Applied Linear Algebra

Gábor P. Nagy and Viktor Vígh

University of Szeged
Bolyai Institute

Winter 2014

Table of contents I

Table of contents II

Section 1

Introduction, review

Subsection 1

Complex numbers

Complex numbers

Definition: Complex numbers

Let i be a symbol with $i \notin \mathbb{R}$ and $x, y \in \mathbb{R}$ real numbers. The sum $z = x + iy$ is called a complex number, the set of all complex numbers is denoted by $\mathbb{C}$. The complex number

$$\bar{z} = x - iy$$

is the conjugate of the complex number z. For $z_1 = x_1 + iy_1$, $z_2 = x_2 + iy_2$ we define their sum and product as follows:

$$\begin{aligned} z_1 + z_2 &= x_1 + x_2 + i(y_1 + y_2), \\ z_1 z_2 &= (x_1 x_2 - y_1 y_2) + i(x_1 y_2 + y_1 x_2). \end{aligned}$$

Dividing complex numbers

Observe that for a complex number $z = x + iy \neq 0$ the product

$$z\bar{z} = (x + iy)(x - iy) = x^2 + y^2 \neq 0$$

is a real number.

Definition: reciprocal of a complex number

The reciprocal of the complex number $z = x + iy \neq 0$ is the complex number

$$z^{-1} = \frac{x}{x^2 + y^2} - \frac{y}{x^2 + y^2} i.$$

Proposition

Let $c_1, c_2 \in \mathbb{C}$ be complex numbers. If $c_1 \neq 0$, then the equation $c_1 z = c_2$ has the unique solution $z = c_1^{-1} c_2 \in \mathbb{C}$ in the set of complex numbers.

Basic properties of complex numbers

Proposition: basic properties

For any complex numbers $a, b, c \in \mathbb{C}$ the following hold:

- $a + b = b + a$, $ab = ba$ (commutativity)
- $(a + b) + c = a + (b + c)$, $(ab)c = a(bc)$ (associativity)
- $0 + a = a$, $1a = a$ (neutral elements)
- $-a \in \mathbb{C}$, and if $b \neq 0$ then $b^{-1} \in \mathbb{C}$ (inverse elements)
- $(a + b)c = ac + bc$ (multiplication distributes over addition)

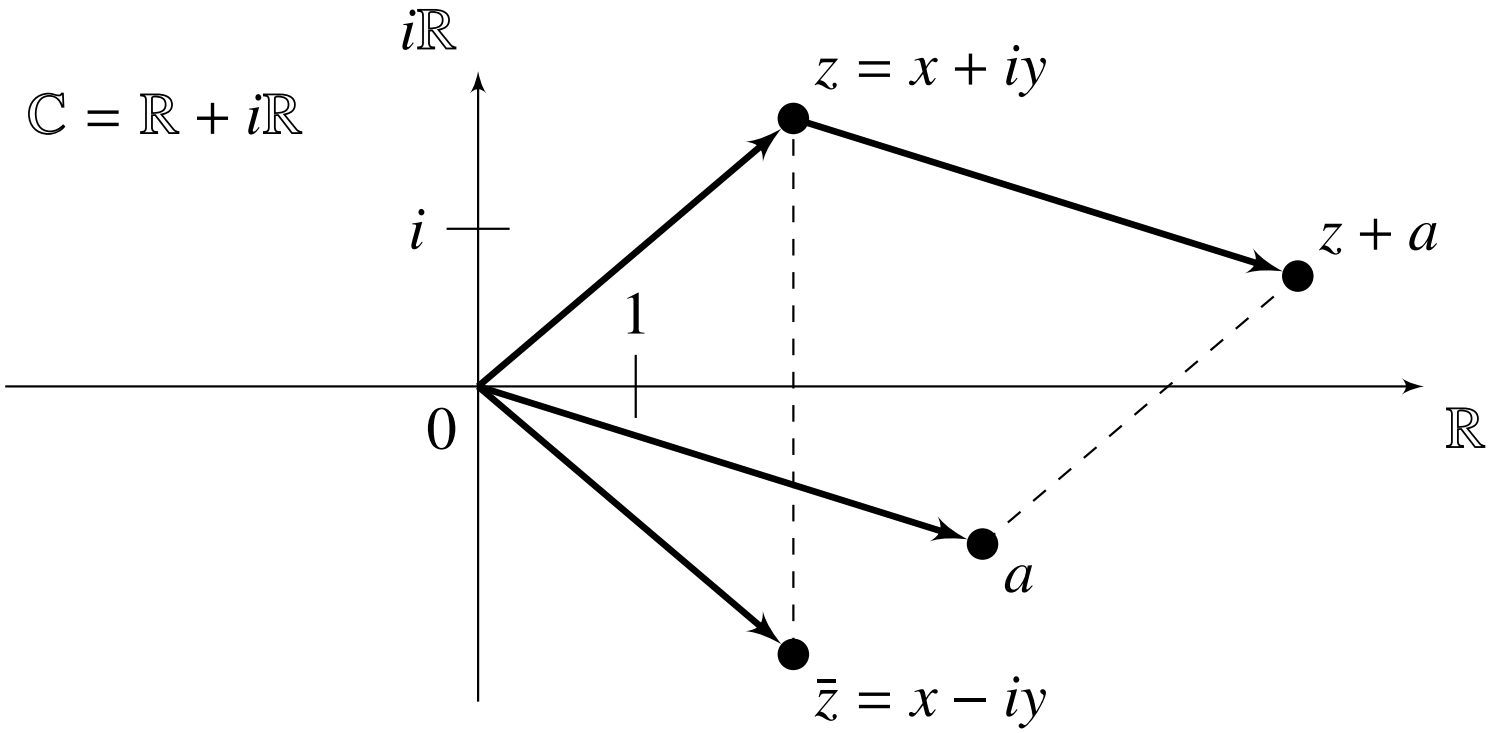
That is, $\mathbb{C}$ is a field.

