \text{proj}_{\boldsymbol{v}_2}(\boldsymbol{u}_3) \\
&\vdots \\
\boldsymbol{v}_m &= \boldsymbol{u}_m - \text{proj}_{\boldsymbol{v}_1}(\boldsymbol{u}_m) - \text{proj}_{\boldsymbol{v}_2}(\boldsymbol{u}_m) - \cdots - \text{proj}_{\boldsymbol{v}_{m-1}}(\boldsymbol{u}_m)
\end{aligned}$$

Then $\{\boldsymbol{v}_1, \ldots, \boldsymbol{v}_m\}$ is an orthogonal basis of U. Furthermore, let

$$\boldsymbol{w}_k = \frac{\boldsymbol{v}_k}{\|\boldsymbol{v}_k\|} \quad , \; k = 1, \ldots, m$$

Then $\{\boldsymbol{w}_1, \ldots, \boldsymbol{w}_m\}$ is an orthonormal basis of U.

Example. Construct an orthonormal basis of the subspace U spanned by

$$\boldsymbol{u}_1 = \begin{bmatrix} 1 \\ 0 \\ 1 \\ 0 \end{bmatrix} \qquad \boldsymbol{u}_2 = \begin{bmatrix} 1 \\ 1 \\ 1 \\ 0 \end{bmatrix} \qquad \boldsymbol{u}_3 = \begin{bmatrix} 1 \\ 1 \\ 0 \\ 0 \end{bmatrix}$$

Compute

$$\begin{aligned}
\boldsymbol{v}_1 &= \boldsymbol{u}_1 \\
\boldsymbol{v}_2 &= \boldsymbol{u}_2 - \text{proj}_{\boldsymbol{v}_1}(\boldsymbol{u}_2) \\
\boldsymbol{v}_3 &= \boldsymbol{u}_3 - \text{proj}_{\boldsymbol{v}_1}(\boldsymbol{u}_3) - \text{proj}_{\boldsymbol{v}_2}(\boldsymbol{u}_3)
\end{aligned}$$

and we find an orthogonal basis

$$\boldsymbol{v}_1 = \begin{bmatrix} 1 \\ 0 \\ 1 \\ 0 \end{bmatrix} \qquad \boldsymbol{v}_2 = \begin{bmatrix} 0 \\ 1 \\ 0 \\ 0 \end{bmatrix} \qquad \boldsymbol{v}_3 = \frac{1}{2}\begin{bmatrix} 1 \\ 0 \\ -1 \\ 0 \end{bmatrix}$$

and an orthonormal basis

$$\boldsymbol{w}_1 = \frac{1}{\sqrt{2}}\begin{bmatrix}1\\0\\1\\0\end{bmatrix} \qquad \boldsymbol{w}_2 = \begin{bmatrix}0\\1\\0\\0\end{bmatrix} \qquad \boldsymbol{w}_3 = \frac{1}{\sqrt{2}}\begin{bmatrix}1\\0\\-1\\0\end{bmatrix}$$

Definition. Let $U \subset \mathbb{R}^n$ be a subspace and let $\{\boldsymbol{u}_1, \ldots, \boldsymbol{u}_m\}$ be an orthogonal basis of U. The **orthogonal projection** [KN, p.420] of a vector $\boldsymbol{x}$ onto U is

$$\text{proj}_U(\boldsymbol{x}) = \frac{\boldsymbol{x}\cdot\boldsymbol{u}_1}{\|\boldsymbol{u}_1\|^2}\boldsymbol{u}_1 + \cdots + \frac{\boldsymbol{x}\cdot\boldsymbol{u}_m}{\|\boldsymbol{u}_m\|^2}\boldsymbol{u}_m$$

Note. Projection onto U is given by matrix multiplication

$$\text{proj}_{\boldsymbol{U}}(\boldsymbol{x}) = P\boldsymbol{x} \quad \text{where} \quad P = \frac{1}{\|\boldsymbol{u}_1\|^2}\boldsymbol{u}_1\boldsymbol{u}_1^T + \cdots + \frac{1}{\|\boldsymbol{u}_m\|^2}\boldsymbol{u}_m\boldsymbol{u}_m^T$$

Note that $P^2 = P$, $P^T = P$ and $\text{rank}(P) = m$.

Definition. A matrix P is an **orthogonal projector** (or **orthogonal projection matrix**) if $P^2 = P$ and $P^T = P$ [MH, p.110].

Proposition. Let P be the orthogonal projection onto U. Then $I - P$ is the orthogonal projection matrix onto $U^\perp$.

Example. Find the orthogonal projection matrix P which projects onto the subspace spanned by the vectors

$$\boldsymbol{u}_1 = \begin{bmatrix}1\\0\\-1\end{bmatrix} \qquad \boldsymbol{u}_2 = \begin{bmatrix}1\\1\\1\end{bmatrix}$$

Compute $\boldsymbol{u}_1 \cdot \boldsymbol{u}_2 = 0$ therefore the vectors are orthogonal. Compute

$$P = \frac{1}{\|\boldsymbol{u}_1\|^2}\boldsymbol{u}_1\boldsymbol{u}_1^T + \frac{1}{\|\boldsymbol{u}_2\|^2}\boldsymbol{u}_2\boldsymbol{u}_2^T = \frac{1}{2}\begin{bmatrix}1\\0\\-1\end{bmatrix}\begin{bmatrix}1 & 0 & -1\end{bmatrix} + \frac{1}{3}\begin{bmatrix}1\\1\\1\end{bmatrix}\begin{bmatrix}1 & 1 & 1\end{bmatrix}$$

$$= \frac{1}{2}\begin{bmatrix}1 & 0 & -1\\0 & 0 & 0\\-1 & 0 & 1\end{bmatrix} + \frac{1}{3}\begin{bmatrix}1 & 1 & 1\\1 & 1 & 1\\1 & 1 & 1\end{bmatrix} = \frac{1}{6}\begin{bmatrix}5 & 2 & 1\\2 & 2 & 2\\-1 & 2 & 5\end{bmatrix}$$

Example. Find the orthogonal projection matrix $P_\perp$ which projects onto $U^\perp$ where U the subspace spanned by the vectors

$$\boldsymbol{u}_1 = \begin{bmatrix} 1 \\ 0 \\ -1 \end{bmatrix} \qquad \boldsymbol{u}_2 = \begin{bmatrix} 1 \\ 1 \\ 1 \end{bmatrix}$$

as in the previous example. Compute

$$P_\perp = I - P = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} - \frac{1}{6}\begin{bmatrix} 5 & 2 & 1 \\ 2 & 2 & 2 \\ -1 & 2 & 5 \end{bmatrix} = \frac{1}{6}\begin{bmatrix} 1 & -2 & -1 \\ -2 & 4 & -2 \\ 1 & -2 & 1 \end{bmatrix}$$

Note that

$$\boldsymbol{u}_3 = \begin{bmatrix} 1 \\ -2 \\ 1 \end{bmatrix}$$

is orthogonal to both $\boldsymbol{u}_1$ and $\boldsymbol{u}_2$ and is a basis of the orthogonal complement $U^\perp$. Therefore we could also compute

$$P_\perp = \frac{1}{\|\boldsymbol{u}_3\|^2}\boldsymbol{u}_3\boldsymbol{u}_3^T = \frac{1}{6}\begin{bmatrix} 1 \\ -2 \\ 1 \end{bmatrix}\begin{bmatrix} 1 & -2 & 1 \end{bmatrix} = \frac{1}{6}\begin{bmatrix} 1 & -2 & -1 \\ -2 & 4 & -2 \\ 1 & -2 & 1 \end{bmatrix}$$

Theorem. Let $U \subset \mathbb{R}^n$ be a subspace and let $\boldsymbol{x} \in \mathbb{R}^n$. Then

$$\boldsymbol{x} - \text{proj}_U(\boldsymbol{x}) \in U^\perp$$

and $\text{proj}_U(\boldsymbol{x})$ is the closest vector in U to $\boldsymbol{x}$ in the sense that

$$\|\boldsymbol{x} - \text{proj}_U(\boldsymbol{x})\| < \|\boldsymbol{x} - \boldsymbol{y}\| \qquad \text{for all } \boldsymbol{y} \in U \ , \ \boldsymbol{y} \neq \text{proj}_U(\boldsymbol{x})$$

See the **Projection Theorem** [KN, p.420].

2.3 QR Decomposition by Gram-Schmidt Orthogonalization

Big Idea. The QR decomposition is given by $A = QR$ where Q is an orthogonal matrix and R is an upper triangular matrix. There are several ways to compute the QR decomposition including Gram-Schmidt orthogonalization and elementary reflectors.

Definition. A matrix A is **orthogonal** [KN, p.424] if $A^TA = AA^T = I$.

Proposition. If A is an orthogonal matrix, then:

- $\|A\boldsymbol{x}\| = \|\boldsymbol{x}\|$ for all $\boldsymbol{x} \in \mathbb{R}^n$ since

$$\|A\boldsymbol{x}\|^2 = (A\boldsymbol{x})^T A\boldsymbol{x} = \boldsymbol{x}^T A^T A\boldsymbol{x} = \boldsymbol{x}^T\boldsymbol{x} = \|\boldsymbol{x}\|^2$$

- the columns of A are orthonormal.
- the rows of A are orthonormal.

See [KN, p.424].

Example. Rotations and reflections are examples of orthogonal matrices.

Note. An orthogonal matrix and an orthogonal projector are **not** the same thing but they are related. If P is an orthogonal projector then $Q = I - 2P$ is an orthogonal (and symmetric) matrix. In fact, if P projects onto a subspace U then Q is the reflection through $U^\perp$.

Definition. Let A be an $n \times m$ matrix with $\text{rank}(A) = m$ and let $\boldsymbol{a}_1, \ldots, \boldsymbol{a}_m$ be the columns of A. Apply the Gram-Schmidt algorithm to the columns and construct an orthonormal basis $\{\boldsymbol{w}_1, \ldots, \boldsymbol{w}_m\}$ of the column space. Project the columns onto the basis

$$\begin{aligned}
\boldsymbol{a}_1 &= (\boldsymbol{w}_1 \cdot \boldsymbol{a}_1)\boldsymbol{w}_1 \\
\boldsymbol{a}_2 &= (\boldsymbol{w}_1 \cdot \boldsymbol{a}_2)\boldsymbol{w}_1 + (\boldsymbol{w}_2 \cdot \boldsymbol{a}_2)\boldsymbol{w}_2 \\
&\vdots \\
\boldsymbol{a}_m &= (\boldsymbol{w}_1 \cdot \boldsymbol{a}_m)\boldsymbol{w}_1 + (\boldsymbol{w}_2 \cdot \boldsymbol{a}_m)\boldsymbol{w}_2 + \cdots + (\boldsymbol{w}_m \cdot \boldsymbol{a}_m)\boldsymbol{w}_m
\end{aligned}$$

where $\boldsymbol{a}_k \in \text{span}\{\boldsymbol{w}_1, \ldots, \boldsymbol{w}_k\}$ by construction. Write as matrix multiplication

$$A = Q_1 R_1$$

where

$$Q_1 = \begin{bmatrix} \boldsymbol{w}_1 & \cdots & \boldsymbol{w}_m \end{bmatrix} \qquad R_1 = \begin{bmatrix} \boldsymbol{w}_1 \cdot \boldsymbol{a}_1 & \boldsymbol{w}_1 \cdot \boldsymbol{a}_2 & \cdots & \boldsymbol{w}_1 \cdot \boldsymbol{a}_m \\ & \boldsymbol{w}_2 \cdot \boldsymbol{a}_2 & \cdots & \boldsymbol{w}_2 \cdot \boldsymbol{a}_m \\ & & \ddots & \vdots \\ & & & \boldsymbol{w}_m \cdot \boldsymbol{a}_m \end{bmatrix}$$

Extend the basis to an orthonormal basis $\{\boldsymbol{w}_1, \ldots, \boldsymbol{w}_m, \boldsymbol{w}_{m+1}, \ldots, \boldsymbol{w}_n\}$ of $\mathbb{R}^n$ where

$\{\boldsymbol{w}_{m+1}, \ldots, \boldsymbol{w}_n\}$ is *any* orthonormal basis of the orthogonal complement $\text{col}(A)^\perp$ and let

$$Q_2 = \begin{bmatrix} & & \\ \boldsymbol{w}_{m+1} & \cdots & \boldsymbol{w}_n \\ & & \end{bmatrix}$$

Finally, the **QR decomposition** of A is

$$A = QR = \begin{bmatrix} Q_1 & Q_2 \end{bmatrix} \begin{bmatrix} R_1 \\ 0 \end{bmatrix}$$

where Q is a $n \times n$ orthogonal matrix and R is a $n \times m$ upper triangular matrix. See Wikipedia: QR decomposition and also [KN, p.437].

Example. Compute the QR decomposition for the matrix

$$A = \begin{bmatrix} 1 & 1 & 1 \\ 0 & 1 & 1 \\ 1 & 1 & 0 \\ 0 & 0 & 0 \end{bmatrix}$$

In a previous example, we found an orthonormal basis of the column space

$$\boldsymbol{w}_1 = \frac{1}{\sqrt{2}} \begin{bmatrix} 1 \\ 0 \\ 1 \\ 0 \end{bmatrix} \qquad \boldsymbol{w}_2 = \begin{bmatrix} 0 \\ 1 \\ 0 \\ 0 \end{bmatrix} \qquad \boldsymbol{w}_3 = \frac{1}{\sqrt{2}} \begin{bmatrix} 1 \\ 0 \\ -1 \\ 0 \end{bmatrix}$$

Extend to an orthonormal basis of $\mathbb{R}^4$ by $\boldsymbol{w}_4 = \begin{bmatrix} 0 & 0 & 0 & 1 \end{bmatrix}^T$. Therefore we have

$$Q = \begin{bmatrix} 1/\sqrt{2} & 0 & 1/\sqrt{2} & 0 \\ 0 & 1 & 0 & 0 \\ 1/\sqrt{2} & 0 & -1/\sqrt{2} & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

and

$$R = \begin{bmatrix} \boldsymbol{w}_1 \cdot \boldsymbol{a}_1 & \boldsymbol{w}_1 \cdot \boldsymbol{a}_2 & \boldsymbol{w}_1 \cdot \boldsymbol{a}_3 \\ 0 & \boldsymbol{w}_2 \cdot \boldsymbol{a}_2 & \boldsymbol{w}_2 \cdot \boldsymbol{a}_3 \\ 0 & 0 & \boldsymbol{w}_3 \cdot \boldsymbol{a}_3 \\ 0 & 0 & 0 \end{bmatrix} = \begin{bmatrix} \sqrt{2} & \sqrt{2} & 1/\sqrt{2} \\ 0 & 1 & 1 \\ 0 & 0 & 1/\sqrt{2} \\ 0 & 0 & 0 \end{bmatrix}$$

Note. The Gram-Schmidt algorithm shows that the QR decomposition exists but it is not the most efficient way to compute the QR decomposition. Software such as MATLAB `qr` (see documentation) and SciPy `scipy.linalg.qr` (see documentation) which is built on LAPACK

(see documentation) use elementary reflectors to construct the matrices Q and R.

2.4 QR Decomposition by Elementary Reflectors

Big Idea. The QR decomposition is given by $A = QR$ where Q is an orthogonal matrix and R is an upper triangular matrix. There are several ways to compute the QR decomposition including Gram-Schmidt orthogonalization and elementary reflectors.

Note. The Gram-Schmidt algorithm shows that the QR decomposition exists but it is not the most efficient way to compute the QR decomposition. Software such as MATLAB `qr` (see documentation) and SciPy `scipy.linalg.qr` (see documentation) which is built on LAPACK (see documentation) use elementary reflectors to construct the matrices Q and R.

Definition. An **elementary reflector** (or **Householder transformation**) [MH, p.120] is matrix of the form

$$H = I - \frac{2}{\|\boldsymbol{u}\|^2}\boldsymbol{u}\boldsymbol{u}^T$$

for some nonzero vector $\boldsymbol{u} \in \mathbb{R}^n$. Note that a reflector is an orthogonal matrix since $H^T = H$ and $H^2 = I$. Note also that if P is the orthogonal projection onto $\boldsymbol{u}$ then $H = I - 2P$ and H is the reflection through the hyperplane orthogonal to $\boldsymbol{u}$.

Definition. Let $\{\boldsymbol{e}_1, \ldots, \boldsymbol{e}_n\}$ be the standard orthonormal basis of $\mathbb{R}^n$

$$\boldsymbol{e}_1 = \begin{bmatrix} 1 \\ 0 \\ \vdots \\ 0 \end{bmatrix} \quad \boldsymbol{e}_2 = \begin{bmatrix} 0 \\ 1 \\ \vdots \\ 0 \end{bmatrix} \quad \cdots \quad \boldsymbol{e}_n = \begin{bmatrix} 0 \\ 0 \\ \vdots \\ 1 \end{bmatrix}$$

Proposition. Let $\boldsymbol{a} \in \mathbb{R}^n$ and $\alpha = -\text{sign}(a_1)\|\boldsymbol{a}\|$. Let $\boldsymbol{u} = \boldsymbol{a} - \alpha\boldsymbol{e}_1$, let P be the orthogonal projector onto $\boldsymbol{u}$ and let $H = I - 2P$ be the corresponding elementary reflector. Then

$$H\boldsymbol{a} = \alpha\boldsymbol{e}_1 = \begin{bmatrix} \alpha \\ 0 \\ \vdots \\ 0 \end{bmatrix}$$

More generally, let $\boldsymbol{a} \in \mathbb{R}^n$ and partition the vector

$$\boldsymbol{a} = \begin{bmatrix} \boldsymbol{a}_1 \\ \boldsymbol{a}_2 \end{bmatrix} \quad \text{where} \quad \boldsymbol{a}_1 = \begin{bmatrix} a_1 \\ \vdots \\ a_{k-1} \end{bmatrix} \in \mathbb{R}^{k-1} \quad \text{and} \quad \boldsymbol{a}_2 = \begin{bmatrix} a_k \\ \vdots \\ a_n \end{bmatrix} \in \mathbb{R}^{n-k+1}$$

Let $\alpha = -\text{sign}(a_k)\|\boldsymbol{a}_2\|$ and let

$$\boldsymbol{u} = \begin{bmatrix} \boldsymbol{0} \\ \boldsymbol{a}_2 \end{bmatrix} - \alpha \boldsymbol{e}_k = \begin{bmatrix} 0 \\ \vdots \\ 0 \\ a_k - \alpha \\ a_{k+1} \\ \vdots \\ a_n \end{bmatrix}$$

and let H be the corresponding elementary reflector. Then

$$H\boldsymbol{a} = \begin{bmatrix} a_1 \\ \vdots \\ a_{k-1} \\ \alpha \\ 0 \\ \vdots \\ 0 \end{bmatrix}$$

Proof. See [MH, p.120]. □

Theorem. Let A be an $n \times m$ matrix with $n > m$. There exists a sequence of elementary reflectors $H_1, \ldots, H_m$ such that $H_m \cdots H_1 A = R$ is upper triangular and therefore

$$A = QR$$

where $Q = H_1 \cdots H_m$.

Proof. For each column, construct an elementary reflector to annihilate the entries below the diagonal. For example, if A has 3 columns and 4 rows then

$$A = \begin{bmatrix} * & * & * \\ * & * & * \\ * & * & * \\ * & * & * \end{bmatrix} \quad H_1 A = \begin{bmatrix} * & * & * \\ 0 & * & * \\ 0 & * & * \\ 0 & * & * \end{bmatrix} \quad H_2 H_1 A = \begin{bmatrix} * & * & * \\ 0 & * & * \\ 0 & 0 & * \\ 0 & 0 & * \end{bmatrix} \quad H_3 H_2 H_1 A = \begin{bmatrix} * & * & * \\ 0 & * & * \\ 0 & 0 & * \\ 0 & 0 & 0 \end{bmatrix}$$

Since each H is an elementary reflector, we have $A = H_1^{-1} H_2^{-1} H_3^{-1} R = H_1 H_2 H_3 R$. □

Example. Find the QR decomposition of $A = \begin{bmatrix} 1 & 1 & 1 \\ 0 & 1 & 1 \\ 1 & 1 & 0 \end{bmatrix}$ by elementary reflectors.