Conjugation preserves addition and multiplication: $\overline{a + b} = \bar{a} + \bar{b}$, $\overline{ab} = \bar{a}\,\bar{b}$.

Furthermore, $i^2 = -1$, $z\bar{z} \in \mathbb{R}$, $z\bar{z} \geq 0$ and $z\bar{z} = 0 \Leftrightarrow z = 0$.

The complex plane

We may represent complex numbers as vectors.

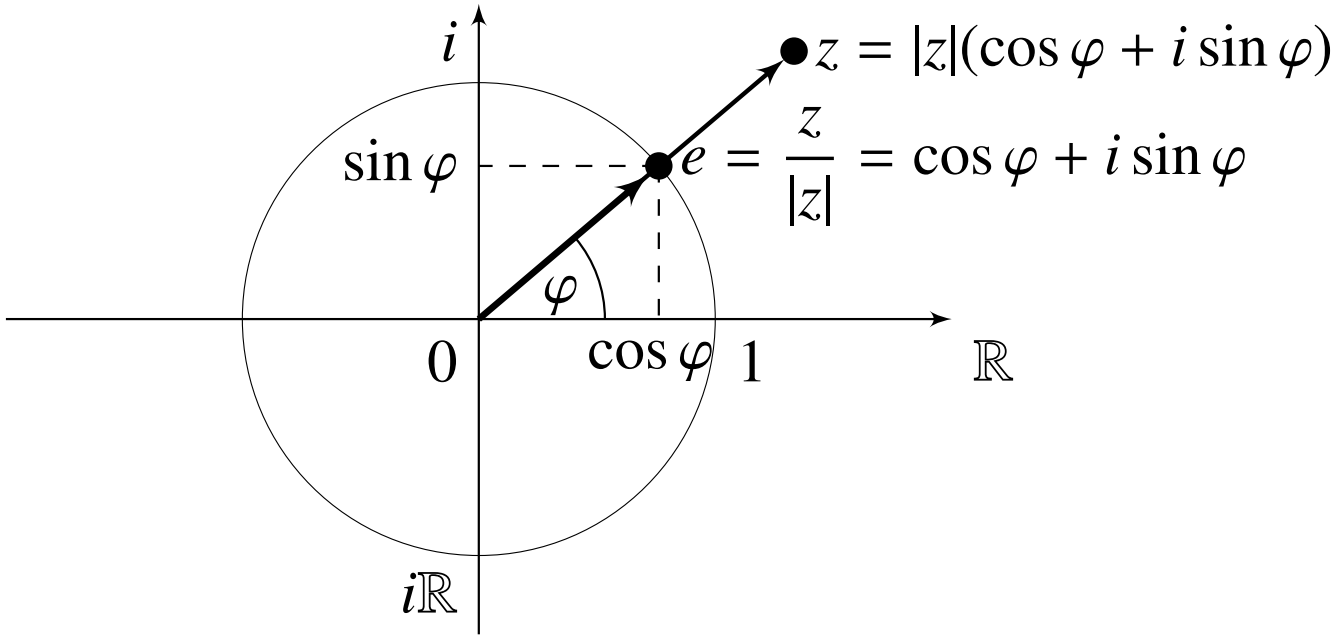


Absolute value, argument, polar form

Definition

Let $z = x + iy \in \mathbb{C}$ be a complex number. The real number $|z| = \sqrt{z\bar{z}} = \sqrt{x^2 + y^2}$ is called the norm or absolute value of the complex number z.

Any complex number $z \in \mathbb{C}$ can be written in the form $z = |z|(\cos\varphi + i\sin\varphi)$, where φ is the argument of z. This is called the polar or trigonometric form of the complex number z.

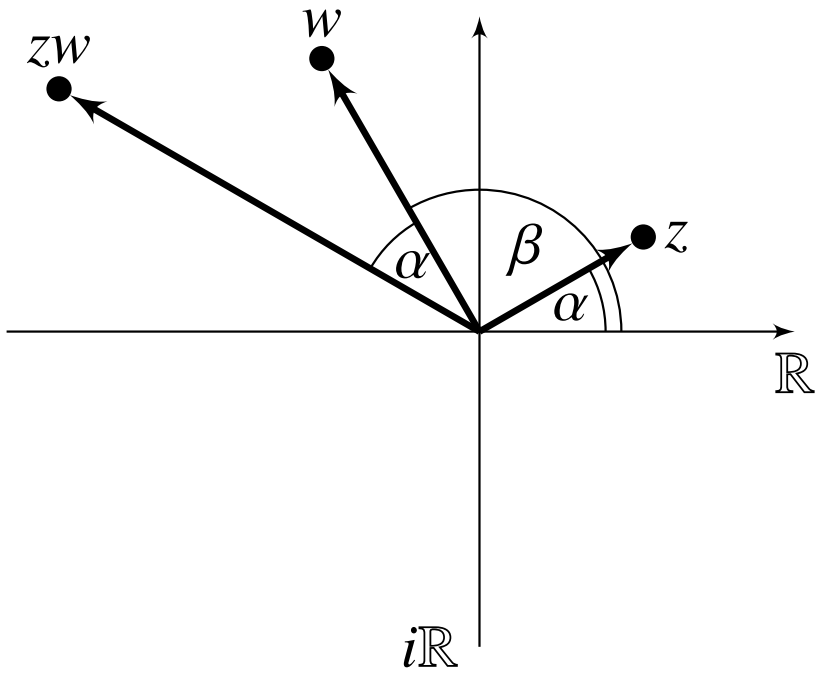


Multiplication via polar form

Proposition

When multiplying two complex numbers their absolute values multiplies out, while their arguments add up.

$$\begin{aligned} z &= r(\cos\alpha + i\sin\alpha) \\ w &= s(\cos\beta + i\sin\beta) \\ zw &= rs(\cos(\alpha+\beta) + i\sin(\alpha+\beta)) \end{aligned}$$



Corollary:

$$z^n = r^n(\cos(n\alpha) + i\sin(n\alpha))$$

Subsection 2

Vectors and matrices

Vectors, Matrices

Definition: Vector

An ordered n-tuple of (real or complex) numbers is called an n-dimensional (real or complex) vector: $\mathbf{v} = (v_1, v_2, \ldots, v_n) \in \mathbb{R}^n$ or $\mathbb{C}^n$.

Definition: Matrix

A matrix is a rectangular array of (real or complex) numbers arranged in rows and columns. The individual items in a matrix are called its elements or entries.

Example

A is a 2×3 matrix. The set of all 2×3 real matrices is denoted by $\mathbb{R}^{2\times 3}$.

$$A = \begin{bmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \end{bmatrix} \in \mathbb{R}^{2\times 3}.$$

Normally we put matrices between square brackets.