Construct the vector

$$\boldsymbol{u}_1 = \boldsymbol{a}_1 - \alpha \boldsymbol{e}_1 = \begin{bmatrix} 1 \\ 0 \\ 1 \end{bmatrix} + \sqrt{2} \begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix} = \begin{bmatrix} 1+\sqrt{2} \\ 0 \\ 1 \end{bmatrix}$$

Compute the norm squared

$$\|\boldsymbol{u}_1\|^2 = (1+\sqrt{2})^2 + 1 = 4 + 2\sqrt{2}$$

and construct the elementary reflector

$$\begin{aligned} H_1 &= I - \frac{2}{\|\boldsymbol{u}_1\|^2} \boldsymbol{u}_1 \boldsymbol{u}_1^T = I - \frac{2}{4+2\sqrt{2}} \begin{bmatrix} 1+\sqrt{2} \\ 0 \\ 1 \end{bmatrix} \begin{bmatrix} 1+\sqrt{2} & 0 & 1 \end{bmatrix} \\ &= I - \frac{1}{2+\sqrt{2}} \begin{bmatrix} 3+2\sqrt{2} & 0 & 1+\sqrt{2} \\ 0 & 0 & 0 \\ 1+\sqrt{2} & 0 & 1 \end{bmatrix} = \frac{1}{2+\sqrt{2}} \begin{bmatrix} -1-\sqrt{2} & 0 & -1-\sqrt{2} \\ 0 & 2+\sqrt{2} & 0 \\ -1-\sqrt{2} & 0 & 1+\sqrt{2} \end{bmatrix} \\ &= \frac{1}{\sqrt{2}} \begin{bmatrix} -1 & 0 & -1 \\ 0 & \sqrt{2} & 0 \\ -1 & 0 & 1 \end{bmatrix} \end{aligned}$$

Compute

$$H_1 A = \frac{1}{\sqrt{2}} \begin{bmatrix} -1 & 0 & -1 \\ 0 & \sqrt{2} & 0 \\ -1 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 1 & 1 \\ 0 & 1 & 1 \\ 1 & 1 & 0 \end{bmatrix} = \frac{1}{\sqrt{2}} \begin{bmatrix} -2 & -2 & -1 \\ 0 & \sqrt{2} & \sqrt{2} \\ 0 & 0 & -1 \end{bmatrix}$$

Since the result is already upper triangular we have $A = QR$ where

$$Q = \frac{1}{\sqrt{2}} \begin{bmatrix} -1 & 0 & -1 \\ 0 & \sqrt{2} & 0 \\ -1 & 0 & 1 \end{bmatrix} \qquad R = \frac{1}{\sqrt{2}} \begin{bmatrix} -2 & -2 & -1 \\ 0 & \sqrt{2} & \sqrt{2} \\ 0 & 0 & -1 \end{bmatrix}$$

2.5 Least Squares Approximation

Big Idea. Find the best approximation of the system $A\boldsymbol{x} \cong \boldsymbol{b}$ by minimizing the distance $\|A\boldsymbol{x} - \boldsymbol{b}\|$. There are several methods to find the approximation including the normal equations and the QR decomposition.

Definition. Let A be an $m \times n$ matrix with $m > n$ and $\text{rank}(A) = n$. The best approximation of the system $A\boldsymbol{x} \cong \boldsymbol{b}$ is the vector $\boldsymbol{x}$ which minimizes the distance $\|A\boldsymbol{x} - \boldsymbol{b}\|$. Since $\|\cdot\|$ is the 2-norm, the best approximation is called the **least squares approximation**.

Proposition. Let A be an $m \times n$ matrix with $m > n$ and $\text{rank}(A) = n$. The least squares approximation of the system $A\boldsymbol{x} \cong \boldsymbol{b}$ is the solution of the **normal equations** [KN, p.311]

$$A^T A\boldsymbol{x} = A^T\boldsymbol{b}$$

Proof. If $\boldsymbol{x} \in \mathbb{R}^n$, then $A\boldsymbol{x} \in \text{col}(A)$. The projection theorem [KN, p.420] states that the point in $\text{col}(A)$ nearest to $\boldsymbol{b} \in \mathbb{R}^m$ is the orthogonal projection of $\boldsymbol{b}$ onto $\text{col}(A)$. If $\boldsymbol{x}$ is the vector such that $A\boldsymbol{x} = \text{proj}_{\text{col}(A)}(\boldsymbol{b})$, then $A\boldsymbol{x} - \boldsymbol{b}$ is in $\text{col}(A)^\perp$ and therefore

$$A^T(A\boldsymbol{x} - \boldsymbol{b}) = 0 \quad \Rightarrow \quad A^T A\boldsymbol{x} = A^T\boldsymbol{b}$$

We assume $\text{rank}(A) = n$, therefore $A^T A$ is nonsingular and the solution exists and is unique. □

Proposition. Let A be an $m \times n$ matrix with $m > n$ and $\text{rank}(A) = n$. The least squares approximation of the system $A\boldsymbol{x} \cong \boldsymbol{b}$ is the solution of the system of equations

$$R_1\boldsymbol{x} = \boldsymbol{c}_1 \quad \text{where } A = QR = [Q_1 \;\; Q_2]\begin{bmatrix} R_1 \\ 0 \end{bmatrix} \text{ and } Q^T\boldsymbol{b} = \begin{bmatrix} \boldsymbol{c}_1 \\ \boldsymbol{c}_2 \end{bmatrix}$$

Proof. The matrix Q is orthogonal therefore

$$\|A\boldsymbol{x} - \boldsymbol{b}\|^2 = \|Q(R\boldsymbol{x} - Q^T\boldsymbol{b})\|^2 = \|R\boldsymbol{x} - Q^T\boldsymbol{b}\|^2 = \left\| \begin{bmatrix} R_1\boldsymbol{x} \\ 0 \end{bmatrix} - \begin{bmatrix} \boldsymbol{c}_1 \\ \boldsymbol{c}_2 \end{bmatrix} \right\|^2 = \|R_1\boldsymbol{x} - \boldsymbol{c}_1\|^2 + \|\boldsymbol{c}_2\|^2$$

where we use the Pythagoras theorem in the last equality. The vector $\boldsymbol{c}_2$ does not depend on $\boldsymbol{x}$ therefore the minimum value of $\|A\boldsymbol{x} - \boldsymbol{b}\|$ occurs when $R_1\boldsymbol{x} = \boldsymbol{c}_1$. □

2.6 Fitting Models to Data

Big Idea. Least squares data fitting computes coefficients $c_1, \ldots, c_n$ such that the model function $f(t, \boldsymbol{c}) = c_1 f_1(t) + \cdots + c_n f_n(t)$ best fits the data $(t_1, y_1), \ldots, (t_m, y_m)$.

Definition. Suppose we have m points

$$(t_1, y_1), \ldots, (t_m, y_m)$$

and we want to find a line

$$y = c_1 + c_2 t$$

that "best fits" the data. There are different ways to quantify what "best fit" means but

the most common method is called **least squares linear regression**. In least squares linear regression, we want to minimize the sum of squared errors

$$SSE = \sum_i (y_i - (c_1 + c_2 t_i))^2$$

In matrix notation, the sum of squared errors is

$$SSE = \|\boldsymbol{y} - A\boldsymbol{c}\|^2$$

where

$$\boldsymbol{y} = \begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_m \end{bmatrix} \quad A = \begin{bmatrix} 1 & t_1 \\ 1 & t_2 \\ \vdots & \vdots \\ 1 & t_m \end{bmatrix} \quad \boldsymbol{c} = \begin{bmatrix} c_1 \\ c_2 \end{bmatrix}$$

We assume that $m \geq 2$ and $t_i \neq t_j$ for all $i \neq j$ (which implies $\text{rank}(A) = 2$). Therefore the vector of coefficients

$$\boldsymbol{c} = \begin{bmatrix} c_1 \\ c_2 \end{bmatrix}$$

is the least squares approximation of the system $A\boldsymbol{c} \cong \boldsymbol{y}$.

Definition. More generally, given m data points

$$(t_1, y_1), \ldots, (t_m, y_m)$$

and a **model function** $f(t, \boldsymbol{c})$ which depends on parameters $c_1, \ldots, c_n$, the **least squares data fitting problem** consists of computing parameters $c_1, \ldots, c_n$ which minimize the sum of squared errors

$$SSE = \sum_i (y_i - f(t_i, \boldsymbol{c}))^2$$

If the model function is of the form

$$f(t, \boldsymbol{c}) = c_1 f_1(t) + \cdots + c_n f_1(t)$$

for some functions $f_1(t), \ldots, f_n(t)$ then we say the data fitting problem is **linear** [MH, p.106] (but note the function $f_1, \ldots, f_n$ are not necessarily linear). In the linear case, use matrix notation to write the sum of squared errors as

$$SSE = \|\boldsymbol{y} - A\boldsymbol{c}\|^2$$

where

$$\boldsymbol{y} = \begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_m \end{bmatrix} \quad A = \begin{bmatrix} f_1(t_1) & f_2(t_1) & \cdots & f_n(t_1) \\ f_1(t_2) & f_2(t_2) & \cdots & f_n(t_2) \\ \vdots & & & \vdots \\ f_1(t_m) & f_2(t_m) & \cdots & f_n(t_m) \end{bmatrix} \quad \boldsymbol{c} = \begin{bmatrix} c_1 \\ c_2 \\ \vdots \\ c_n \end{bmatrix}$$

We assume that $m \geq n$ and $f_1, \ldots, f_n$ are linearly independenty (which implies $\text{rank}(A) = n$). Therefore the vector of coefficients

$$\boldsymbol{c} = \begin{bmatrix} c_1 \\ c_2 \\ \vdots \\ c_n \end{bmatrix}$$

is the least squares approximation of the system $A\boldsymbol{c} \cong \boldsymbol{y}$.

2.7 Exercises

1. Determine whether the statement is **True** or **False**.

   (a) Let $\boldsymbol{u} \in \mathbb{R}^n$ be a nonzero vector and let $H = I - \dfrac{2}{\|\boldsymbol{u}\|^2}\boldsymbol{u}\boldsymbol{u}^T$ be the corresponding elementary reflector. Then $H\boldsymbol{v} = \boldsymbol{v}$ for all $\boldsymbol{v} \in \text{span}\{\boldsymbol{u}\}^\perp$.

   (b) Let $U, V \subset \mathbb{R}^n$ be subspaces such that U and V are orthogonal. If $\dim(U) = m$ then $\dim(U) = m$ then $\dim(V) = n - m$.

   (c) If A^TA is a diagonal matrix, then the columns of A are orthogonal.

   (d) If AA^T is a diagonal matrix, then the columns of A are orthogonal.

   (e) If A^TA is a diagonal matrix, then the rows of A are orthogonal.

   (f) If AA^T is a diagonal matrix, then the rows of A are orthogonal.

   (g) Let $U \subset \mathbb{R}^n$ be a subspace. If P_1 is the orthogonal projector onto U and P_2 is the orthogonal projector onto the orthogonal complement $U^\perp$, then $I = P_1 + P_2$.

   (h) Let $U \subset \mathbb{R}^n$ be a subspace. If P_1 is the orthogonal projector onto U and P_2 is the orthogonal projector onto the orthogonal complement $U^\perp$, then $P_1P_2 = P_2P_1 = 0$.

   (i) Let $\boldsymbol{u}_1, \boldsymbol{u}_2, \boldsymbol{u}_3 \in \mathbb{R}^3$ be nonzero vectors. If $\boldsymbol{u}_1$ is orthogonal to $\boldsymbol{u}_2$, and $\boldsymbol{u}_2$ is orthogonal to $\boldsymbol{u}_3$ then $\boldsymbol{u}_1$ is orthogonal to $\boldsymbol{u}_3$.

   (j) Let A be a $m \times n$ matrix with $m \geq n$ and let $\boldsymbol{b} \in \mathbb{R}^m$. There is a unique vector $\boldsymbol{x} \in \mathbb{R}^n$ which minimizes the norm of the residual $\|A\boldsymbol{x} - \boldsymbol{b}\|$.

   (k) Let $A = QR$ where Q is an orthogonal matrix and R is upper triangular. Then $\|A\|_p = \|R\|_p$ for any $p \geq 1$.

2. Find the elementary reflector H corresponding to the vector $\boldsymbol{u} = \begin{bmatrix} 2 & 1 & 0 & 1 \end{bmatrix}^T$.

3. Let $U \subset \mathbb{R}^3$ be the subspace spanned by

$$\boldsymbol{u}_1 = \begin{bmatrix} 1 \\ 1 \\ 1 \end{bmatrix} \quad \boldsymbol{u}_2 = \begin{bmatrix} -1 \\ 1 \\ 1 \end{bmatrix}$$

   Find the vector $\boldsymbol{x} \in U$ which is closest to the vector

$$\boldsymbol{b} = \begin{bmatrix} 1 \\ 2 \\ 1 \end{bmatrix}$$

4. Let $U \subset \mathbb{R}^3$ be the subspace spanned by

$$\boldsymbol{u}_1 = \begin{bmatrix} 1 \\ 1 \\ 1 \end{bmatrix} \quad \boldsymbol{u}_2 = \begin{bmatrix} 1 \\ 2 \\ 1 \end{bmatrix}$$

   Find the vector $\boldsymbol{x} \in U$ which is closest to the vector

$$\boldsymbol{b} = \begin{bmatrix} 1 \\ 1 \\ 2 \end{bmatrix}$$

5. Find an elementary reflector H such that $HA = \begin{bmatrix} * & * & * \\ 0 & * & * \\ 0 & * & * \\ 0 & * & * \end{bmatrix}$ where

   (a) $A = \begin{bmatrix} 1 & 2 & 1 \\ 0 & 0 & 1 \\ 2 & 2 & 1 \\ 2 & 1 & 0 \end{bmatrix}$

   (b) $A = \begin{bmatrix} 1 & 2 & 1 \\ 1 & 0 & 1 \\ 1 & 2 & 1 \\ 1 & 1 & 0 \end{bmatrix}$

6. Let $A = QR$ where

$$Q = \begin{bmatrix} 0 & 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 \end{bmatrix} \quad R = \begin{bmatrix} 1 & 1 & 1 & 1 \\ 0 & 1 & 1 & 1 \\ 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 \end{bmatrix}$$

   Find the least squares approximation of the system $A\boldsymbol{x} = \boldsymbol{b}$ for:

   (a) $\boldsymbol{b} = \begin{bmatrix} -2 & -1 & 0 & 1 & 2 \end{bmatrix}^T$.

   (b) $\boldsymbol{b} = \begin{bmatrix} 1 & -2 & 2 & 0 & 1 \end{bmatrix}^T$.

7. Setup (but do not solve) a linear system $B\boldsymbol{c} = \boldsymbol{y}$ where the solution is the coefficient vector $\boldsymbol{c} = [c_0 \;\; c_1 \;\; c_2]^T$ such that the function

$$f(t) = c_0 + c_1 \cos(2\pi t) + c_2 \sin(2\pi t)$$

   bests fits the data $(0, 1), (1/4, 3), (1/2, 2), (3/4, -1), (1, 0)$.

Chapter 3

Eigenvalue Problems

3.1 Review: Eigenvalues and Eigenvectors

Big Idea. An $n \times n$ matrix A is diagonalizable if there exist n linearly independent eigenvectors of A. If A is diagonalizable with $A = PDP^{-1}$ then the columns of P are eigenvectors of A and the diagonal entries of D are eigenvalues.

Definition. An **eigenvalue** of a square matrix A [KN, p.173] is a number λ such that

$$A\boldsymbol{v} = \lambda\boldsymbol{v}$$

for some nonzero vector $\boldsymbol{v}$. The vector $\boldsymbol{v}$ is called an **eigenvector** for the eigenvalue λ.

Note. If λ is an eigenvalue of A with eigenvector $\boldsymbol{v}$ then $(A - \lambda I)\boldsymbol{v} = \mathbf{0}$ which implies that $A - \lambda I$ is not invertible and therefore $\det(A - \lambda I) = 0$. This suggests that to find eigenvalues and eigenvectors of A we should:

1. Find λ such that $\det(A - \lambda I) = 0$.
2. Given λ, find solutions of the linear system $(A - \lambda I)\boldsymbol{v} = \mathbf{0}$.

This works when A is a small matrix and we have done this in previous linear algebra courses. However, this is impractical when A is a large matrix. For example, if A is $n \times n$ for $n \geq 5$, then $\det(A - \lambda I) = 0$ is a polynomial equation of degree n and there is no formula for the roots. We'll see better algorithms for computing eigenvalues in later sections.

Definition. Let A be an $n \times n$ matrix. The **characteristic polynomial** of A [KN, p.173] is

$$c_A(x) = \det(A - xI)$$

Then $c_A(x)$ has degree n and the roots of $c_A(x)$ are the eigenvalues of A.

Definition. A matrix A is **diagonalizable** [KN, p.178] if there exists an invertible matrix P and a diagonal matrix D such that $A = PDP^{-1}$.

Proposition. If A is diagonalizable with $A = PDP^{-1}$ then the diagonal entries of D are eigenvalues of A and the columns of P are eigenvectors [KN, p.179].

Proof. Let $\boldsymbol{v}_1, \ldots, \boldsymbol{v}_n$ be the columns of P and let $\lambda_1, \ldots, \lambda_n$ be the diagonal entries of D

$$P = \begin{bmatrix} \boldsymbol{v}_1 & \cdots & \boldsymbol{v}_n \end{bmatrix} \qquad D = \begin{bmatrix} \lambda_1 & & \\ & \ddots & \\ & & \lambda_n \end{bmatrix}$$

Matrix multiplication $AP = PD$ yields the equation

$$A\begin{bmatrix} & & \\ \boldsymbol{v}_1 & \cdots & \boldsymbol{v}_n \\ & & \end{bmatrix} = \begin{bmatrix} & & \\ \boldsymbol{v}_1 & \cdots & \boldsymbol{v}_n \\ & & \end{bmatrix}\begin{bmatrix} \lambda_1 & & \\ & \ddots & \\ & & \lambda_n \end{bmatrix}$$

$$\begin{bmatrix} & & \\ A\boldsymbol{v}_1 & \cdots & A\boldsymbol{v}_n \\ & & \end{bmatrix} = \begin{bmatrix} & & \\ \lambda_1\boldsymbol{v}_1 & \cdots & \lambda_n\boldsymbol{v}_n \\ & & \end{bmatrix}$$

Therefore $A\boldsymbol{v}_i = \lambda_i\boldsymbol{v}_i$ for each $i = 1, \ldots, n$. □

Proposition. If A has distinct eigenvalues, then A is diagonalizable [KN, p.181].

Proof. Let $\lambda_1, \ldots, \lambda_n$ be the distinct eigenvalues of A. That is, $\lambda_i \neq \lambda_j$ for $i \neq j$. Each λ_i has a corresponding eigenvector $\boldsymbol{v}_i$. Let $\boldsymbol{v}_i$ be the ith column of P and let λ_i be the ith diagonal entry of D. Then $A = PDP^{-1}$. □

Definition. Let λ be an eigenvalue of A. The **multiplicity** of λ [KN, p.180] is the number of times λ occurs as a root of the characteristic polynomial $c_A(x)$.

Note. Not every matrix is diagonalizable. For example, consider the matrix

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

Then $c_A(x) = (x-3)^2$ and there is only one eigenvalue $\lambda = 3$ and it has multiplicity 2. Solving the equation $(A - 3I)\boldsymbol{v} = \boldsymbol{0}$ yields only one independent solution

$$\boldsymbol{v} = \begin{bmatrix} 1 \\ 0 \end{bmatrix}$$

Therefore A does not have enough eigenvectors to be diagonalizable.

Theorem. A matrix A is diagonalizable if every eigenvalue λ with multiplicity m admits m linearly independent eigenvectors [KN, p.181].

3.2 Spectral Theorem

Big Idea. If A is a symmetric matrix then the eigenvalues of A are real numbers, eigenvectors (for distinct eigenvalues) are orthogonal and A is orthogonally diagonalizable $A = PDP^T$.

Proposition. All eigenvalues of a symmetric matrix are real numbers [KN, p.307].

Proof. Let λ be an eigenvalue of a symmetric matrix A with eigenvector $\boldsymbol{v}$. Compute $\boldsymbol{v} \cdot \overline{(A\boldsymbol{v})}$ in two different ways. First, compute

$$\boldsymbol{v} \cdot \overline{(A\boldsymbol{v})} = \boldsymbol{v} \cdot \overline{(\lambda \boldsymbol{v})} = \overline{\lambda}\, \boldsymbol{v} \cdot \overline{\boldsymbol{v}} = \overline{\lambda}\, \|\boldsymbol{v}\|^2$$

Now compute

$$\boldsymbol{v} \cdot \overline{(A\boldsymbol{v})} = \boldsymbol{v} \cdot (A\overline{\boldsymbol{v}}) = (A^T\boldsymbol{v}) \cdot \overline{\boldsymbol{v}} = (A\boldsymbol{v}) \cdot \overline{\boldsymbol{v}} = \lambda\, \boldsymbol{v} \cdot \overline{\boldsymbol{v}} = \lambda\, \|\boldsymbol{v}\|^2$$

Since $\|\boldsymbol{v}\| \neq 0$ we have $\lambda = \overline{\lambda}$ and therefore λ is a real number. □

Proposition. Let A be a symmetric matrix and let λ_1 and λ_2 be distinct eigenvalues of A with eigenvectors $\boldsymbol{v}_1$ and $\boldsymbol{v}_2$ respectively. Then $\boldsymbol{v}_1$ and $\boldsymbol{v}_2$ are orthogonal [KN, p.427].

Proof. Compute $(A\boldsymbol{v}_1) \cdot \boldsymbol{v}_2$ in two different ways. First, compute

$$(A\boldsymbol{v}_1) \cdot \boldsymbol{v}_2 = (\lambda_1 \boldsymbol{v}_1) \cdot \boldsymbol{v}_2 = \lambda_1\, \boldsymbol{v}_1 \cdot \boldsymbol{v}_2$$

Now compute

$$(A\boldsymbol{v}_1) \cdot \boldsymbol{v}_2 = \boldsymbol{v}_1 \cdot (A^T\boldsymbol{v}_2) = \boldsymbol{v}_1 \cdot (A\boldsymbol{v}_2) = \lambda_2\, \boldsymbol{v}_1 \cdot \boldsymbol{v}_2$$

Therefore

$$\lambda_1\, \boldsymbol{v}_1 \cdot \boldsymbol{v}_2 = \lambda_2\, \boldsymbol{v}_1 \cdot \boldsymbol{v}_2 \quad \Rightarrow \quad (\lambda_1 - \lambda_2)\, \boldsymbol{v}_1 \cdot \boldsymbol{v}_2 = 0 \quad \Rightarrow \quad \boldsymbol{v}_1 \cdot \boldsymbol{v}_2 = 0$$

since $\lambda_1 - \lambda_2 \neq 0$ because the eigenvalues are distinct. □

Theorem. Let A be a symmetric matrix. Then there exists an orthogonal matrix P and diagonal matrix D such that $A = PDP^T$. In other words, A is orthogonally diagonalizable [KN, p.425].

Note. If A is symmetric with $A = PDP^T$ then

$$P = \begin{bmatrix} & & \\ \boldsymbol{v}_1 & \cdots & \boldsymbol{v}_n \\ & & \end{bmatrix} \qquad D = \begin{bmatrix} \lambda_1 & & \\ & \ddots & \\ & & \lambda_n \end{bmatrix}$$

where $\lambda_1, \ldots, \lambda_n$ are the eigenvalues of A with corresponding orthonormal eigenvectors $\boldsymbol{v}_1, \ldots, \boldsymbol{v}_n$.

3.3 Singular Value Decomposition

Big Idea. Any $m \times n$ matrix A has a singular value decomposition $A = P\Sigma Q^T$ where P and Q are orthogonal matrices and Σ is a diagonal $m \times n$ matrix.

Note. If A is any $m \times n$ matrix, then AA^T and A^TA are both symmetric therefore both are orthogonally diagonalizable

$$AA^T = PD_1P^T \qquad A^TA = QD_2Q^T$$

Proposition. Let A be an $m \times n$ matrix.

1. If λ is a non-zero eigenvalue of AA^T then λ is an eigenvalue of A^TA, and vice versa.
2. All eigenvalues of AA^T (and A^TA) are non-negative (that is, $\lambda \geq 0$).

See [KN, p.446].

Definition. The matrices AA^T and A^TA have the same set of positive eigenvalues. Label the eigenvalues in decreasing order $\lambda_1 \geq \lambda_2 \geq \cdots \geq \lambda_r > 0$. Then

$$\sigma_i = \sqrt{\lambda_i} \quad , \quad i = 1, \ldots, r$$

are called the **singular values** of A [KN, p.447].

Theorem. Let A be an $m \times n$ matrix and let $\sigma_1 \geq \sigma_2 \geq \cdots \geq \sigma_r > 0$ be the singular values of A. Then there are orthogonal matrices P and Q such that

$$A = P\Sigma Q^T \quad \text{where} \quad \Sigma = \left[\begin{array}{ccc|c} \sigma_1 & & & \\ & \ddots & & \mathbf{0} \\ & & \sigma_r & \\ \hline & \mathbf{0} & & \mathbf{0} \end{array}\right]_{m \times n}$$

This is called the **singular value decomposition** of A [KN, p.449].

Proof. Let $\boldsymbol{q}_1, \ldots, \boldsymbol{q}_n$ be orthonormal eigenvectors of A^TA chosen in order such that

$$A^TA\boldsymbol{q}_i = \sigma_i^2\boldsymbol{q}_i \quad , \quad i = 1, \ldots, r \qquad\qquad A^TA\boldsymbol{q}_i = \mathbf{0} \quad , \quad i = r+1, \ldots, n$$

Note that in fact $A\boldsymbol{q}_i = \mathbf{0}$ for $i = r+1, \ldots, n$ since

$$\|A\boldsymbol{q}_i\|^2 = \boldsymbol{q}_i^TA^TA\boldsymbol{q}_i = 0 \quad , \quad i = r+1, \ldots, n$$

Let Q be the orthogonal matrix

$$Q = \begin{bmatrix} \boldsymbol{q}_1 & \cdots & \boldsymbol{q}_n \end{bmatrix}$$

Now construct the matrix P. Let

$$\boldsymbol{p}_i = \frac{1}{\sigma_i} A\boldsymbol{q}_i \quad , \quad i = 1, \ldots, r$$

Note that

$$AA^T\boldsymbol{p}_i = AA^T\left(\frac{1}{\sigma_i}A\boldsymbol{q}_i\right) = \frac{1}{\sigma_i}A\left(A^TA\boldsymbol{q}_i\right) = \sigma_i A\boldsymbol{q}_i = \sigma_i^2\boldsymbol{p}_i$$

therefore each $\boldsymbol{p}_i$ is an eigenvector for AA^T with eigenvalue σ_i^2. Note also that

$$\|\boldsymbol{p}_i\|^2 = \boldsymbol{p}_i^T\boldsymbol{p}_i = \frac{1}{\sigma_i^2}\boldsymbol{q}_i^T A^T A\boldsymbol{q}_i = \boldsymbol{q}_i^T\boldsymbol{q}_i = 1$$

therefore each $\boldsymbol{p}_i$ is a unit vector. Extend (by Gram-Schmidt algorithm) to an orthonormal basis $\boldsymbol{p}_1, \ldots, \boldsymbol{p}_r, \boldsymbol{p}_{r+1}, \ldots, \boldsymbol{p}_m$ of $\mathbb{R}^m$. Define the orthogonal matrix

$$P = \begin{bmatrix} \boldsymbol{p}_1 & \cdots & \boldsymbol{p}_m \end{bmatrix}$$

Compute

$$AQ = \begin{bmatrix} A\boldsymbol{q}_1 & \cdots & A\boldsymbol{q}_r & A\boldsymbol{q}_{r+1} & \cdots & A\boldsymbol{q}_n \end{bmatrix} = \begin{bmatrix} \sigma_1\boldsymbol{p}_1 & \cdots & \sigma_r\boldsymbol{p}_r & \mathbf{0} & \cdots & \mathbf{0} \end{bmatrix}$$

Finally, compute

$$P\Sigma = \begin{bmatrix} \boldsymbol{p}_1 & \cdots & \boldsymbol{p}_r & \boldsymbol{p}_{r+1} & \cdots & \boldsymbol{p}_n \end{bmatrix} \left[\begin{array}{ccc|c} \sigma_1 & & & \\ & \ddots & & \mathbf{0} \\ & & \sigma_r & \\ \hline & \mathbf{0} & & \mathbf{0} \end{array}\right] = \begin{bmatrix} \sigma_1\boldsymbol{p}_1 & \cdots & \sigma_r\boldsymbol{p}_r & \mathbf{0} & \cdots & \mathbf{0} \end{bmatrix}$$

Therefore $A = P\Sigma Q^T$. □

Note. In the construction of the SVD, we may chose to first construct either P or Q. The connection between the columns for $i = 1, \ldots, r$ are given by the equations:

$$\boldsymbol{q}_i = \frac{1}{\sigma_i}A^T\boldsymbol{p}_i \qquad A^TA\boldsymbol{q}_i = \sigma_i^2\boldsymbol{q}_i \qquad \|\boldsymbol{q}_i\| = 1$$

$$\boldsymbol{p}_i = \frac{1}{\sigma_i}A\boldsymbol{q}_i \qquad AA^T\boldsymbol{p}_i = \sigma_i^2\boldsymbol{p}_i \qquad \|\boldsymbol{p}_i\| = 1$$

Example. Construct the SVD for

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

Since $A^T A$ is a smaller matrix, let us first construct Q. Compute

$$A^T A = \begin{bmatrix} 2 & 0 \\ 0 & 3 \end{bmatrix}$$

Therefore $\sigma_1 = \sqrt{3}$ and $\sigma_2 = \sqrt{2}$. By inspection, we find

$$\boldsymbol{q}_1 = \begin{bmatrix} 0 \\ 1 \end{bmatrix} \quad \boldsymbol{q}_2 = \begin{bmatrix} 1 \\ 0 \end{bmatrix} \quad \Rightarrow \quad Q = \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix}$$

Construct the matrix P

$$\boldsymbol{p}_1 = \frac{1}{\sigma_1} A\boldsymbol{q}_1 = \frac{1}{\sqrt{3}} \begin{bmatrix} 1 \\ -1 \\ 1 \end{bmatrix} \quad \boldsymbol{p}_2 = \frac{1}{\sigma_2} A\boldsymbol{q}_2 = \frac{1}{\sqrt{2}} \begin{bmatrix} 1 \\ 1 \\ 0 \end{bmatrix}$$

Extend to an orthonormal basis of $\mathbb{R}^3$ by finding $\boldsymbol{p}_3$ orthogonal to $\boldsymbol{p}_1$ and $\boldsymbol{p}_2$. There are different ways of doing this. Setup equations $\boldsymbol{p}_1 \cdot \boldsymbol{p}_3 = 0$ and $\boldsymbol{p}_2 \cdot \boldsymbol{p}_3 = 0$ in a linear system and solve

$$\left[\begin{array}{ccc|c} 1 & -1 & 1 & 0 \\ 1 & 1 & 0 & 0 \end{array}\right] \quad \Rightarrow \quad \boldsymbol{p}_3 = \frac{1}{\sqrt{6}} \begin{bmatrix} -1 \\ 1 \\ 2 \end{bmatrix}$$

Therefore the SVD is given by

$$A = P\Sigma Q^T = \begin{bmatrix} 1/\sqrt{3} & 1/\sqrt{2} & -1/\sqrt{6} \\ -1/\sqrt{3} & 1/\sqrt{2} & 1/\sqrt{6} \\ 1/\sqrt{3} & 0 & 2/\sqrt{6} \end{bmatrix} \begin{bmatrix} \sqrt{3} & 0 \\ 0 & \sqrt{2} \\ 0 & 0 \end{bmatrix} \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix}^T$$

Proposition. Let $A = P\Sigma Q^T$.

- $\text{rank}(A) = r$
- $\|A\| = \sigma_1$
- $\|A^{-1}\| = 1/\sigma_r$
- $\text{cond}(A) = \sigma_1/\sigma_r$
- $\text{null}(A) = \text{span}\{\boldsymbol{q}_{r+1}, \ldots, \boldsymbol{q}_n\}$
- $\text{range}(A) = \text{span}\{\boldsymbol{p}_1, \ldots, \boldsymbol{p}_r\}$

3.4 Principal Component Analysis

Big Idea. An $n \times p$ data matrix X represents a set of n samples in $\mathbb{R}^p$ and projecting the data onto the first k principal components allows us to view the data in $\mathbb{R}^k$. Usually $k = 2$ such that we can visualize the data in 2D.

Definition. Let $\boldsymbol{x}_1, \ldots, \boldsymbol{x}_n \in \mathbb{R}^p$ (viewed as row vectors) and let X be the $n \times p$ data matrix where row k is given by $\boldsymbol{x}_k$. Assume the data is **normalized** such that the mean value of each column of X is 0. The unit vector $\boldsymbol{w}_1$ which maximizes the sum

$$\sum_{i=1}^{n} (\boldsymbol{x}_i \cdot \boldsymbol{w}_1)^2 = \|X\boldsymbol{w}_1\|^2$$

is called the **first weight vector** of X (see Wikipedia: Principal component analysis). More generally, given weight vectors $\boldsymbol{w}_1, \ldots, \boldsymbol{w}_{k-1}$, the k**th weight vector** of X is the unit vector $\boldsymbol{w}_k$ which maximizes

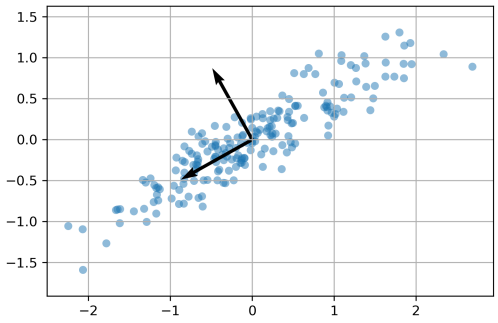
$$\|X_k\boldsymbol{w}_k\|^2$$

where X_k is the projection of the data matrix X onto $\text{span}\{\boldsymbol{w}_1, \ldots, \boldsymbol{w}_{k-1}\}^\perp$

$$X_k = X - \sum_{j=1}^{k-1} X\boldsymbol{w}_j\boldsymbol{w}_j^T$$

The projection coefficient $\boldsymbol{x}_i \cdot \boldsymbol{w}_k$ is called the k**th principal component** of a data vector $\boldsymbol{x}_i$.

Note. Each $(\boldsymbol{x}_k \cdot \boldsymbol{w}_1)^2$ is the length squared of the orthogonal projection of $\boldsymbol{x}_k$ onto $\boldsymbol{w}_1$. Therefore the first weight vector $\boldsymbol{w}_1$ points in the direction which captures the most information (ie. the maximum variance) of the data, and the second weight vector $\boldsymbol{w}_2$ is orthogonal to $\boldsymbol{w}_1$.



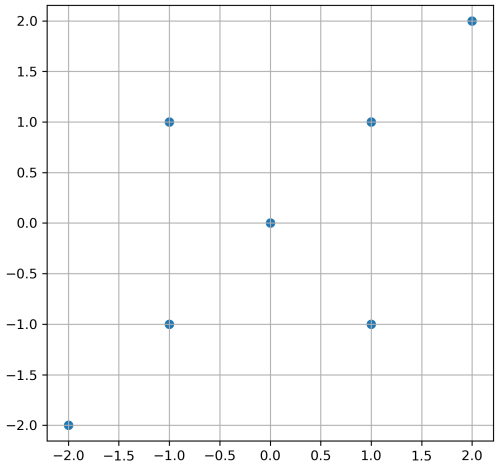
Proposition. The weight vectors $\boldsymbol{w}_1, \ldots, \boldsymbol{w}_p$ are the right singular vectors of the matrix X. In other words, let $X = P\Sigma Q^T$ be a singular value decomposition of X and let $\boldsymbol{q}_1, \ldots, \boldsymbol{q}_p$ be the columns of Q corresponding to singular values $\sigma_1 > \cdots > \sigma_p > 0$. Then $\boldsymbol{w}_1 = \boldsymbol{q}_1, \ldots, \boldsymbol{w}_p = \boldsymbol{q}_p$.

Proof. Let $X = P\Sigma Q^T$ be a singular value decomposition of X and note

$$\|X\boldsymbol{w}\|^2 = \|P\Sigma Q^T\boldsymbol{w}\|^2 = \|\Sigma Q^T\boldsymbol{w}\|^2$$

since P is orthogonal. Since Σ is diagonal with diagonal entires $\sigma_1 > \cdots > \sigma_p$, the maximum value of $\|X\boldsymbol{w}\|^2$ occurs when $Q^T\boldsymbol{w} = \begin{bmatrix}1 & 0 & \cdots & 0\end{bmatrix}^T$ therefore $\boldsymbol{w}_1 = \boldsymbol{q}_1$. For general k, note that the singular value decomposition $X_k = P_k\Sigma_k Q_k^T$ is obtained from X by removing the singular values $\sigma_1, \ldots, \sigma_{k-1}$. Therefore the largest singular value of X_k is σ_k with corresponding right singular vector $\boldsymbol{q}_k$ and therefore $\boldsymbol{w}_k = \boldsymbol{q}_k$. □

Example. Find the first weight vector for the data given in the image below.



We expect $\boldsymbol{w}_1 = \begin{bmatrix} 1/\sqrt{2} & 1/\sqrt{2} \end{bmatrix}^T$ since that direction clearly captures the most information. Form the data matrix

$$X^T = \begin{bmatrix} -2 & -1 & -1 & 0 & 1 & 1 & 2 \\ -2 & -1 & 1 & 0 & -1 & 1 & 2 \end{bmatrix}$$

We don't need to compute the full SVD of X but just the first right singular vector. Compute

$$X^T X = \begin{bmatrix} 12 & 8 \\ 8 & 12 \end{bmatrix}$$

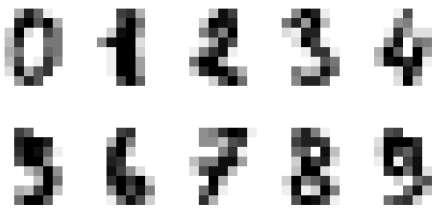
The characteristic polynomial of $X^T X$ is

$$\det(xI - X^T X) = (x-12)^2 - 8^2 = x^2 - 24x + 80 = (x-4)(x-20)$$

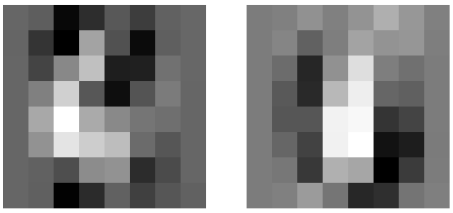
The right singular vector $\boldsymbol{q}_1$ for X is a unit eigenvector for $X^T X$ for the eigenvalue $\lambda_1 = 20$. Compute

$$(X^T X - 20I)\boldsymbol{w}_1 = \boldsymbol{0} \quad \Rightarrow \quad \left[\begin{array}{cc|c} -8 & 8 & 0 \\ 8 & -8 & 0 \end{array}\right] \quad \Rightarrow \quad \boldsymbol{w}_1 = \frac{1}{\sqrt{2}} \begin{bmatrix} 1 \\ 1 \end{bmatrix}$$

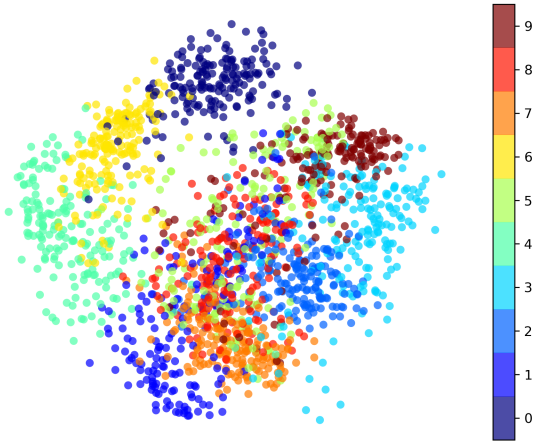
Example. The digits dataset from sklearn is a 1797×64 data matrix X such that each row represents an 8×8 pixel image of a handwritten number. The first 10 rows of X (reshaped from vectors of length 64 to 8×8 matrices to visualize) are:



Compute the first 2 weight vectors and find (again $\boldsymbol{w}_1, \boldsymbol{w}_2$ reshaped from vectors of length 64 to 8×8 matrices to visualize)



We can see $\boldsymbol{w}_1$ looks like a 3 and $\boldsymbol{w}_2$ looks like 0. Project the entire dataset onto these weight vectors and label each data point by a color according to the digit:



We can see that the 3s are to the right in the horizontal direction since these points most similar to $\boldsymbol{w}_1$, and the 0s are at the top in the vertical direction since these points most similar to $\boldsymbol{w}_2$. We can make other interesting observations such as the 4s are opposite to the 3s and orthogonal to 0s, and 7s and 1s are opposite to 0s and orthogonal to 3s.

3.5 Pseudoinverse, Least Squares and the SVD Expansion

Big Idea. The pseudoinverse A^+ solves the least squares problem $A\boldsymbol{x} \cong \boldsymbol{b}$ by $\boldsymbol{x} = A^+\boldsymbol{b}$.

Definition. Let A be a $m \times n$ matrix of rank r and let $A = P\Sigma Q^T$ be a SVD of A. The **pseudoinverse** of A [KN, p.458] is

$$A^+ = Q\Sigma^+ P^T$$

where

$$\Sigma^+ = \left[\begin{array}{ccc|c} \sigma_1^{-1} & & & \\ & \ddots & & \mathbf{0} \\ & & \sigma_r^{-1} & \\ \hline & \mathbf{0} & & \mathbf{0} \end{array}\right]_{n\times m}$$

Note. If A is invertible, then $A^+ = A^{-1}$.

Theorem. Let A be an $m \times n$ matrix and let $\boldsymbol{b} \in \mathbb{R}^m$. The least squares approximation of the system $A\boldsymbol{x} \cong \boldsymbol{b}$ is given by $\boldsymbol{x} = A^+\boldsymbol{b}$.