Vectors, Matrices

Notation

We denote by a_{ij} the jth elementh of the ith row in the matrix A. If we want to emphasize the notation, we may write $A = [a_{ij}]_{n\times m}$, where $n \times m$ stands for the size of A.

Example

$$A = \begin{bmatrix} 1 & 0 & -4 & 0 \\ 5 & -1 & 7 & 0 \end{bmatrix} \in \mathbb{R}^{2\times 4}$$

$a_{21} = 5$, $a_{13} = -4$

Conventions

We consider vectors as column vectors, i. e. vectors are $n \times 1$ matrices. We do not distinguish between a 1×1 matrix and its only element, unless it is neccessary. If we do not specify the size of a matrix, it is assumed to be a square matrix of size $n \times n$.

Operations with vectors

Definition: addition, multiplication by scalar

Let $\mathbf{v} = (v_1, v_2, \ldots, v_n)^T$ and $\mathbf{w} = (w_1, w_2, \ldots, w_n)^T$ be two n-dimensional (real or complex) vectors, and λ a (real or complex) scalar. The sum of $\mathbf{v}$ and $\mathbf{w}$ is $\mathbf{v} + \mathbf{w} = (v_1 + w_1, v_2 + w_2, \ldots, v_n + w_n)^T$, while $\lambda\mathbf{v} = (\lambda v_1, \lambda v_2, \ldots, \lambda v_n)^T$.

Definition: linear combination

Let $\mathbf{v}_1, \mathbf{v}_2, \ldots, \mathbf{v}_m$ be vectors of the same size, and $\lambda_1, \lambda_2, \ldots, \lambda_m$ scalars. The vector $\mathbf{w} = \lambda_1\mathbf{v}_1 + \lambda_2\mathbf{v}_2 + \ldots + \lambda_m\mathbf{v}_m$ is called the linear combination of $\mathbf{v}_1, \mathbf{v}_2, \ldots, \mathbf{v}_m$ with coefficients $\lambda_1, \lambda_2, \ldots, \lambda_m$.

Example

For $\mathbf{v} = (2, 1, -3)^T$, $\mathbf{w} = (1, 0, -2)^T$, $\lambda = -4$ and $\mu = 2$ we have $\lambda\mathbf{v} + \mu\mathbf{w} = (-6, -4, 8)^T$.

Subspace, basis

Definition: subspace

The set $\emptyset \neq H \subseteq \mathbb{R}^n$ (or $\subseteq \mathbb{C}^n$) is a subspace if for any $\mathbf{v}, \mathbf{w} \in H$, and for any scalar λ we have $\mathbf{v} + \mathbf{w} \in H$ and $\lambda\mathbf{v} \in H$, that is H is closed under addition and multiplication by scalar.

Example

Well known examples in $\mathbb{R}^3$ are lines and planes that contain the origin. Also notice that $\mathbb{R}^n$ itself and the one point set consisting of the zero vector are subspaces.

Definition: basis

Let H be a subspace in $\mathbb{R}^n$ (or in $\mathbb{C}^n$). The vectors $\mathbf{b}_1, \ldots, \mathbf{b}_m$ form a basis of H if any vector $\mathbf{v} \in H$ can be written as the linear combination of $\mathbf{b}_1, \ldots, \mathbf{b}_m$ in a unique way.

Linear independence, dimension

Definition: linear independence

The vectors $\mathbf{v}_1, \mathbf{v}_2, \ldots, \mathbf{v}_m$ are linearly independent, if from $\mathbf{0} = \lambda_1\mathbf{v}_1 + \lambda_2\mathbf{v}_2 + \ldots + \lambda_m\mathbf{v}_m$ it follows that $\lambda_1 = \lambda_2 = \ldots = \lambda_m = 0$, that is the zero vector can be written as a linear combination of $\mathbf{v}_1, \mathbf{v}_2, \ldots, \mathbf{v}_m$ only in a trivial way.

Note that the elements of a basis are linearly independent by definition.

Proposition

For every subspace H there exists a basis. Furthermore any two bases of H consist of the same number of vectors. This number is called the dimension of the subspace H. (The subspace consisting of the zero vector only is considered to be zero dimensional.)