Proof. Let $A = P\Sigma Q^T$ be a SVD of A. Let $P^T\boldsymbol{b} = \boldsymbol{c}$ and write

$$\boldsymbol{c} = \begin{bmatrix} c_1 \\ \vdots \\ c_m \end{bmatrix} \in \mathbb{R}^m$$

Since P is an orthogonal matrix we have

$$\|A\boldsymbol{x} - \boldsymbol{b}\| = \|P\Sigma Q^T\boldsymbol{x} - P\boldsymbol{c}\| = \|\Sigma Q^T\boldsymbol{x} - \boldsymbol{c}\|$$

The matrix Σ is of the form

$$\Sigma = \left[\begin{array}{ccc|c} \sigma_1 & & & \\ & \ddots & & \mathbf{0} \\ & & \sigma_r & \\ \hline & \mathbf{0} & & \mathbf{0} \end{array}\right]_{m \times n}$$

and so only the first r entries of $\Sigma\boldsymbol{v}$ are nonzero for any vector $\boldsymbol{v} \in \mathbb{R}^n$. Therefore the minimum value $\|A\boldsymbol{x} - \boldsymbol{b}\| = \|\Sigma Q^T\boldsymbol{x} - \boldsymbol{c}\|$ occurs when

$$\Sigma Q^T\boldsymbol{x} = \begin{bmatrix} c_1 \\ \vdots \\ c_r \\ \mathbf{0} \end{bmatrix}$$

and so $\boldsymbol{x} = Q\Sigma^+\boldsymbol{c}$. Altogether, we have

$$\boldsymbol{x} = Q\Sigma^+P^T\boldsymbol{b} = A^+\boldsymbol{b}$$

□

Definition. Let $A = P\Sigma Q^T$. The **SVD expansion** of A is

$$A = \sum_{i=1}^{r} \sigma_i \boldsymbol{p}_i \boldsymbol{q}_i^T = \sigma_1 \boldsymbol{p}_1 \boldsymbol{q}_1^T + \cdots + \sigma_r \boldsymbol{p}_r \boldsymbol{q}_r^T$$

Note that each product $\boldsymbol{p}_i\boldsymbol{q}_i^T$ is a $m \times n$ matrix of rank 1.

Proposition. Let A be a $m \times n$ matrix and let B be a $n \times \ell$ matrix with SVD expansions

$$A = \sum_{i=1}^{r} \sigma_i \boldsymbol{p}_i \boldsymbol{q}_i^T \quad \text{and} \quad B = \sum_{j=1}^{s} \mu_j \boldsymbol{u}_j \boldsymbol{v}_j^T$$

If $\{\boldsymbol{q}_1, \ldots, \boldsymbol{q}_r\}$ and $\{\boldsymbol{u}_1, \ldots, \boldsymbol{u}_s\}$ are orthogonal sets of vectors, then $AB = 0$.

Definition. Let $A = P\Sigma Q^T$. The **truncated SVD expansion of rank** k of A is

$$A_k = \sum_{i=1}^{k} \sigma_i \boldsymbol{p}_i \boldsymbol{q}_i^T = \sigma_1 \boldsymbol{p}_1 \boldsymbol{q}_1^T + \cdots + \sigma_k \boldsymbol{p}_k \boldsymbol{q}_k^T$$

Note. Suppose we want to solve the system $A\boldsymbol{x} = \boldsymbol{b}$ however the right side is corrupted by noise $\boldsymbol{e}$ and we must work with the system

$$A\hat{\boldsymbol{x}} = \boldsymbol{b} + \boldsymbol{e}$$

Solving directly we get

$$\hat{\boldsymbol{x}} = A^{-1}(\boldsymbol{b} + \boldsymbol{e}) = A^{-1}\boldsymbol{b} + A^{-1}\boldsymbol{e}$$

and the term $A^{-1}\boldsymbol{e}$ is called the **inverted noise** which may dominate the true solution $\boldsymbol{x} = A^{-1}\boldsymbol{b}$. From the SVD expansion, we see that most of A is composed of the terms $\sigma_i \boldsymbol{p}_i \boldsymbol{q}_i^T$ for *large* singular values σ_i. If we know that the error $\boldsymbol{e}$ is unrelated to A in the sense that $\boldsymbol{e}$ is (mostly) orthogonal to the singular vectors $\boldsymbol{p}_i$ of A corresponding to large singular values, then the truncated SVD expansion of the pseudoinverse

$$A_k^+ = \sum_{i=1}^{k} \frac{1}{\sigma_i} \boldsymbol{q}_i \boldsymbol{p}_i^T$$

gives a better solution

$$\hat{\boldsymbol{x}} = A_k^+(\boldsymbol{b} + \boldsymbol{e}) = A_k^+\boldsymbol{b} + A_k^+\boldsymbol{e}$$

since the term $A_k^+\boldsymbol{e}$ will be smaller. In other words, we avoid terms $\sigma_i^{-1}\boldsymbol{p}_i\boldsymbol{q}_i^T$ in the SVD expansion of A^{-1} for small singular values σ_i which produce large values σ_i^{-1} which may amplify the error. This is the strategy for image debluring and computed tomography in the next sections.

3.6 Image Deblurring

Big Idea. Matrix multiplication by Toeplitz matrices perform blurring operations on a matrix $A_c X A_r^T = B$. Introducing some error E, we need to consider the truncated pseudoinverses $(A_c)_k^+$ and $(A_r)_k^+$ to reduce inverted noise.

Definition. A **Toeplitz matrix** [HNO, p.34] has constant values along the diagonals such as

$$\begin{bmatrix} p_3 & p_2 & p_1 & & \\ p_4 & p_3 & p_2 & p_1 & \\ p_5 & p_4 & p_3 & p_2 & p_1 \\ & p_5 & p_4 & p_3 & p_2 \\ & & p_5 & p_4 & p_3 \end{bmatrix}$$

Note. Multiplying on the left by a Toeplitz matrix will spread (or "blur") the values of a matrix X vertically in the columns. For example, consider the Toeplitz matrix A_c (where "c" stands for "columns") and image matrix X and compute

$$A_cX = \begin{bmatrix} 1 & 1 & 0 & 0 & 0 \\ 1 & 1 & 1 & 0 & 0 \\ 0 & 1 & 1 & 1 & 0 \\ 0 & 0 & 1 & 1 & 1 \\ 0 & 0 & 0 & 1 & 1 \end{bmatrix} \begin{bmatrix} 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 1 & 1 & 0 \\ 0 & 1 & 1 & 1 & 0 \\ 0 & 1 & 1 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 \end{bmatrix} = \begin{bmatrix} 0 & 1 & 1 & 1 & 0 \\ 0 & 2 & 2 & 2 & 0 \\ 0 & 3 & 3 & 3 & 0 \\ 0 & 2 & 2 & 2 & 0 \\ 0 & 1 & 1 & 1 & 0 \end{bmatrix}$$

Similarly, multiplying on the right by a Toeplitz matrix will spread the values of a matrix X horizontally in the rows. For example, consider the Toeplitz matrix A_r (where "r" stands for "rows") and image matrix X and compute

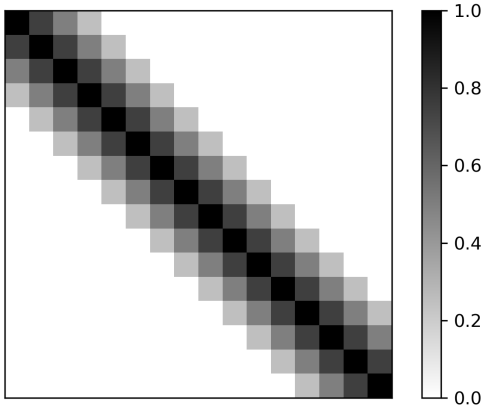
$$XA_r^T = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 1 & 1 & 0 \\ 0 & 1 & 1 & 1 & 0 \\ 0 & 1 & 1 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} 1 & 0.5 & 0 & 0 & 0 \\ 0.5 & 1 & 0.5 & 0 & 0 \\ 0 & 0.5 & 1 & 0.5 & 0 \\ 0 & 0 & 0.5 & 1 & 0.5 \\ 0 & 0 & 0 & 0.5 & 1 \end{bmatrix}^T = \begin{bmatrix} 0.0 & 0.0 & 0.0 & 0.0 & 0.0 \\ 0.5 & 1.5 & 2.0 & 1.5 & 0.5 \\ 0.5 & 1.5 & 2.0 & 1.5 & 0.5 \\ 0.5 & 1.5 & 2.0 & 1.5 & 0.5 \\ 0.0 & 0.0 & 0.0 & 0.0 & 0.0 \end{bmatrix}$$

By convention, we take the transpose of the matrix A_r when we use it to blur the rows.

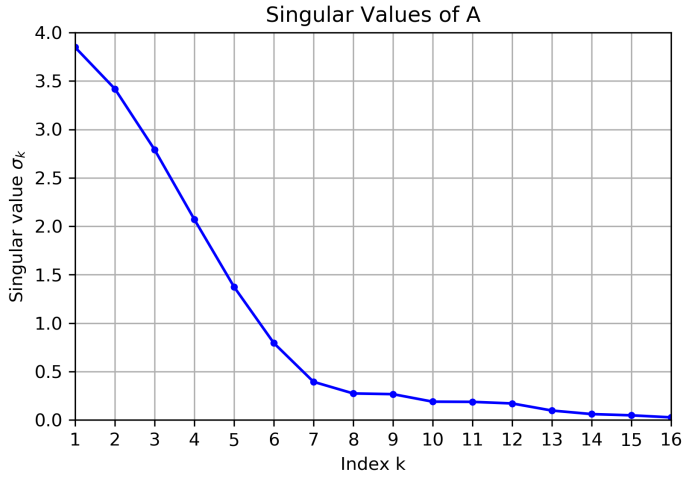
Example. Consider the symmetric Toeplitz matrix A where the first row is given by

$$\begin{bmatrix} 1.00 & 0.75 & 0.50 & 0.25 & 0.00 & \cdots & 0.00 \end{bmatrix}$$

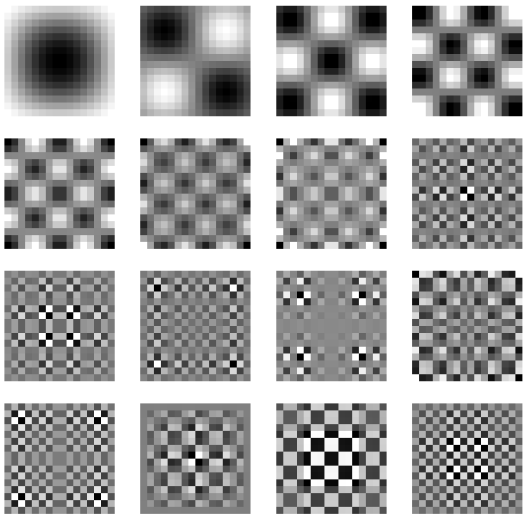
We can visualize the matrix A



Compute the singular value decomposition and plot the singular values

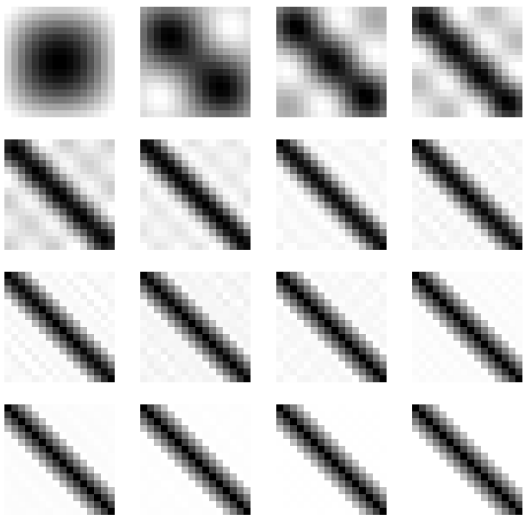


For each $i = 1, \ldots, 16$, plot the matrix $\sigma_i \boldsymbol{p}_i \boldsymbol{q}_i^T$:



For each $k = 1, \ldots, 16$, plot the truncated SVD expansion of rank k

$$A_k = \sum_{i=1}^{k} \sigma_i \boldsymbol{p}_i \boldsymbol{q}_i^T$$

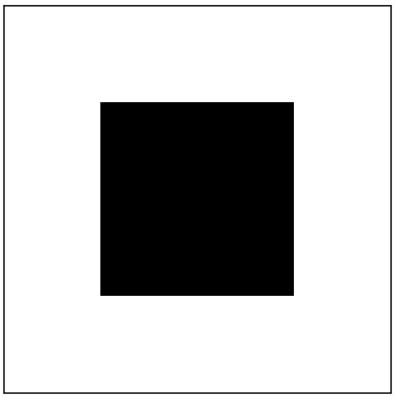


This shows that A_k is very close to A for $k \geq 12$ and therefore we can (and will) drop the

smallest singular values of A to compute the truncated pseudoinverse:

$$A_k^+ = \sum_{i=1}^{k} \frac{1}{\sigma_i} \boldsymbol{q}_i \boldsymbol{p}_i^T$$

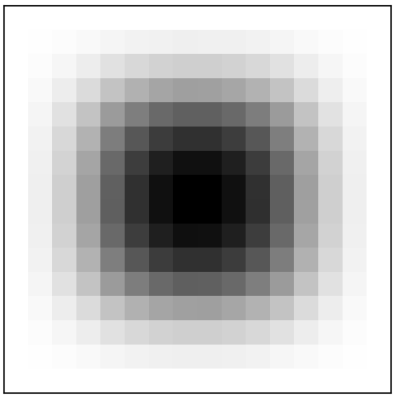
Example. Consider the symmetric Toeplitz matrix A from the previous example, and consider an image matrix X with a block of 1s in the center:



Apply A to both sides and denote the blurred image as B

$$AXA^T = B$$

Now suppose we add some random error E to the blurred image

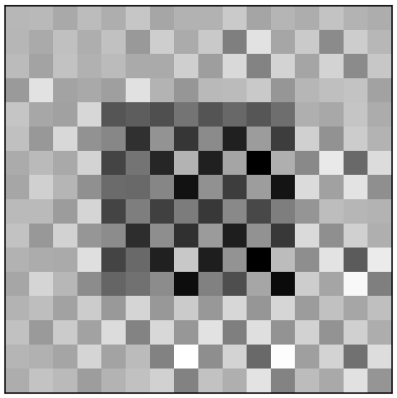


We want to recover the original image by solving the equation

$$A\hat{X}A^T = B + E$$

If we solve the equation directly we get the image plus the inverted error

$$\hat{X} = A^{-1}(B + E)(A^T)^{-1} = X + A^{-1}E(A^T)^{-1}$$

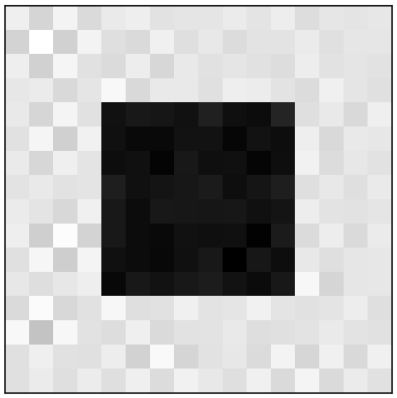


In the previous example, we found that A can be approximated by A_k therefore we form the truncated pseudoinverse

$$A_k^+ = \sum_{i=1}^{k} \frac{1}{\sigma_i} \boldsymbol{q}_i \boldsymbol{p}_i^T$$

and compute for $k = 12$

$$\hat{X} = A_k^+(B + E)(A^T)_k^+ = A_k^+ B(A^T)_k^+ + A_k^+ E(A^T)_k^+$$



The result is much better because the inverted error term $A_k^+ E(A^T)_k^+$ is smaller and so we avoid inverting too much of the error with small singular values.

3.7 Computed Tomography

Under construction

3.8 Computing Eigenvalues

Big Idea. It is not practical to compute eigenvalues of a matrix A by finding roots of the characteristic polynomial $c_A(x)$. Instead, there are several efficient algorithms for numerically approximating eigenvalues without using $c_A(x)$ such as the power method.

Definition. Let A be a square matrix. An eigenvalue λ of A is called a **dominant eigenvalue** [KN, p.441] if λ has multiplicity 1 and $|\lambda| > |\mu|$ for all other eigenvalues μ.

Definition. Let A be an $n \times n$ matrix with eigenvalues $\lambda_1, \ldots, \lambda_n$ and corresponding eigenvectors $\boldsymbol{v}_1, \ldots, \boldsymbol{v}_n$ with dominant eigenvalue λ_1. Let $\boldsymbol{x}_0$ be any vector which is a linear combination of the eigenvectors of A

$$\boldsymbol{x}_0 = c_1\boldsymbol{v}_1 + \cdots + c_n\boldsymbol{v}_n$$

such that $c_1 \neq 0$. Then

$$A^k\boldsymbol{x}_0 = c_1\lambda_1^k\boldsymbol{v}_1 + \cdots + c_n\lambda_n^k\boldsymbol{v}_n$$

and therefore

$$(1/\lambda_1^k)A^k\boldsymbol{x}_0 = c_1\boldsymbol{v}_1 + c_2(\lambda_2/\lambda_1)^k\boldsymbol{v}_2 + \cdots + c_n(\lambda_n/\lambda_1)^k\boldsymbol{v}_n \quad \to \quad c_1\boldsymbol{v}_1 \text{ as } k \to \infty$$

because each term $|\lambda_i/\lambda_1| < 1$ and so $\lambda_i/\lambda_1 \to 0$ as $k \to \infty$. This method of approximating $\boldsymbol{v}_1$ is called **power iteration** [MH, p.172] (or the **power method**).

Note. The entries in the vector $A^k\boldsymbol{x}_0$ may get very large as k increases therefore it is helpful to normalize at each step. The simplest way is to divide by the ∞-norm

$$\boldsymbol{x}_{k+1} = \frac{A\boldsymbol{x}_k}{\|A\boldsymbol{x}_k\|_\infty}$$

This is called **normalized power iteration** [MH, p.174]. Note that $\|A\boldsymbol{x}_k\|_\infty$ gives an approximation of λ_1 at each step.

Example. Approximate the dominant eigenvalue and eigenvector of the matrix

$$A = \begin{bmatrix} 1 & 1 & 0 \\ 1 & 1 & 1 \\ 0 & 1 & 1 \end{bmatrix}$$

by 4 iterations of the normalized power method. Choose a random starting vector

$$\boldsymbol{x}_0 = \begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix}$$

and compute

$$A\boldsymbol{x}_0 = \begin{bmatrix} 1 & 1 & 0 \\ 1 & 1 & 1 \\ 0 & 1 & 1 \end{bmatrix} \begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix} = \begin{bmatrix} 1 \\ 1 \\ 0 \end{bmatrix} \qquad \boldsymbol{x}_1 = \frac{A\boldsymbol{x}_0}{\|A\boldsymbol{x}_0\|_\infty} = \begin{bmatrix} 1 \\ 1 \\ 0 \end{bmatrix}$$

$$A\boldsymbol{x}_1 = \begin{bmatrix} 1 & 1 & 0 \\ 1 & 1 & 1 \\ 0 & 1 & 1 \end{bmatrix} \begin{bmatrix} 1 \\ 1 \\ 0 \end{bmatrix} = \begin{bmatrix} 2 \\ 2 \\ 1 \end{bmatrix} \qquad \boldsymbol{x}_2 = \frac{A\boldsymbol{x}_1}{\|A\boldsymbol{x}_1\|_\infty} = \begin{bmatrix} 1 \\ 1 \\ 0.5 \end{bmatrix}$$

$$A\boldsymbol{x}_2 = \begin{bmatrix} 1 & 1 & 0 \\ 1 & 1 & 1 \\ 0 & 1 & 1 \end{bmatrix} \begin{bmatrix} 1 \\ 1 \\ 0.5 \end{bmatrix} = \begin{bmatrix} 2 \\ 2.5 \\ 1.5 \end{bmatrix} \qquad \boldsymbol{x}_3 = \frac{A\boldsymbol{x}_2}{\|A\boldsymbol{x}_2\|_\infty} = \begin{bmatrix} 0.8 \\ 1 \\ 0.6 \end{bmatrix}$$

$$A\boldsymbol{x}_3 = \begin{bmatrix} 1 & 1 & 0 \\ 1 & 1 & 1 \\ 0 & 1 & 1 \end{bmatrix} \begin{bmatrix} 0.8 \\ 1 \\ 0.6 \end{bmatrix} = \begin{bmatrix} 1.8 \\ 2.4 \\ 1.6 \end{bmatrix} \qquad \boldsymbol{x}_4 = \frac{A\boldsymbol{x}_3}{\|A\boldsymbol{x}_3\|_\infty} = \begin{bmatrix} 0.75 \\ 1 \\ 0.67 \end{bmatrix}$$

Therefore we get approximations

$$\lambda_1 \approx 2.4 \qquad \boldsymbol{v}_1 \approx \begin{bmatrix} 0.75 \\ 1 \\ 0.67 \end{bmatrix}$$

The actual dominant eigenvector is

$$\boldsymbol{v}_1 = \begin{bmatrix} 1/\sqrt{2} \\ 1 \\ 1/\sqrt{2} \end{bmatrix}$$

and we verify

$$\begin{bmatrix} 1 & 1 & 0 \\ 1 & 1 & 1 \\ 0 & 1 & 1 \end{bmatrix} \begin{bmatrix} 1/\sqrt{2} \\ 1 \\ 1/\sqrt{2} \end{bmatrix} = \begin{bmatrix} \frac{1+\sqrt{2}}{\sqrt{2}} \\ 1+\sqrt{2} \\ \frac{1+\sqrt{2}}{\sqrt{2}} \end{bmatrix} = (1+\sqrt{2}) \begin{bmatrix} \frac{1}{\sqrt{2}} \\ 1 \\ \frac{1}{\sqrt{2}} \end{bmatrix}$$

therefore $\lambda_1 \approx 2.4142$.

Definition. Let A be an $n \times n$ matrix with eigenvalues $\lambda_1, \ldots, \lambda_n$ (in increasing order $\lambda_1 < \lambda_2 < \cdots < \lambda_n$) with corresponding eigenvectors $\boldsymbol{v}_1, \ldots, \boldsymbol{v}_n$. Then $1/\lambda_1, \ldots, 1/\lambda_n$ are the eigenvalues of A^{-1} (in decreasing order) with corresponding eigenvectors $\boldsymbol{v}_1, \ldots, \boldsymbol{v}_n$. **Inverse iteration** [MH, p.175] is power iteration applied to A^{-1} to find the dominant eigenvalue $1/\lambda_n$ of A^{-1} (equivalently, the smallest eigenvalue λ_n of A) with eigenvector $\boldsymbol{v}_n$. At each step, solve the system and normalize

$$\boldsymbol{y}_{k+1} = A^{-1}\boldsymbol{x}_k \quad \Rightarrow \quad A\boldsymbol{y}_{k+1} = \boldsymbol{x}_k \quad \Rightarrow \quad \boldsymbol{x}_{k+1} = \frac{\boldsymbol{y}_{k+1}}{\|\boldsymbol{y}_{k+1}\|_\infty}$$

Example. Compute 2 steps of inverse iterations for the matrix

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

Since we are going to repeatedly solve systems $A\boldsymbol{x} = \boldsymbol{b}$, we should find the LU decomposition and use forward and backward substitution

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

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

Therefore

$$A = LU = \begin{bmatrix} 1 & 0 & 0 \\ 1 & 1 & 0 \\ 1 & -1 & 1 \end{bmatrix} \begin{bmatrix} 1 & 1 & 1 \\ 0 & 1 & -1 \\ 0 & 0 & 1 \end{bmatrix}$$

and in fact we see that A is positive definite and this is the Cholesky decomposition. Choose a random starting vector

$$\boldsymbol{x}_0 = \begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix}$$

and compute

$$\begin{bmatrix} 1 & 0 & 0 \\ 1 & 1 & 0 \\ 1 & -1 & 1 \end{bmatrix} \boldsymbol{z}_1 = \begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix} \qquad \boldsymbol{z}_1 = \begin{bmatrix} 1 \\ -1 \\ -2 \end{bmatrix}$$

$$\begin{bmatrix} 1 & 1 & 1 \\ 0 & 1 & -1 \\ 0 & 0 & 1 \end{bmatrix} \boldsymbol{y}_1 = \begin{bmatrix} 1 \\ -1 \\ -2 \end{bmatrix} \qquad \boldsymbol{y}_1 = \begin{bmatrix} 6 \\ -3 \\ -2 \end{bmatrix} \qquad \boldsymbol{x}_1 = \begin{bmatrix} 1 \\ -1/2 \\ -1/3 \end{bmatrix}$$

$$\begin{bmatrix} 1 & 0 & 0 \\ 1 & 1 & 0 \\ 1 & -1 & 1 \end{bmatrix} \boldsymbol{z}_2 = \begin{bmatrix} 1 \\ -1/2 \\ -1/3 \end{bmatrix} \qquad \boldsymbol{z}_2 = \begin{bmatrix} 1 \\ -3/2 \\ -17/6 \end{bmatrix}$$

$$\begin{bmatrix} 1 & 1 & 1 \\ 0 & 1 & -1 \\ 0 & 0 & 1 \end{bmatrix} \boldsymbol{y}_2 = \begin{bmatrix} 1 \\ -3/2 \\ -17/6 \end{bmatrix} \qquad \boldsymbol{y}_2 = \begin{bmatrix} 49/6 \\ -13/3 \\ -17/6 \end{bmatrix} \qquad \boldsymbol{x}_2 = \begin{bmatrix} 1 \\ -26/49 \\ -17/49 \end{bmatrix}$$

Our approximation of the eigenvector of A corresponding to the smallest eigenvalue is

$$\boldsymbol{v} = \begin{bmatrix} 1 \\ -26/49 \\ -17/49 \end{bmatrix} \approx \begin{bmatrix} 1.00 \\ -0.53 \\ -0.35 \end{bmatrix}$$

with eigenvalue $\lambda \approx 0.12$ given by

$$\begin{bmatrix} 1 & 1 & 1 \\ 1 & 2 & 0 \\ 1 & 0 & 3 \end{bmatrix} \begin{bmatrix} 1.00 \\ -0.53 \\ -0.35 \end{bmatrix} = \begin{bmatrix} 0.12 \\ -0.06 \\ -0.05 \end{bmatrix} = 0.12 \begin{bmatrix} 1.00 \\ -0.50 \\ -0.42 \end{bmatrix}$$

The actual eigenvector is approximately

$$\boldsymbol{v} \approx \begin{bmatrix} 1.000 \\ -0.532 \\ -0.347 \end{bmatrix}$$

with eigenvalue

$$\lambda \approx 0.12061476$$

3.9 PageRank Beyond the Web

Big Idea. The PageRank vector is the dominant eigenvector of the adjacency matrix of a directed graph and it ranks the importance of each vertex in the graph.

Definition. Consider a directed graph G with N vertices (see Wikipedia: Directed graph). The **adjacency matrix** is the $N \times N$ matrix $A = [a_{i,j}]$ where

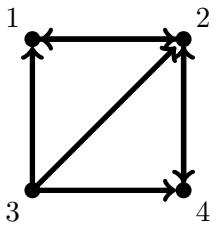
$$a_{i,j} = \begin{cases} 1 & \text{if there is an edge to vertex } i \text{ from vertex } j \\ 0 & \text{if not} \end{cases}$$

Suppose the vertices of G represent a collection webpages and the edges represent links from one webpage to another. (We only count one link maximum from one webpage to another and no links from a webpage to itself.) The **stochastic matrix** [KN, p.134] of G represents the process of clicking a random link on a webpage and is given by $P = [p_{i,j}]$ where

$$p_{i,j} = \frac{a_{i,j}}{\text{total \# of links from webpage } j}$$

The entry $p_{i,j}$ is the probability of clicking to webpage i from webpage j.

Example. Consider the directed graph



Construct the adjacency matrix A and the stochastic matrix P

$$A = \begin{bmatrix} 0 & 1 & 1 & 0 \\ 1 & 0 & 1 & 1 \\ 0 & 0 & 0 & 0 \\ 0 & 1 & 1 & 0 \end{bmatrix} \qquad P = \begin{bmatrix} 0 & 1/2 & 1/3 & 0 \\ 1 & 0 & 1/3 & 1 \\ 0 & 0 & 0 & 0 \\ 0 & 1/2 & 1/3 & 0 \end{bmatrix}$$

Definition. The **Google matrix** of a directed graph G is

$$\alpha P + (1-\alpha)\boldsymbol{v}\boldsymbol{e}^T$$

where P is the stochastic matrix of G, $0 < \alpha < 1$ is the **teleportation parameter** [DG, p.322], $\boldsymbol{v}$ is the **teleportation distribution vector** and $\boldsymbol{e}^T = \begin{bmatrix} 1 & \cdots & 1 \end{bmatrix}$ is a vector of 1s. Note $\boldsymbol{v}\boldsymbol{e}^T$ is the matrix with vector $\boldsymbol{v}$ in every column.

Note. The teleportation vector $\boldsymbol{v}$ has entries between 0 and 1 and the entries sum to 1. In other words, it is a stochastic vector. The vector $\boldsymbol{v}$ is usually chosen to be $\boldsymbol{v} = (1/N)\boldsymbol{e}$ where N is the number of vertices in the graph. The stochastic matrix $\boldsymbol{v}\boldsymbol{e}^T$ then represents the process of transitioning to a random webpage with uniform probability. The Google matrix is a stochastic matrix which represents the process: at each step, do either:

- probability α: click a random link on the webpage to visit another webapge
- probability $1-\alpha$: teleport to any webpage according to the distribution $\boldsymbol{v}$

The teleportation parameter α is usually chosen to be $\alpha = 0.85$.

Proposition. Let G be a directed graph and let P be the stochastic matrix for G. Choose parameters $0 < \alpha < 0$ and $\boldsymbol{v}$. There exists a unique steady state vector $\boldsymbol{x}$ (with entries between 0 and 1 and the entries sum to 1) such that

$$\left(\alpha P + (1-\alpha)\boldsymbol{v}\boldsymbol{e}^T\right)\boldsymbol{x} = \boldsymbol{x}$$

The vector $\boldsymbol{x}$ is called the **PageRank** vector and the entry x_i is the PageRank of the webpage at vertex i. The Google search result lists the webpages in order of their PageRank.

Note. A directed graph G represents a collection of webpages that contain the words in a Google search. The PageRank vector ranks the importance of the webpages for the search. There are usually hundreds of millions webpages in the graph therefore the Google matrix is HUGE! But the founders of Google showed that the power iteration algorithm converges well enough after about 50 iterations to find the webpages with the top PageRank.

Example. Find the Google matrix for the directed graph in the example above for $\alpha = 0.85$ and $\boldsymbol{v} = (1/N)\boldsymbol{e}$. Compute

$$\alpha P + (1-\alpha)\boldsymbol{v}\boldsymbol{e}^T = 0.85 \begin{bmatrix} 0 & 1/2 & 1/3 & 0 \\ 1 & 0 & 1/3 & 1 \\ 0 & 0 & 0 & 0 \\ 0 & 1/2 & 1/3 & 0 \end{bmatrix} + \frac{0.15}{4} \begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 \end{bmatrix}$$

$$= \begin{bmatrix} 0.0375 & 0.4625 & 0.3208 & 0.0375 \\ 0.8875 & 0.0375 & 0.3208 & 0.8875 \\ 0.0375 & 0.0375 & 0.0375 & 0.0375 \\ 0.0375 & 0.4625 & 0.3208 & 0.0375 \end{bmatrix}$$

Compute 50 iterations of the power method to approximate the PageRank vector

$$\boldsymbol{x} \approx \begin{bmatrix} 0.2472 \\ 0.4681 \\ 0.0375 \\ 0.2472 \end{bmatrix}$$

Clearly, vertex 2 is the most important in the graph.

3.10 Exercises

1. Determine whether the statement is **True** or **False**.

   (a) Let $\boldsymbol{v}_1, \boldsymbol{v}_2 \in \mathbb{R}^2$ be linearly independent vectors. Let λ_1, λ_2 be real numbers. Then there exists a unique 2 by 2 matrix A with eigenvalues λ_1, λ_2 and corresponding eigenvectors $\boldsymbol{v}_1, \boldsymbol{v}_2$.

   (b) The singular value decomposition $A = P\Sigma Q^T$ is unique.

   (c) The inverse iteration algorithm (without normalization) computes a recursive sequence $A\boldsymbol{x}_{k+1} = \boldsymbol{x}_k$ where $\boldsymbol{x}_k$ converges to:

      i. the largest (in absolute value $|\lambda|$) eigenvalue of A
      ii. an eigenvector corresponding to the largest (in absolute value $|\lambda|$) eigenvalue of A
      iii. the smallest (in absolute value $|\lambda|$) eigenvalue of A
      iv. an eigenvector corresponding to the smallest (in absolute value $|\lambda|$) eigenvalue of A

   (d) In the power iteration algorithm, we divide by $\|A\boldsymbol{x}_k\|_\infty$ in each step to:

      i. make the algorithm run faster
      ii. prevent the entries of the vectors $\boldsymbol{x}_k$ from becoming too large/small
      iii. produce a more accurate result

   (e) Let λ be a (nonzero) eigenvalue of an invertible matrix A.

      i. λ^{-1} is an eigenvalue of A^{-1}
      ii. λ is an eigenvalue of A^T
      iii. λ^2 is an eigenvalue of AA^T
      iv. λ is an eigenvalue of PAP^{-1} for any invertible matrix P
      v. $\lambda \neq 0$

   (f) Let $\boldsymbol{u} \in \mathbb{R}^n$ be a nonzero vector and let $H = I - \frac{2}{\|\boldsymbol{u}\|^2}\boldsymbol{u}\boldsymbol{u}^T$ be the corresponding elementary reflector. Then $\lambda = -1$ is an eigenvalue of H with multiplicity 1.

   (g) Let $U \subset \mathbb{R}^n$ be a subspace with $\dim(U) = m$ such that $0 < m < n$, and let P be the orthogonal projection matrix onto U. Then $\lambda = 0$ is an eigenvalue for P with multiplicity m.

   (h) Suppose A and B are symmetric $n \times n$ matrices. Then the eigenvectors of AB corresponding to distinct eigenvalues are orthogonal.

   (i) Let A be any $m \times n$ matrix. If λ is an eigenvalue of AA^T then λ is a real number and $\lambda \geq 0$.

   (j) Let A be any $m \times n$ matrix. If $\boldsymbol{v}_1, \boldsymbol{v}_2$ are eigenvectors of AA^T for distinct eigenvalues then $\boldsymbol{v}_1 \cdot \boldsymbol{v}_1 = 0$.

   (k) Let $\boldsymbol{u} \in \mathbb{R}^n$ and let $H = I - \frac{2}{\|\boldsymbol{u}\|^2}\boldsymbol{u}\boldsymbol{u}^T$ be the corresponding elementary reflector. The characteristic polynomial of H is $(x-1)^{n-1}(x+1)$.

   (l) Let P be an orthogonal projection matrix. All the eigenvalues of P are either 1 or 0.

   (m) Let λ be an eigenvalue of an invertible matrix A. Identify all True statements:

      i. λ^{-1} is an eigenvalue of A^{-1}
      ii. λ is an eigenvalue of A^T
      iii. λ^2 is an eigenvalue of AA^T

iv. λ is an eigenvalue of PAP^{-1} for any invertible matrix P

v. $\lambda \neq 0$

2. Let A be a 2×2 matrix with eigenvalues $\lambda_1 = 1$ and $\lambda_2 = 1/2$ and corresponding eigenvectors

$$\boldsymbol{v}_1 = \begin{bmatrix} 1 \\ 2 \end{bmatrix} \qquad \boldsymbol{v}_2 = \begin{bmatrix} -1 \\ 1 \end{bmatrix}$$

If we choose $\boldsymbol{x}_0 = \begin{bmatrix} 1 \\ 5 \end{bmatrix}$ then the sequence $\boldsymbol{x}_{k+1} = A\boldsymbol{x}_k$ converges to what?

3. Find the singular value decomposition of $A = \begin{bmatrix} 1 & 2 & -1 \\ 2 & 1 & 4 \end{bmatrix}$.

4. Suppose A is a symmetric 3×3 matrix with distinct eigenvalues $\lambda_1, \lambda_2, \lambda_3$ and eigenvectors

$$\boldsymbol{v}_1 = \begin{bmatrix} 1 \\ 1 \\ 1 \end{bmatrix} \qquad \boldsymbol{v}_2 = \begin{bmatrix} 1 \\ 0 \\ -1 \end{bmatrix}$$

Determine eigenvector $\boldsymbol{v}_3$ for eigenvalue λ_3.

5. Find the singular value decomposition of $A = \begin{bmatrix} 1 & 1 & 1 \\ -1 & 2 & -1 \\ 1 & 0 & -1 \end{bmatrix}$.

6. Determine whether the statement is **True** or **False**. (Assume all data matrices are normalized.)

   (a) Let X be a $n \times p$ data matrix and let $\boldsymbol{x}_i, \boldsymbol{x}_j \in \mathbb{R}^p$ be two different rows of X such that $\|\boldsymbol{x}_i\| < \|\boldsymbol{x}_j\|$. If $\boldsymbol{w}_1$ is the first weight vector of X, then $|\boldsymbol{x}_i \cdot \boldsymbol{w}_1| < |\boldsymbol{x}_j \cdot \boldsymbol{w}_1|$.

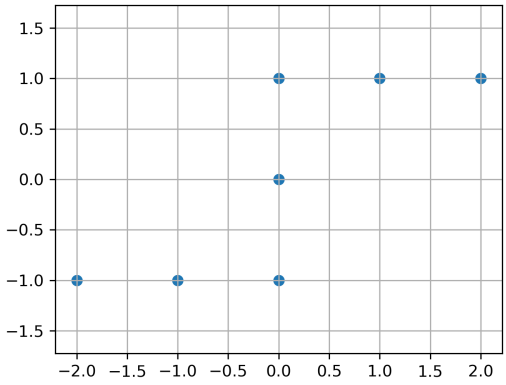
   (b) Let X be a $n \times p$ data matrix and let $\boldsymbol{x}_i, \boldsymbol{x}_j \in \mathbb{R}^p$ be two different rows of X such that $\boldsymbol{x}_i \cdot \boldsymbol{x}_j = 0$. If $\boldsymbol{w}_1$ is the first weight vector of X and $\boldsymbol{x}_i \cdot \boldsymbol{w}_1 = 0$ then $\boldsymbol{x}_j \cdot \boldsymbol{w}_1 = 0$.

   (c) Let X be a $n \times 2$ data matrix and let Y be the matrix with the same columns as X but switched. (In other words, the first column of Y is the same as the second column of X, and the second column of Y is the first column of X.) If X and Y represent the same set of data points, then all the singular values of X equal.

   (d) Let X be a $n \times 2$ data matrix and let Y be the matrix with the same columns as X but switched. (In other words, the first column of Y is the same as the second column of X, and the second column of Y is the first column of X.) If X and Y represent the same set of data points, then $\boldsymbol{w}_1 = \begin{bmatrix} 1/\sqrt{2} & 1/\sqrt{2} \end{bmatrix}^T$.

   (e) It is necessary to compute all the eigenvectors of the Google matrix to find the PageRank vector of a directed graph.

7. Find the weight vectors for the data matrix X representing the points:



8. Suppose X is a 100×4 data matrix such that

$$X^T X = \begin{bmatrix} 2 & 0 & 0 & 0 \\ 0 & 1.5 & 0 & 0 \\ 0 & 0 & 2 & 1 \\ 0 & 0 & 1 & 2 \end{bmatrix}$$

Find all the weight vectors of X.

9. Suppose we want to solve a system $A\boldsymbol{x} = \boldsymbol{b}$. A small change $\Delta\boldsymbol{b}$ produces a change in the solution

$$A(\boldsymbol{x} + \Delta\boldsymbol{x}) = \boldsymbol{b} + \Delta\boldsymbol{b}$$

Describe the unit vector $\Delta\boldsymbol{b}$ that will produce the largest change $\|\Delta\boldsymbol{x}\|$.

10. Find the rank 2 pseudo inverse

$$A_2^+ = \frac{1}{\sigma_1}\boldsymbol{q}_1\boldsymbol{p}_1^T + \frac{1}{\sigma_2}\boldsymbol{q}_2\boldsymbol{p}_2^T$$

of the matrix

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

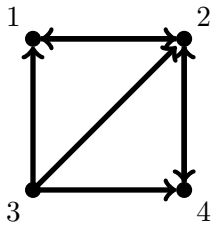
(Note: the columns of A are orthogonal.)

11. Let A be a $m \times n$ matrix with singular value decomposition $A = P\Sigma Q^T$. Let $k < \min\{m, n\}$ and let

$$A_k = \sum_{i=1}^{k} \sigma_i \boldsymbol{p}_i \boldsymbol{q}_i^T$$

Describe the singular value decomposition of $A - A_k$.

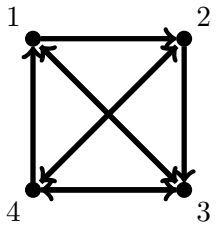
12. Consider the same directed graph as in the example in the section on PageRank:



As $\alpha \to 1$, describe what happens to the PageRank x_3 of vertex 3.

13. Let G be the complete directed graph with N vertices. In other words, there is an edge from each vertex to every other vertex in G (excluding edges from a vertex to itself). Describe the Google matrix and the PageRank vector for the complete directed graph.

14. Find the Google matrix $\alpha P + (1-\alpha)\boldsymbol{v}\boldsymbol{e}^T$ for the directed graph



using teleportation parameter $\alpha = 0.5$ and uniform distribution vector $\boldsymbol{v}$. Let $\boldsymbol{x}_0 = \begin{bmatrix}1 & 0 & 0 & 0\end{bmatrix}^T$ and use Python to compute 50 iterations of the power method to approximate the PageRank vector.

15. Find the Google matrix $\alpha P + (1-\alpha)\boldsymbol{v}\boldsymbol{e}^T$ for the directed graph in the previous exercise using teleportation parameter $\alpha = 0.8$ and distribution vector $\boldsymbol{v} = \begin{bmatrix}0 & 1/2 & 1/2 & 0\end{bmatrix}^T$. Let $\boldsymbol{x}_0 = \begin{bmatrix}1 & 0 & 0 & 0\end{bmatrix}^T$ and use Python to compute 50 iterations of the power method to approximate the PageRank vector.

Chapter 4

Discrete Fourier Transform

4.1 Review: Complex Numbers, Vectors and Matrices

Big Idea. A complex number can be represented in the form $z = a + ib$ and also in polar form $z = re^{i\theta}$. The set of vectors of length n with complex entries is a complex vector space $\mathbb{C}^n$ with inner product $\langle \boldsymbol{u}, \boldsymbol{v} \rangle = \boldsymbol{u}^T \overline{\boldsymbol{v}}$.

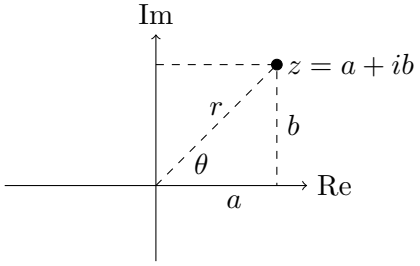
Definition. A **complex number** [KN, p.597] is of the form

$$z = a + ib$$

where $i = \sqrt{-1}$ and $a, b \in \mathbb{R}$. The complex number i satisfies $i^2 = -1$, the real number a is called the **real part** of z and b is the **imaginary part**, and we write $\mathrm{Re}(z) = a$ and $\mathrm{Im}(z) = b$. The **polar form** [KN, p.601] of a complex number $z = a + ib$ is

$$z = re^{i\theta}$$

where $r = \sqrt{a^2 + b^2}$ and $\theta = \arctan(b/a)$. We visualize the **set of complex numbers** $\mathbb{C}$ as a 2-dimensional real vector space:



Theorem. Euler's formula is

$$e^{i\theta} = \cos\theta + i\sin\theta$$

See [KN, p.602].

Definition. Let $z = a + ib$ and $z = re^{i\theta}$ in polar form.

1. The **modulus** of z is $|z| = r = \sqrt{a^2 + b^2}$.
2. The **angle** (or **argument**) of z is $\angle z = \theta = \arctan(b/a)$ (or $\arg(z) = \theta$).
3. The **conjugate** of z is $\overline{z} = a - ib = re^{-i\theta}$.

See [KN, p.599].

Proposition.

$$z^{-1} = \frac{\overline{z}}{|z|^2}$$

Proof. Let $z = a + ib$. Then

$$z^{-1} = \frac{1}{z} = \frac{\overline{z}}{z\overline{z}}$$

and we see

$$z\overline{z} = (a + ib)(a - ib) = a^2 + b^2 = |z|^2$$

□

Definition. The **complex vector space** $\mathbb{C}^n$ [KN, p.461] is the set of vectors of length n

$$\boldsymbol{v} = \begin{bmatrix} v_1 \\ \vdots \\ v_n \end{bmatrix}$$

with complex entries $v_1, \ldots, v_n \in \mathbb{C}$. The **conjugate** of a vector $\boldsymbol{v} \in \mathbb{C}^n$ is given by the conjugate of each entry

$$\overline{\boldsymbol{v}} = \begin{bmatrix} \overline{v}_1 \\ \vdots \\ \overline{v}_n \end{bmatrix}$$

Definition. The **standard inner product** [KN, p.462] of vectors $\boldsymbol{u}, \boldsymbol{v} \in \mathbb{C}^n$ is

$$\langle \boldsymbol{u}, \boldsymbol{v} \rangle = \boldsymbol{u}^T \overline{\boldsymbol{v}} = u_1\overline{v}_1 + \cdots + u_n\overline{v}_n$$

Proposition. Let $\boldsymbol{u}, \boldsymbol{v} \in \mathbb{C}^n$ and let $c \in \mathbb{C}$.

1. $\langle c\,\boldsymbol{u}, \boldsymbol{v} \rangle = c\,\langle \boldsymbol{u}, \boldsymbol{v} \rangle$
2. $\langle \boldsymbol{u}, c\,\boldsymbol{v} \rangle = \overline{c}\,\langle \boldsymbol{u}, \boldsymbol{v} \rangle$
3. $\langle \boldsymbol{u}, \boldsymbol{v} \rangle = \overline{\langle \boldsymbol{v}, \boldsymbol{u} \rangle}$
4. $\langle \boldsymbol{v}, \boldsymbol{v} \rangle \geq 0$ for all $\boldsymbol{v}$, and $\langle \boldsymbol{v}, \boldsymbol{v} \rangle = 0$ if and only if $\boldsymbol{v} = \boldsymbol{0}$ is the zero vector.

See [KN, p.462].

Definition. The **norm** [KN, p.463] of $\boldsymbol{v} \in \mathbb{C}^n$ is

$$\|\boldsymbol{v}\| = \sqrt{\langle \boldsymbol{v}, \boldsymbol{v} \rangle} = \sqrt{|v_1|^2 + \cdots + |v_n|^2}$$

Definition. Complex vectors $\boldsymbol{u}, \boldsymbol{v} \in \mathbb{C}^n$ are **orthogonal** [KN, p.466] if $\langle \boldsymbol{u}, \boldsymbol{v} \rangle = 0$.

Definition. A complex matrix A is **hermitian** [KN, p.464] if $A = \overline{A}^T$.

Proposition. If A is hermitian then $\langle A\boldsymbol{u}, \boldsymbol{v} \rangle = \langle \boldsymbol{u}, A\boldsymbol{v} \rangle$ for all $\boldsymbol{u}, \boldsymbol{v} \in \mathbb{C}^n$.

Proof. See [KN, p.464]. □

Definition. A complex matrix A is **unitary** [KN, p.466] if $A^{-1} = \overline{A}^T$.

Proposition. If A is unitary then $\langle A\boldsymbol{u}, A\boldsymbol{v} \rangle = \langle \boldsymbol{u}, \boldsymbol{v} \rangle$ for all $\boldsymbol{u}, \boldsymbol{v} \in \mathbb{C}^n$.

Proof. See [KN, p.466]. □

4.2 Discrete Fourier Transform

Big Idea. The discrete Fourier transform (DFT) is the orthogonal projection onto the Fourier basis vectors $\boldsymbol{f}_0, \ldots, \boldsymbol{f}_{N-1}$.

Definition. An **Nth root of unity** is a complex number ω such that $\omega^N = 1$.

Proposition. Let $\omega_N = e^{2\pi i/N}$. Then ω_N is an Nth root of unity and $1, \omega_N, \omega_N^2, \ldots, \omega_N^{N-1}$ are all the Nth roots of unity.

Proposition. Let $\omega_N = e^{2\pi i/N}$.

1. $\overline{\omega}_N = \omega_N^{-1} = \omega_N^{N-1}$
2. $\omega_N^k = \cos\left(\frac{2\pi k}{N}\right) + i \sin\left(\frac{2\pi k}{N}\right)$

Proposition. Let $\omega_N = e^{2\pi i/N}$ and let k be an integer such that $0 < k < N$. Then

$$\sum_{n=0}^{N-1} \omega_N^{nk} = 0$$

Proof. The sum is a geometric series

$$\sum_{n=0}^{N-1} r^n = \frac{1 - r^N}{1 - r}$$

and so with $r = \omega_N^k$ we have

$$\sum_{n=0}^{N-1} \omega_N^{nk} = \frac{1 - \omega_N^{kN}}{1 - \omega_N^k} = 0$$

since $\omega_N^{kN} = 1$ and $\omega_N^k \neq 1$. □

Definition. The **standard basis** of $\mathbb{C}^N$ is $\boldsymbol{e}_0, \ldots, \boldsymbol{e}_{N-1}$ where $\boldsymbol{e}_k$ is the vector with all 0s except 1 in index k

$$\boldsymbol{e}_k = \begin{bmatrix} \mathbf{0} \\ 1 \\ \mathbf{0} \end{bmatrix} \leftarrow \text{index } k$$

Use 0-indexing (as in Python) such that the first entry is at index 0. For example, for $N = 3$,

$$\boldsymbol{e}_0 = \begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix} \qquad \boldsymbol{e}_1 = \begin{bmatrix} 0 \\ 1 \\ 0 \end{bmatrix} \qquad \boldsymbol{e}_2 = \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix}$$

Definition. Let N be a positive integer and let $\omega_N = e^{2\pi i/N}$. The **Fourier basis** of $\mathbb{C}^N$ is $\boldsymbol{f}_0, \ldots, \boldsymbol{f}_{N-1}$ where

$$\boldsymbol{f}_k = \begin{bmatrix} 1 \\ \omega_N^k \\ \omega_N^{2k} \\ \vdots \\ \omega_N^{(N-1)k} \end{bmatrix}$$

Example. For $N = 2$, $\omega_2 = -1$ and the Fourier basis of $\mathbb{C}^2$ is

$$\boldsymbol{f}_0 = \begin{bmatrix} 1 \\ 1 \end{bmatrix} \qquad \boldsymbol{f}_1 = \begin{bmatrix} 1 \\ -1 \end{bmatrix}$$

For $N = 3$, $\omega_3 = e^{2\pi i/3} = (-1 + \sqrt{3}i)/2$ and the Fourier basis of $\mathbb{C}^3$ is

$$\boldsymbol{f}_0 = \begin{bmatrix} 1 \\ 1 \\ 1 \end{bmatrix} \qquad \boldsymbol{f}_1 = \begin{bmatrix} 1 \\ (-1+\sqrt{3}i)/2 \\ (-1-\sqrt{3}i)/2 \end{bmatrix} \qquad \boldsymbol{f}_2 = \begin{bmatrix} 1 \\ (-1-\sqrt{3}i)/2 \\ (-1+\sqrt{3}i)/2 \end{bmatrix}$$

For $N = 4$, $\omega_4 = i$ and the Fourier basis of $\mathbb{C}^4$ is

$$\boldsymbol{f}_0 = \begin{bmatrix} 1 \\ 1 \\ 1 \\ 1 \end{bmatrix} \qquad \boldsymbol{f}_1 = \begin{bmatrix} 1 \\ i \\ -1 \\ -i \end{bmatrix} \qquad \boldsymbol{f}_2 = \begin{bmatrix} 1 \\ -1 \\ 1 \\ -1 \end{bmatrix} \qquad \boldsymbol{f}_3 = \begin{bmatrix} 1 \\ -i \\ -1 \\ i \end{bmatrix}$$

Proposition. The Fourier basis $\boldsymbol{f}_0, \ldots, \boldsymbol{f}_{N-1}$ satisfies

$$\langle \boldsymbol{f}_k, \boldsymbol{f}_\ell \rangle = \begin{cases} N & \text{if } k = \ell \\ 0 & \text{otherwise} \end{cases}$$

Therefore the Fourier basis is an orthogonal basis of $\mathbb{C}^N$.

Proof. Compute

$$\langle \boldsymbol{f}_k, \boldsymbol{f}_\ell \rangle = \sum_{n=0}^{N-1} \omega_N^{nk} \omega_N^{-n\ell} = \sum_{n=0}^{N-1} \omega_N^{n(k-\ell)}$$

We showed in a previous proposition that the sum is equal to 0 if $k \neq \ell$. If $k = \ell$ then clearly the sum is equal to N. □

Proposition. Let $0 < k < N$. Then

$$\overline{\boldsymbol{f}}_k = \boldsymbol{f}_{N-k}$$

Proof. By definition and using $\omega_N^N = 1$ we have

$$\overline{\boldsymbol{f}}_k = \begin{bmatrix} 1 \\ \overline{\omega}_N^k \\ \overline{\omega}_N^{2k} \\ \vdots \\ \overline{\omega}_N^{(N-1)k} \end{bmatrix} = \begin{bmatrix} 1 \\ \omega_N^{-k} \\ \omega_N^{-2k} \\ \vdots \\ \omega_N^{-(N-1)k} \end{bmatrix} = \begin{bmatrix} 1 \\ \omega_N^{N-k} \\ \omega_N^{2(N-k)} \\ \vdots \\ \omega_N^{(N-1)(N-k)} \end{bmatrix} = \boldsymbol{f}_{N-k}$$

□

Definition. Let $\boldsymbol{x} \in \mathbb{C}^N$. The **discrete Fourier transform** of $\boldsymbol{x}$ is

$$\mathrm{DFT}(\boldsymbol{x}) = F_N \boldsymbol{x}$$

where F_N is the **Fourier matrix**

$$F_N = \begin{bmatrix} \overline{\boldsymbol{f}}_0^T \\ \overline{\boldsymbol{f}}_1^T \\ \vdots \\ \overline{\boldsymbol{f}}_{N-1}^T \end{bmatrix} = \begin{bmatrix} 1 & 1 & 1 & \cdots & 1 \\ 1 & \overline{\omega}_N & \overline{\omega}_N^2 & \cdots & \overline{\omega}_N^{N-1} \\ 1 & \overline{\omega}_N^2 & \overline{\omega}_N^4 & \cdots & \overline{\omega}_N^{2(N-1)} \\ 1 & \vdots & \vdots & \ddots & \vdots \\ 1 & \overline{\omega}_N^{N-1} & \overline{\omega}_N^{2(N-1)} & \cdots & \overline{\omega}_N^{(N-1)^2} \end{bmatrix}$$

Note. Expand $\boldsymbol{x}$ in terms of the Fourier basis

$$\boldsymbol{x} = \frac{\langle \boldsymbol{x}, \boldsymbol{f}_0 \rangle}{\langle \boldsymbol{f}_0, \boldsymbol{f}_0 \rangle} \boldsymbol{f}_0 + \cdots + \frac{\langle \boldsymbol{x}, \boldsymbol{f}_{N-1} \rangle}{\langle \boldsymbol{f}_{N-1}, \boldsymbol{f}_{N-1} \rangle} \boldsymbol{f}_{N-1}$$

Note that $\langle \boldsymbol{f}_k, \boldsymbol{f}_k \rangle = N$ for each $k = 0, \ldots, N-1$ and write as matrix multiplication

$$\boldsymbol{x} = \frac{1}{N} \begin{bmatrix} \boldsymbol{f}_0 & \cdots & \boldsymbol{f}_{N-1} \end{bmatrix} \begin{bmatrix} \langle \boldsymbol{x}, \boldsymbol{f}_0 \rangle \\ \langle \boldsymbol{x}, \boldsymbol{f}_1 \rangle \\ \vdots \\ \langle \boldsymbol{x}, \boldsymbol{f}_{N-1} \rangle \end{bmatrix} = \frac{1}{N} \begin{bmatrix} \boldsymbol{f}_0 & \cdots & \boldsymbol{f}_{N-1} \end{bmatrix} \begin{bmatrix} \overline{\boldsymbol{f}}_0^T \\ \overline{\boldsymbol{f}}_1^T \\ \vdots \\ \overline{\boldsymbol{f}}_{N-1}^T \end{bmatrix} \boldsymbol{x}$$

Therefore $F_N \boldsymbol{x}$ is the vector of coefficients of $\boldsymbol{x}$ with respect to the Fourier basis (up to multiplication by N).

Definition. The DFT is used to study sound, images and any kind of information that can be represented by a vector $\boldsymbol{x} \in \mathbb{C}^N$. Therefore, in the context of the DFT, we use the term **signal** to refer to a (column) vector $\boldsymbol{x} \in \mathbb{C}^N$ and we use the notation

$$\boldsymbol{x} = \begin{bmatrix} x_0 & x_1 & x_2 & \cdots & x_{N-1} \end{bmatrix}^T \qquad \boldsymbol{x}[n] = x_n$$

Proposition. Let $\boldsymbol{x}$ be a real signal (that is, $\boldsymbol{x}[k] \in \mathbb{R}$ for each $k = 0, \ldots, N-1$) and let $\boldsymbol{y} = \mathrm{DFT}(\boldsymbol{x})$. Then

$$\overline{\boldsymbol{y}[k]} = \boldsymbol{y}[N-k]$$

Proof. Compute from the definition

$$\begin{aligned} \overline{\boldsymbol{y}[k]} = \overline{\langle \boldsymbol{x}, \boldsymbol{f}_k \rangle} = \langle \boldsymbol{f}_k, \boldsymbol{x} \rangle &= \sum_{n=0}^{N-1} \omega_N^{nk} \overline{x}_n \\ &= \sum_{n=0}^{N-1} \omega_N^{nk-nN} x_n \\ &= \sum_{n=0}^{N-1} \omega_N^{-(N-k)n} x_n = \boldsymbol{y}[N-k] \end{aligned}$$

□

Proposition. For each $k = 0, \ldots, N-1$, we have

$$\mathrm{DFT}(\boldsymbol{f}_k) = N\boldsymbol{e}_k$$

where $\boldsymbol{e}_k$ is the kth standard basis vector.

Proof. By definition of DFT and the fact $\langle \boldsymbol{f}_k, \boldsymbol{f}_\ell \rangle = 0$ when $k \neq \ell$ and $\langle \boldsymbol{f}_k, \boldsymbol{f}_k \rangle = N$, compute

$$\mathrm{DFT}(\boldsymbol{f}_k) = \begin{bmatrix} \overline{\boldsymbol{f}}_0^T \\ \vdots \\ \overline{\boldsymbol{f}}_{N-1}^T \end{bmatrix} \boldsymbol{f}_k = \begin{bmatrix} \langle \boldsymbol{f}_k, \boldsymbol{f}_0 \rangle \\ \vdots \\ \langle \boldsymbol{f}_k, \boldsymbol{f}_{N-1} \rangle \end{bmatrix} = \begin{bmatrix} \boldsymbol{0} \\ N \\ \boldsymbol{0} \end{bmatrix} \leftarrow \text{index } k$$

and so $\mathrm{DFT}(\boldsymbol{f}_k) = N\boldsymbol{e}_k$. □

Definition. Let $\boldsymbol{y} \in \mathbb{C}^N$. The **inverse discrete Fourier transform** of $\boldsymbol{y}$ is

$$\mathrm{IDFT}(\boldsymbol{y}) = \frac{1}{N} \overline{F}_N^T \boldsymbol{y}$$

Note. The Fourier matrix F_N is *not* unitary however the matrix $\frac{1}{\sqrt{N}}F_N$ is unitary.

4.3 Frequency, Amplitude and Phase

Big Idea. The DFT of a signal computes the amplitude and phase of each frequency in the signal.

Definition. The DFT is used to study sound, images and any kind of information that can be represented by a vector $\boldsymbol{x} \in \mathbf{C}^N$. Therefore, in the context of the DFT, we use the term **signal** to refer to a vector $\boldsymbol{x} \in \mathbf{C}^N$ and we use the notation $\boldsymbol{x}[n] = x_n$ to refer to the entries

$$\boldsymbol{x} = \begin{bmatrix} x_0 & x_1 & x_2 & \cdots & x_{N-1} \end{bmatrix}^T$$

Definition. Let N be a positive integer and let

$$\boldsymbol{n} = \begin{bmatrix} 0 & 1 & 2 & \cdots & N-1 \end{bmatrix}^T \qquad \boldsymbol{t} = (1/N)\boldsymbol{n} = \begin{bmatrix} 0 & 1/N & 2/N & \cdots & (N-1)/N \end{bmatrix}^T$$

A **sinusoid** is a signal of the form

$$\boldsymbol{x} = A\cos(2\pi k\boldsymbol{t} + \phi)$$

where k is the **frequency** (in periods per N samples), A is the **amplitude** and ϕ is the **phase**. Here we use vector notation

$$A\cos(2\pi k\boldsymbol{t} + \phi) = \begin{bmatrix} A\cos(\phi) \\ A\cos(2\pi k(1/N) + \phi) \\ A\cos(2\pi k(2/N) + \phi) \\ \vdots \\ A\cos(2\pi k(N-1)/N + \phi) \end{bmatrix}$$

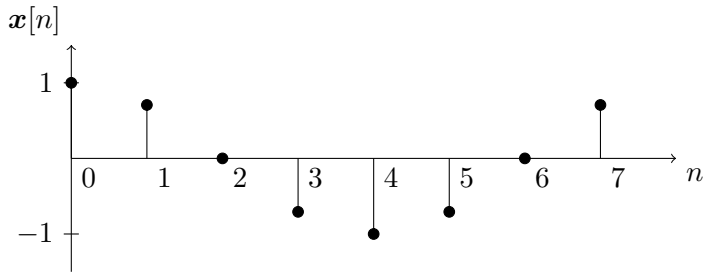
Example. Let $N = 8$ and so

$$\boldsymbol{n} = \begin{bmatrix} 0 & 1 & 2 & 3 & 4 & 5 & 6 & 7 \end{bmatrix}^T$$
$$\boldsymbol{t} = \begin{bmatrix} 0 & 1/8 & 1/4 & 3/8 & 1/2 & 5/8 & 3/4 & 7/8 \end{bmatrix}^T$$

Consider the signal

$$\boldsymbol{x} = \cos(2\pi\boldsymbol{t}) = \begin{bmatrix} 1 & 1/\sqrt{2} & 0 & -1/\sqrt{2} & -1 & -1/\sqrt{2} & 0 & 1/\sqrt{2} \end{bmatrix}^T$$

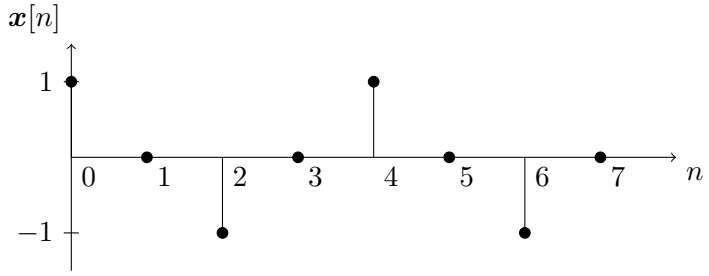
and sketch the signal as a stemplot



Now consider the signal

$$\boldsymbol{x} = \cos(4\pi \boldsymbol{t}) = \begin{bmatrix} 1 & 0 & -1 & 0 & 1 & 0 & -1 \end{bmatrix}^T$$

and sketch the signal as a stemplot



Proposition.

$$\begin{aligned} \boldsymbol{f}_k &= \cos(2\pi k \boldsymbol{t}) + i \sin(2\pi k \boldsymbol{t}) \\ \frac{1}{2}\left(\boldsymbol{f}_k + \overline{\boldsymbol{f}}_k\right) &= \cos(2\pi k \boldsymbol{t}) \\ \frac{1}{2i}\left(\boldsymbol{f}_k - \overline{\boldsymbol{f}}_k\right) &= \sin(2\pi k \boldsymbol{t}) \end{aligned}$$

Proof. We showed in a previous proposition that

$$\omega_N^k = \cos\left(\frac{2\pi k}{N}\right) + i \sin\left(\frac{2\pi k}{N}\right)$$

therefore

$$\boldsymbol{f}_k = \begin{bmatrix} 1 \\ \omega_N^k \\ \omega_N^{2k} \\ \vdots \\ \omega_N^{(N-1)k} \end{bmatrix} = \begin{bmatrix} 1 \\ \cos(2\pi k(1/N)) + i \sin(2\pi k(1/N)) \\ \cos(2\pi k(2/N)) + i \sin(2\pi k(2/N)) \\ \vdots \\ \cos(2\pi k(N-1)/N) + i \sin(2\pi k(N-1)/N) \end{bmatrix} = \cos(2\pi k \boldsymbol{t}) + i \sin(2\pi k \boldsymbol{t})$$

Further,

$$\begin{aligned}\omega_N^{nk} + \omega_N^{-nk} &= \cos\left(\frac{2\pi nk}{N}\right) + i\sin\left(\frac{2\pi nk}{N}\right) + \cos\left(\frac{2\pi nk}{N}\right) - i\sin\left(\frac{2\pi nk}{N}\right) \\ &= 2\cos\left(\frac{2\pi nk}{N}\right)\end{aligned}$$

Therefore

$$\boldsymbol{f}_k + \overline{\boldsymbol{f}}_k = \begin{bmatrix} 1 \\ \omega_N^k \\ \omega_N^{2k} \\ \vdots \\ \omega_N^{(N-1)k} \end{bmatrix} + \begin{bmatrix} 1 \\ \omega_N^{-k} \\ \omega_N^{-2k} \\ \vdots \\ \omega_N^{-(N-1)k} \end{bmatrix} = \begin{bmatrix} 2 \\ 2\cos(2\pi k/N) \\ 2\cos(2\pi n(2k)/N) \\ \vdots \\ 2\cos(2\pi(N-1)k/N) \end{bmatrix} = 2\cos(2\pi k\boldsymbol{t})$$

The last equality is proved similarly. □

Proposition. Let $\boldsymbol{x} = A\cos(2\pi k\boldsymbol{t} + \phi)$. Then

$$\mathrm{DFT}(\boldsymbol{x}) = \frac{AN}{2}e^{i\phi}\,\boldsymbol{e}_k + \frac{AN}{2}e^{-i\phi}\,\boldsymbol{e}_{N-k}$$

Proof. We proved in a previous proposition

$$\cos(2\pi k\boldsymbol{t}) = \frac{1}{2}\left(\boldsymbol{f}_k + \overline{\boldsymbol{f}}_k\right) = \frac{1}{2}\left(\boldsymbol{f}_k + \boldsymbol{f}_{N-k}\right)$$

and we also showed that

$$\mathrm{DFT}(\boldsymbol{f}_k) = N\boldsymbol{e}_k$$

Compute

$$\mathrm{DFT}(\cos(2\pi k\boldsymbol{t})) = \frac{1}{2}\mathrm{DFT}(\left(\boldsymbol{f}_k + \boldsymbol{f}_{N-k}\right)) = \frac{1}{2}\left(N\boldsymbol{e}_k + N\boldsymbol{e}_{N-k}\right)$$

Similarly, compute

$$\mathrm{DFT}(\sin(2\pi k\boldsymbol{t})) = \frac{1}{2i}\mathrm{DFT}(\left(\boldsymbol{f}_k - \boldsymbol{f}_{N-k}\right)) = \frac{1}{2i}\left(N\boldsymbol{e}_k - N\boldsymbol{e}_{N-k}\right)$$

Use the trigonometric identity

$$\cos(\alpha + \beta) = \cos\alpha\cos\beta - \sin\alpha\sin\beta$$

to find

$$\begin{aligned}\mathrm{DFT}(\boldsymbol{x}) &= \mathrm{DFT}(A\cos(2\pi k\boldsymbol{t} + \phi)) \\ &= A\cos(\phi)\,\mathrm{DFT}(\cos(2\pi k\boldsymbol{t})) - A\sin(\phi)\,\mathrm{DFT}(\sin(2\pi k\boldsymbol{t})) \\ &= \frac{A\cos(\phi)}{2}\left(N\boldsymbol{e}_k + N\boldsymbol{e}_{N-k}\right) - \frac{A\sin(\phi)}{2i}\left(N\boldsymbol{e}_k - N\boldsymbol{e}_{N-k}\right)\end{aligned}$$

$$= \frac{AN(\cos(\phi) + i\sin(\phi))}{2}\,\boldsymbol{e}_k + \frac{AN(\cos(\phi) - i\sin(\phi))}{2}\,\boldsymbol{e}_{N-k}$$
$$= \frac{AN}{2}e^{i\phi}\,\boldsymbol{e}_k + \frac{AN}{2}e^{-i\phi}\,\boldsymbol{e}_{N-k}$$

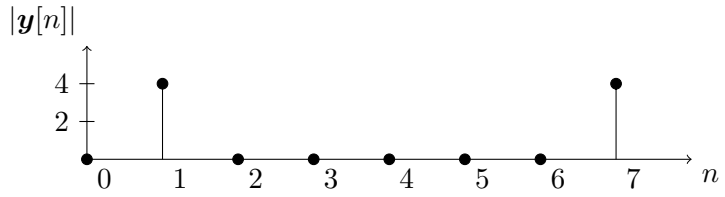
□

Definition. The **magnitude stemplot** of a complex vector $\boldsymbol{y} \in \mathbb{C}^N$ is the plot of the magnitude $|\boldsymbol{y}[n]|$ versus the index n. The **angle stemplot** of $\boldsymbol{y}$ is the plot of the argument $\angle\boldsymbol{y}[n]$ versus the index n.

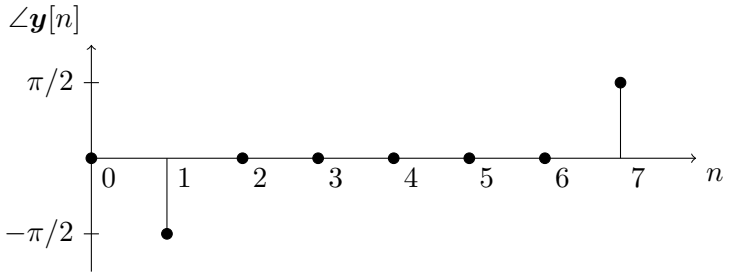
Example. Let $N = 8$ and compute the DFT of the sinusoid $\boldsymbol{x} = \sin(2\pi\boldsymbol{t})$. Since $\boldsymbol{x} = \cos(2\pi\boldsymbol{t} - \pi/2)$ we have $A = 1$, $k = 1$ and $\phi = -\pi/2$, and so

$$\boldsymbol{y} = \text{DFT}(\boldsymbol{x}) = -4i\boldsymbol{e}_1 + 4i\boldsymbol{e}_7 = \begin{bmatrix} 0 & -4i & 0 & 0 & 0 & 0 & 0 & 4i \end{bmatrix}^T$$

The magnitude stemplot of $\boldsymbol{y} = \text{DFT}(\boldsymbol{x})$ is given by



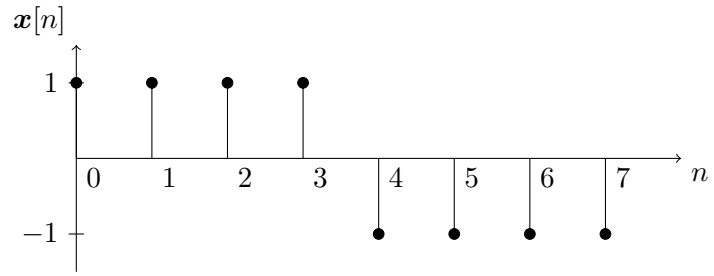
The angle stemplot of $\boldsymbol{y} = \text{DFT}(\boldsymbol{x})$ is given by



Example. Let $N = 8$ and let

$$\boldsymbol{x} = \begin{bmatrix} 1 & 1 & 1 & 1 & -1 & -1 & -1 & -1 \end{bmatrix}^T$$

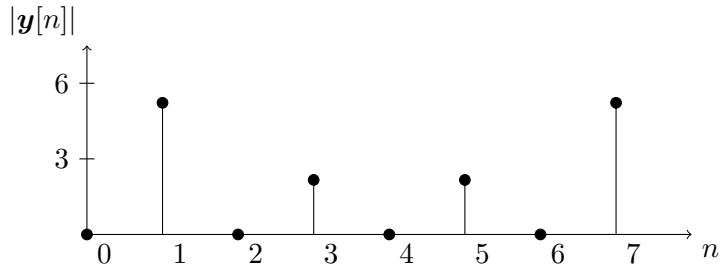
Sketch the signal



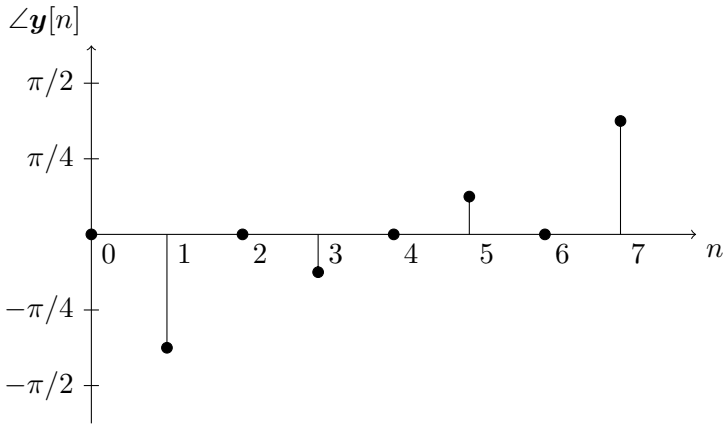
What frequencies occur in this signal? Compute the DFT

$$\boldsymbol{y} = \text{DFT}(\boldsymbol{x}) = \begin{bmatrix} 0 & 2-2(\sqrt{2}+1)i & 0 & 2-2(\sqrt{2}-1)i & 0 & 2+2(\sqrt{2}-1)i & 2+2(\sqrt{2}+1)i \end{bmatrix}^T$$

The magnitude stemplot of $\boldsymbol{y}$ is given by



The angle stemplot of $\boldsymbol{y}$ is given by



Therefore we may rewrite the signal $\boldsymbol{x}$ as a sum of sinusoids

$$\boldsymbol{x} = A_1 \cos(2\pi \boldsymbol{t} + \phi_1) + A_3 \cos(6\pi \boldsymbol{t} + \phi_3)$$

where

$$A_1 = 2\sqrt{4+2\sqrt{2}} \quad \phi_1 = -\frac{3\pi}{8} \quad A_3 = 2\sqrt{4-2\sqrt{2}} \quad \phi_3 = -\frac{\pi}{8}$$

4.4 Fast Fourier Transform

Big Idea. The fast Fourier transform (FFT) is an algorithm for efficiently computing the DFT.

Note. Sound signals are commonly sampled at 44.1 kHz (see Wikipedia: Audio sampling). Therefore computing the DFT for a one second sound signal requires the Fourier matrix F_N for $N = 44100$ which has $44100^2 \approx 2$ billion entries. Yikes! We need an efficient algorithm to compute the DFT in practice.

Theorem. Let $\boldsymbol{x} = \begin{bmatrix} x_0 & x_1 & \cdots & x_{N-1} \end{bmatrix}^T$ be a signal of length N (and assume N is even). Then

$$\text{DFT}(\boldsymbol{x}) = \begin{bmatrix} \text{DFT}(\boldsymbol{x}_{\text{even}}) + D_N \text{DFT}(\boldsymbol{x}_{\text{odd}}) \\ \text{DFT}(\boldsymbol{x}_{\text{even}}) - D_N \text{DFT}(\boldsymbol{x}_{\text{odd}}) \end{bmatrix} = \begin{bmatrix} I & D_N \\ I & -D_N \end{bmatrix} \begin{bmatrix} \text{DFT}(\boldsymbol{x}_{\text{even}}) \\ \text{DFT}(\boldsymbol{x}_{\text{odd}}) \end{bmatrix}$$

where $\boldsymbol{x}_{\text{even}}$ and $\boldsymbol{x}_{\text{odd}}$ are vectors of length $N/2$ consisting of the even and odd indices respectively

$$\boldsymbol{x}_{\text{even}} = \begin{bmatrix} x_0 \\ x_2 \\ \vdots \\ x_{N-2} \end{bmatrix} \qquad \boldsymbol{x}_{\text{odd}} = \begin{bmatrix} x_1 \\ x_3 \\ \vdots \\ x_{N-1} \end{bmatrix}$$

and

$$D_N = \begin{bmatrix} 1 & & & \\ & \omega_N^{-1} & & \\ & & \ddots & \\ & & & \omega_N^{-(N/2-1)} \end{bmatrix}$$

Proof. Let $\boldsymbol{y} = \text{DFT}(\boldsymbol{x})$, look at the kth entry of $\boldsymbol{y}$ and split the sum into the even and odd terms

$$\begin{aligned} \boldsymbol{y}[k] = \langle \boldsymbol{x}, \boldsymbol{f}_k \rangle &= \sum_{n=0}^{N-1} x_n \omega_N^{-nk} \\ &= \sum_{m=0}^{N/2-1} x_{2m} \omega_N^{-2mk} + \sum_{m=0}^{N/2-1} x_{2m+1} \omega_N^{-(2m+1)k} \end{aligned}$$

See that $\omega_N^2 = e^{(2\pi i/N)2} = e^{2\pi i/(N/2)} = \omega_{N/2}$ and write

$$\boldsymbol{y}[k] = \sum_{m=0}^{N/2-1} x_{2m} \omega_{N/2}^{-mk} + \omega_N^{-k} \sum_{m=0}^{N/2-1} x_{2m+1} \omega_{N/2}^{-mk}$$

For $0 \leq k < N/2$, these are the formulas for DFT of vectors of length $N/2$ consisting of the

even and odd indices respectively

$$\boldsymbol{x}_{\text{even}} = \begin{bmatrix} x_0 \\ x_2 \\ \vdots \\ x_{N-2} \end{bmatrix} \qquad \boldsymbol{x}_{\text{odd}} = \begin{bmatrix} x_1 \\ x_3 \\ \vdots \\ x_{N-1} \end{bmatrix}$$

Note that

$$\begin{aligned} \boldsymbol{y}[k+N/2] &= \sum_{m=0}^{N/2-1} x_{2m}\omega_{N/2}^{-m(k+N/2)} + \omega_N^{-(k+N/2)} \sum_{m=0}^{N/2-1} x_{2m+1}\omega_{N/2}^{-m(k+N/2)} \\ &= \sum_{m=0}^{N/2-1} x_{2m}\omega_{N/2}^{-mk} \underbrace{\omega_{N/2}^{-mN/2}}_{1} + \omega_N^{-k} \underbrace{\omega_N^{-N/2}}_{-1} \sum_{m=0}^{N/2-1} x_{2m+1}\omega_{N/2}^{-mk} \underbrace{\omega_{N/2}^{-mN/2}}_{1} \\ &= \sum_{m=0}^{N/2-1} x_{2m}\omega_{N/2}^{-mk} - \omega_N^{-k} \sum_{m=0}^{N/2-1} x_{2m+1}\omega_{N/2}^{-mk} \end{aligned}$$

which again are the DFTs of the even and odd parts of $\boldsymbol{x}$. Put these formulas for $\boldsymbol{y}$ together to get

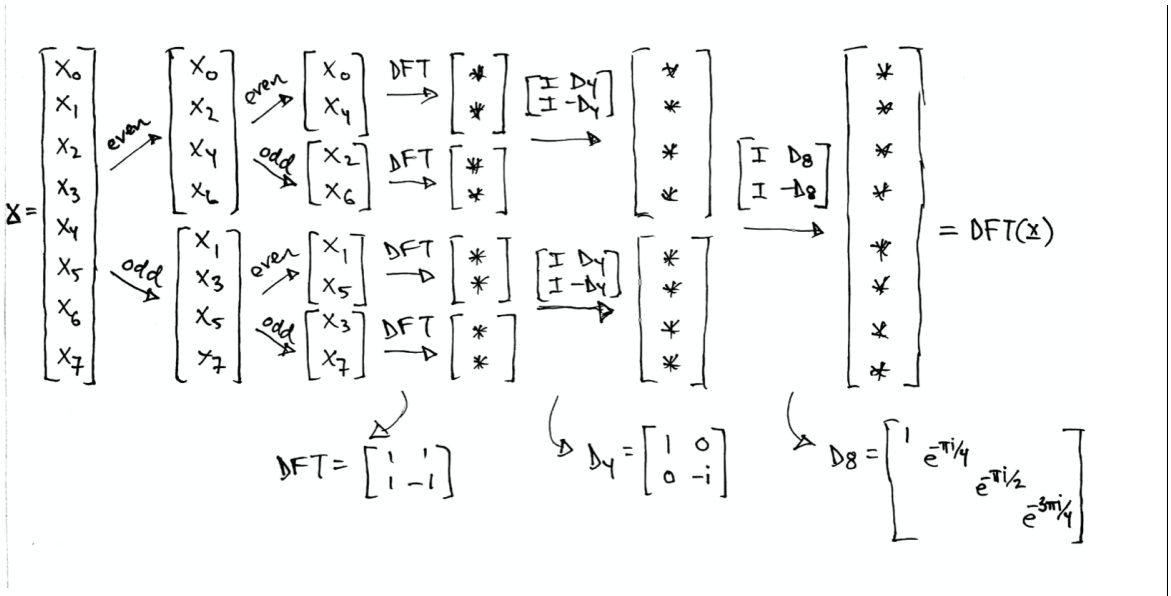
$$\boldsymbol{y} = \text{DFT}(\boldsymbol{x}) = \begin{bmatrix} \text{DFT}(\boldsymbol{x}_{\text{even}}) + D_N \text{DFT}(\boldsymbol{x}_{\text{odd}}) \\ \text{DFT}(\boldsymbol{x}_{\text{even}}) - D_N \text{DFT}(\boldsymbol{x}_{\text{odd}}) \end{bmatrix}$$

where

$$D_N = \begin{bmatrix} 1 & & & \\ & \omega_N^{-1} & & \\ & & \ddots & \\ & & & \omega_N^{-(N/2-1)} \end{bmatrix}$$

□

Note. This form of the fast Fourier transform is called the Cooley-Tukey algorithm. The point is that the DFT computation for a vector of length N can be compute by the DFT of two smaller vectors of length $N/2$ which is faster! And we can keep applying the formulas for smaller and smaller vectors until we are computing the DFT for vectors of size 2 only (if N is a power of 2). For example, to compute $\text{DFT}(\boldsymbol{x})$ for $\boldsymbol{x} \in \mathbb{C}^8$ we visualize the procedure



Example. Use the FFT to compute the DFT of the signal

$$\boldsymbol{x} = \begin{bmatrix} 1 & 1 & 1 & 1 & -1 & -1 & -1 & -1 \end{bmatrix}^T$$

The symmetry in the signal reduces the number of computations even further

$$\begin{bmatrix} x_0 \\ x_4 \end{bmatrix} = \begin{bmatrix} 1 \\ -1 \end{bmatrix} \xrightarrow{\text{DFT}} \begin{bmatrix} 0 \\ 2 \end{bmatrix}$$
$$\begin{bmatrix} x_2 \\ x_6 \end{bmatrix} = \begin{bmatrix} 1 \\ -1 \end{bmatrix} \xrightarrow{\text{DFT}} \begin{bmatrix} 0 \\ 2 \end{bmatrix}$$
$$\longrightarrow \begin{bmatrix} \begin{bmatrix} 0 \\ 2 \end{bmatrix} + \begin{bmatrix} 1 & \\ & -i \end{bmatrix} \begin{bmatrix} 0 \\ 2 \end{bmatrix} \\ \begin{bmatrix} 0 \\ 2 \end{bmatrix} - \begin{bmatrix} 1 & \\ & -i \end{bmatrix} \begin{bmatrix} 0 \\ 2 \end{bmatrix} \end{bmatrix} = \begin{bmatrix} 0 \\ 2-2i \\ 0 \\ 2+2i \end{bmatrix}$$

$$\begin{bmatrix} x_1 \\ x_5 \end{bmatrix} = \begin{bmatrix} 1 \\ -1 \end{bmatrix} \xrightarrow{\text{DFT}} \begin{bmatrix} 0 \\ 2 \end{bmatrix}$$
$$\begin{bmatrix} x_3 \\ x_7 \end{bmatrix} = \begin{bmatrix} 1 \\ -1 \end{bmatrix} \xrightarrow{\text{DFT}} \begin{bmatrix} 0 \\ 2 \end{bmatrix}$$
$$\longrightarrow \begin{bmatrix} \begin{bmatrix} 0 \\ 2 \end{bmatrix} + \begin{bmatrix} 1 & \\ & -i \end{bmatrix} \begin{bmatrix} 0 \\ 2 \end{bmatrix} \\ \begin{bmatrix} 0 \\ 2 \end{bmatrix} - \begin{bmatrix} 1 & \\ & -i \end{bmatrix} \begin{bmatrix} 0 \\ 2 \end{bmatrix} \end{bmatrix} = \begin{bmatrix} 0 \\ 2-2i \\ 0 \\ 2+2i \end{bmatrix}$$

We can write the matrix D_8 as

$$D_8 = \begin{bmatrix} 1 & & & \\ & e^{-\pi i/4} & & \\ & & e^{-\pi i/2} & \\ & & & e^{-3\pi i/4} \end{bmatrix} = \begin{bmatrix} 1 & & & \\ & \frac{1-i}{\sqrt{2}} & & \\ & & -i & \\ & & & \frac{-1-i}{\sqrt{2}} \end{bmatrix}$$

and then compute

$$\mathrm{DFT}(\boldsymbol{x}) = \begin{bmatrix} \begin{bmatrix} 0 \\ 2-2i \\ 0 \\ 2+2i \end{bmatrix} + \begin{bmatrix} 1 & & & \\ & \frac{1-i}{\sqrt{2}} & & \\ & & -i & \\ & & & \frac{-1-i}{\sqrt{2}} \end{bmatrix} \begin{bmatrix} 0 \\ 2-2i \\ 0 \\ 2+2i \end{bmatrix} \\ \begin{bmatrix} 0 \\ 2-2i \\ 0 \\ 2+2i \end{bmatrix} - \begin{bmatrix} 1 & & & \\ & \frac{1-i}{\sqrt{2}} & & \\ & & -i & \\ & & & \frac{-1-i}{\sqrt{2}} \end{bmatrix} \begin{bmatrix} 0 \\ 2-2i \\ 0 \\ 2+2i \end{bmatrix} \end{bmatrix} = \begin{bmatrix} 0 \\ 2-2\left(\sqrt{2}+1\right)i \\ 0 \\ 2-2\left(\sqrt{2}-1\right)i \\ 0 \\ 2+2\left(\sqrt{2}-1\right)i \\ 0 \\ 2+2\left(\sqrt{2}+1\right)i \end{bmatrix}$$

4.5 Convolution Theorem and Filtering

Big Idea. The discrete Fourier transform of the convolution of two signals is equal to the elementwise product of the discrete Fourier transforms of those signals. In other words, convolution in the time domain corresponds via DFT to elementwise multiplication in the frequency domain.

Definition. Let $\boldsymbol{x}, \boldsymbol{y} \in \mathbb{C}^N$. The **convolution** of $\boldsymbol{x}$ and $\boldsymbol{y}$ is the vector $\boldsymbol{x} * \boldsymbol{y} \in \mathbb{C}^N$ given by

$$(\boldsymbol{x} * \boldsymbol{y})[n] = \sum_{m=0}^{N-1} \boldsymbol{x}[m]\boldsymbol{y}[n-m]$$

We interpret $\boldsymbol{y}[n-m]$ using modular arithmetic: if $n-m$ is outside the interval $[0, N-1]$, then we add/subtract multiples of N until the index is inside the interval $[0, N-1]$. See Wikipedia: Convolution.

Example. Let $\boldsymbol{x} = \begin{bmatrix} 1 & 2 & 1 & -1 \end{bmatrix}$ and $\boldsymbol{y} = \begin{bmatrix} 0 & 1/2 & 1/2 & 0 \end{bmatrix}$. Compute

$$\begin{aligned} (\boldsymbol{x} * \boldsymbol{y})[0] = \sum_{m=0}^{3} \boldsymbol{x}[m]\boldsymbol{y}[-m] &= \boldsymbol{x}[0]\boldsymbol{y}[0] + \boldsymbol{x}[1]\boldsymbol{y}[-1] + \boldsymbol{x}[2]\boldsymbol{y}[-2] + \boldsymbol{x}[3]\boldsymbol{y}[-3] \\ &= \boldsymbol{x}[0]\boldsymbol{y}[0] + \boldsymbol{x}[1]\boldsymbol{y}[3] + \boldsymbol{x}[2]\boldsymbol{y}[2] + \boldsymbol{x}[3]\boldsymbol{y}[1] \\ &= (1)(0) + (2)(0) + (1)(1/2) + (-1)(1/2) \\ &= 0 \end{aligned}$$

$$\begin{aligned} (\boldsymbol{x} * \boldsymbol{y})[1] = \sum_{m=0}^{3} \boldsymbol{x}[m]\boldsymbol{y}[1-m] &= \boldsymbol{x}[0]\boldsymbol{y}[1] + \boldsymbol{x}[1]\boldsymbol{y}[0] + \boldsymbol{x}[2]\boldsymbol{y}[-1] + \boldsymbol{x}[3]\boldsymbol{y}[-2] \\ &= \boldsymbol{x}[0]\boldsymbol{y}[1] + \boldsymbol{x}[1]\boldsymbol{y}[0] + \boldsymbol{x}[2]\boldsymbol{y}[3] + \boldsymbol{x}[3]\boldsymbol{y}[2] \end{aligned}$$

$$= (1)(1/2) + (2)(0) + (1)(0) + (-1)(1/2)$$
$$= 0$$

$$(\boldsymbol{x} * \boldsymbol{y})[2] = \sum_{m=0}^{3} \boldsymbol{x}[m]\boldsymbol{y}[2-m] = \boldsymbol{x}[0]\boldsymbol{y}[2] + \boldsymbol{x}[1]\boldsymbol{y}[1] + \boldsymbol{x}[2]\boldsymbol{y}[0] + \boldsymbol{x}[3]\boldsymbol{y}[-1]$$
$$= \boldsymbol{x}[0]\boldsymbol{y}[2] + \boldsymbol{x}[1]\boldsymbol{y}[1] + \boldsymbol{x}[2]\boldsymbol{y}[0] + \boldsymbol{x}[3]\boldsymbol{y}[3]$$
$$= (1)(1/2) + (2)(1/2) + (1)(0) + (-1)(0)$$
$$= 3/2$$

$$(\boldsymbol{x} * \boldsymbol{y})[3] = \sum_{m=0}^{3} \boldsymbol{x}[m]\boldsymbol{y}[3-m] = \boldsymbol{x}[0]\boldsymbol{y}[3] + \boldsymbol{x}[1]\boldsymbol{y}[2] + \boldsymbol{x}[2]\boldsymbol{y}[1] + \boldsymbol{x}[3]\boldsymbol{y}[0]$$
$$= (1)(0) + (2)(1/2) + (1)(1/2) + (-1)(0)$$
$$= 3/2$$

Definition. Let $\boldsymbol{u}, \boldsymbol{v} \in \mathbb{C}^N$. The **elementwise product** (or **Hadamard product**) of $\boldsymbol{u}$ and $\boldsymbol{v}$ is the vector $\boldsymbol{u} \circ \boldsymbol{v} \in \mathbb{C}^N$ given by

$$\boldsymbol{u} \circ \boldsymbol{v} = \begin{bmatrix} u_0 \\ u_1 \\ \vdots \\ u_{N-1} \end{bmatrix} \circ \begin{bmatrix} v_0 \\ v_1 \\ \vdots \\ v_{N-1} \end{bmatrix} = \begin{bmatrix} u_0 v_0 \\ u_1 v_1 \\ \vdots \\ u_{N-1} v_{N-1} \end{bmatrix}$$

See Wikipedia: Hadamard product.

Theorem. Let $\boldsymbol{x}, \boldsymbol{y} \in \mathbb{C}^N$. Then

$$\mathrm{DFT}(\boldsymbol{x} * \boldsymbol{y}) = \mathrm{DFT}(\boldsymbol{x}) \circ \mathrm{DFT}(\boldsymbol{y})$$

where $\circ$ denotes elementwise multiplication. See Wikipedia: Convolution.

Proof. Compute from the definitions

$$(\mathrm{DFT}(\boldsymbol{x} * \boldsymbol{y}))[k] = \sum_{n=0}^{N-1} \sum_{m=0}^{N-1} \boldsymbol{x}[m]\boldsymbol{y}[n-m]\omega_N^{-nk}$$
$$= \sum_{n=0}^{N-1} \sum_{m=0}^{N-1} \boldsymbol{x}[m]\boldsymbol{y}[n-m]\omega_N^{-nk}\omega_N^{mk}\omega_N^{-mk}$$
$$= \sum_{m=0}^{N-1} \boldsymbol{x}[m]\omega_N^{-mk} \sum_{n=0}^{N-1} \boldsymbol{y}[n-m]\omega_N^{-(n-m)k}$$

$$= \sum_{m=0}^{N-1} \boldsymbol{x}[m]\omega_N^{-mk} \sum_{n=0}^{N-1} \boldsymbol{y}[n]\omega_N^{-nk}$$
$$= \mathrm{DFT}(\boldsymbol{x})[k]\ \mathrm{DFT}(\boldsymbol{y})[k]$$

Therefore $\mathrm{DFT}(\boldsymbol{x} * \boldsymbol{y}) = \mathrm{DFT}(\boldsymbol{x}) \cdot \mathrm{DFT}(\boldsymbol{y})$. □

4.6 Exercises

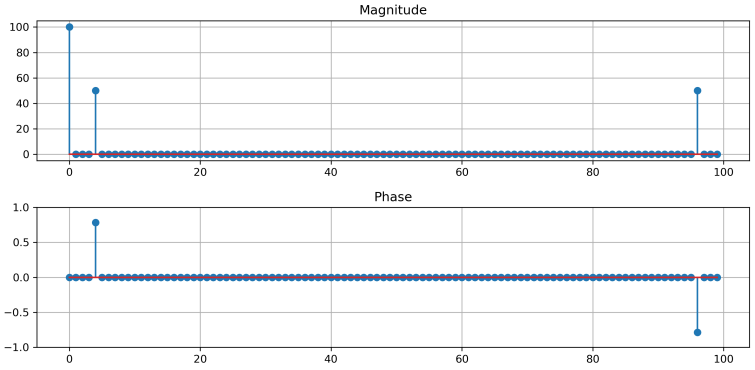
1. Suppose a signal $\boldsymbol{x}$ of length 9 has real values and let $\boldsymbol{y} = \text{DFT}(\boldsymbol{x})$. Determine all the values of $\boldsymbol{y}$ given the values at even indices

$$\boldsymbol{y}[0] = 1 \quad \boldsymbol{y}[2] = 2 + i \quad \boldsymbol{y}[4] = 1 + 2i \quad \boldsymbol{y}[6] = 1 - 3i \quad \boldsymbol{y}[8] = 1 - i$$

2. Find a formula for $\boldsymbol{x}$ as a sum of sinusoids given

$$\text{DFT}(\boldsymbol{x}) = \begin{bmatrix} 1 & 3 - 3i & 2\sqrt{3} + 2i & -4i & 4i & 2\sqrt{3} - 2i & 3 + 3i \end{bmatrix}^T$$

3. Sketch the signal $\boldsymbol{x}$ such that the magnitude and phase plots of $\boldsymbol{y} = \text{DFT}(\boldsymbol{x})$ are



4. Let $\boldsymbol{x} = \begin{bmatrix} 1 & -1 & 2 & 1 \end{bmatrix}^T$. Compute $\text{DFT}(\boldsymbol{x})$ using the fast Fourier transform. Compute $\text{DFT}(\boldsymbol{x})$ also by F_4 and verify it is the same result.

5. Let $\boldsymbol{x} = \begin{bmatrix} 1 & 1 & 0 & 2 & 1 & 2 & 0 & -1 \end{bmatrix}^T$. Compute $\text{DFT}(\boldsymbol{x})$ using the fast Fourier transform.

6. Let N be an even integer and let $\boldsymbol{x} \in \mathbb{R}^N$ such that $\boldsymbol{x}[n] = 1$ if n is even and $\boldsymbol{x}[n] = 0$ if n is odd. Find $\text{DFT}(\boldsymbol{x})$.

7. Let N be an even integer and let $\boldsymbol{x} \in \mathbb{R}^N$ such that $\boldsymbol{x}[n] = 1$ if n is even and $\boldsymbol{x}[n] = -1$ if n is odd. Find $\text{DFT}(\boldsymbol{x})$.

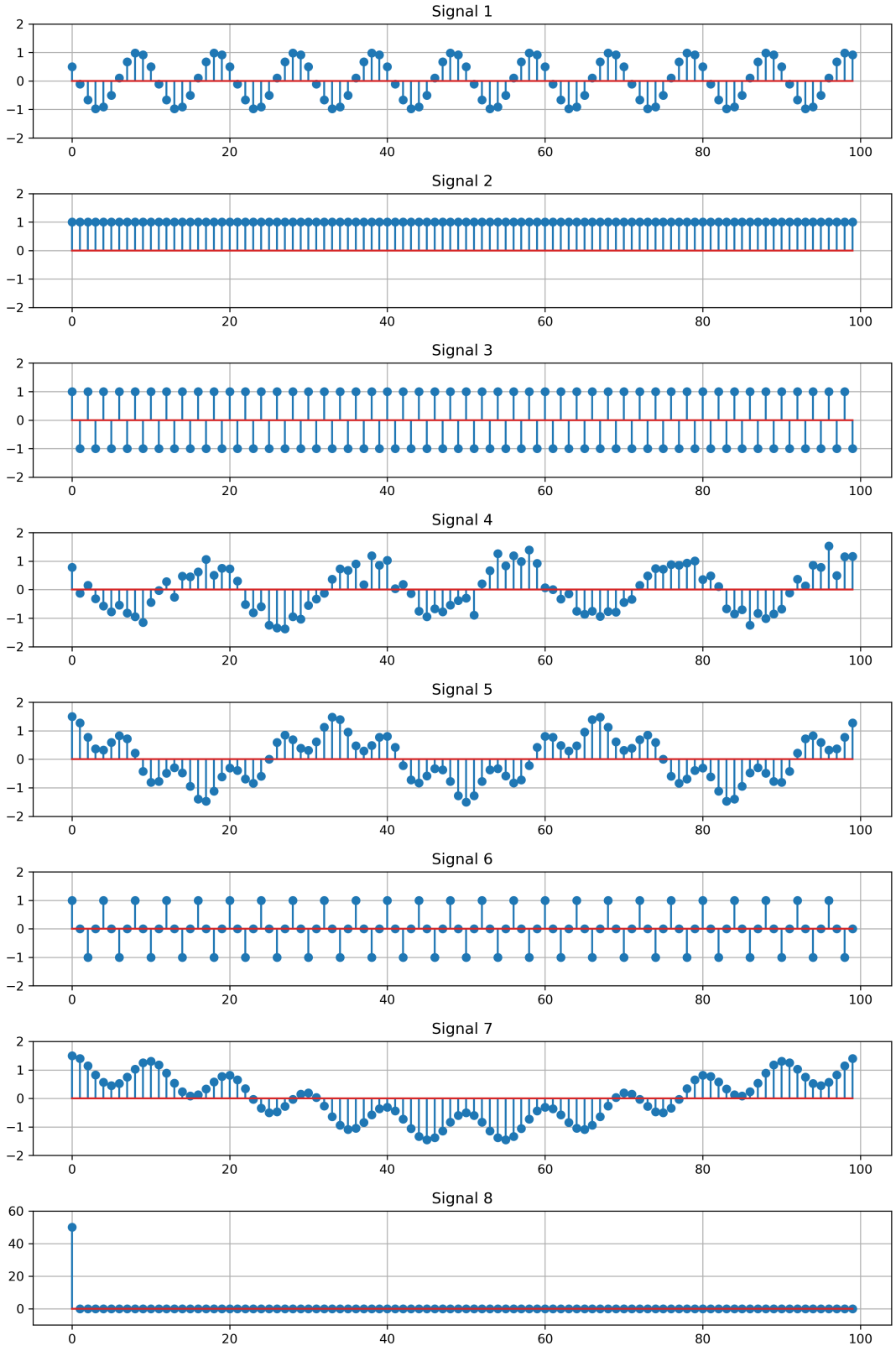
8. Let N be an integer and let $\boldsymbol{x} \in \mathbb{R}^N$ such that $\boldsymbol{x}[0] = 0$ and $\boldsymbol{x}[n] = 1$ for $0 < n < N$. Find $\text{DFT}(\boldsymbol{x})$.

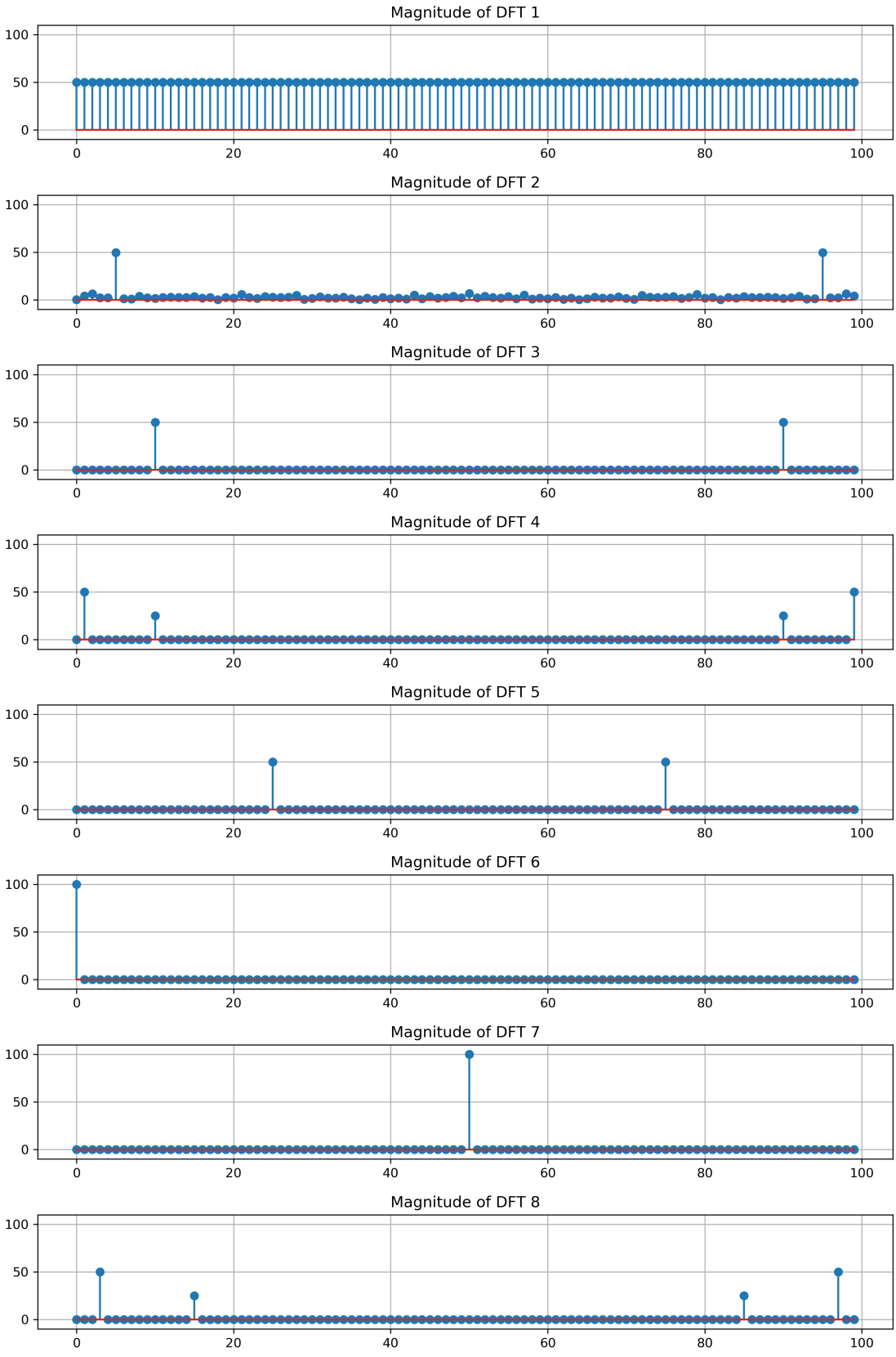
9. Run the following Python code for different values N:

```
N = 100
x = np.random.rand(N)
y = np.fft.fft(x)
plt.stem(np.abs(y),use_line_collection=True)
plt.show()
```

Describe the magnitude plot and explain why it has the same general shape for each random sample. (Recall `np.random.rand` samples from the uniform distribution on $[0, 1]$.)

10. Match the signal with the magnitude plot of its discrete Fourier transform.




Magnitude of DFT 1
Magnitude of DFT 2
Magnitude of DFT 3
Magnitude of DFT 4
Magnitude of DFT 5
Magnitude of DFT 6
Magnitude of DFT 7
Magnitude of DFT 8
0
50
100
0
20
40
60
80
100

Bibliography

[DG] David Gleich, *PageRank beyond the web*, SIAM Review, 57(3):321–363, 2015.

[HNO] Per Christian Hansen, James Nagy, and Dianne O'Leary, *Image Deblurring*, SIAM, 2006.

[MH] Michael Heath, *Scientific Computing*, SIAM, Revised 2nd edition, 2018.

[KN] Keith Nicholson, *Linear Algebra with Applications*, Lyryx, Version 2019 Revision A, 2019.