Scalar product, length

Definition: scalar product, length

Let $\mathbf{v} = (v_1, v_2, \ldots, v_n)^T$ and $\mathbf{w} = (w_1, w_2, \ldots, w_n)^T$ be two n-dimensional vectors. The scalar product of $\mathbf{v}$ and $\mathbf{w}$ is the scalar

$$\langle \mathbf{v}\,,\ \mathbf{w} \rangle = \mathbf{v} \cdot \mathbf{w} = v_1 w_1 + v_2 w_2 + \ldots + v_n w_n.$$

The length (or Euclidean norm) of the vector $\mathbf{v}$ is defined by

$$\|\mathbf{v}\|_2 = \sqrt{\langle \mathbf{v}\,,\ \mathbf{v} \rangle} = \sqrt{v_1^2 + v_2^2 + \ldots + v_n^2}.$$

Example

$$(1, 0, -3)^T \cdot (2, 1, -2)^T = 2 \cdot 1 + 0 \cdot 1 + -3 \cdot -2 = 8$$

$$\|(1, 0, -3)^T\|_2 = \sqrt{1 + 0 + 9} = \sqrt{10}.$$

Properties of the scalar product

Proposition

The geometric meaning of the scalar product is the following statement:

$$\langle \mathbf{v}\,,\ \mathbf{w}\rangle = \|\mathbf{v}\|_2 \cdot \|\mathbf{w}\|_2 \cdot \cos\varphi,$$

where φ is the angle between $\mathbf{v}$ and $\mathbf{w}$.

Elementary properties of the scalar product

- Commutative: $\langle \mathbf{v}\,,\ \mathbf{w}\rangle = \langle \mathbf{w}\,,\ \mathbf{v}\rangle$
- Bilinear: $\langle \lambda\mathbf{v} + \mathbf{w}\,,\ \mathbf{u}\rangle = \lambda\langle \mathbf{v}\,,\ \mathbf{u}\rangle + \langle \mathbf{w}\,,\ \mathbf{u}\rangle$
- Positive definite: $\langle \mathbf{v}\,,\ \mathbf{v}\rangle \geq 0$, and equality holds iff $\mathbf{v} = \mathbf{0}$

Remark. A vector $\mathbf{v}$ is of unit length iff $\langle \mathbf{v}\,,\ \mathbf{v}\rangle = 1$. Two vectors $\mathbf{v}$ and $\mathbf{w}$ are orthogonal (perpendicular) iff $\langle \mathbf{v}\,,\ \mathbf{w}\rangle = 0$.

Special matrices

Definition: Diagonal matrix

The matrix A is diagonal if $a_{ij} = 0$ when $i \neq j$. A 3×3 example is

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

Definition: Tridiagonal matrix

The matrix A is tridiagonal if $a_{ij} = 0$ when $|i - j| > 1$. A 4×4 example is

$$A = \begin{bmatrix} 1 & 2 & 0 & 0 \\ -1 & 5 & 2 & 0 \\ 0 & 3 & -2 & 3 \\ 0 & 0 & 1 & 7 \end{bmatrix}$$

Special matrices

Definition: Upper (lower) triangular matrix

The matrix A is upper (lower) triangular if $a_{ij} = 0$ when $i > j$ ($i < j$). A 3×3 example of an upper triangular matrix is

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

Definition: Upper (lower) Hessenberg matrix

The matrix A is upper (lower) Hessenberg if $a_{ij} = 0$ when $i - 1 > j$ ($i < j - 1$). A 4×4 example of an upper Hessenberg matrix is

$$A = \begin{bmatrix} 1 & 2 & 5 & 9 \\ -1 & 5 & 2 & -1 \\ 0 & 3 & -2 & 3 \\ 0 & 0 & 1 & 7 \end{bmatrix}$$

Special matrices

Definition: Identity matrix

The $n \times n$ identity matrix I_n is a diagonal matrix with $a_{ii} = 1$ for all $i = 1, \ldots, n$. The 3×3 example is

$$I_3 = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

Definition: Zero matrix

A matrix with all zero elements is called a zero matrix. A 3×4 example of a zero matrix is

$$O_{3\times 4} = \begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix}$$

Transpose of a matrix

Definition: Transpose

Let $A = [a_{ij}]_{n\times m}$ be a matrix. Its transpose is the matrix $B = [b_{ji}]_{m\times n}$ with $b_{ji} = a_{ij}$. We denote the transpose of A by A^T.

Example

$$A = \begin{bmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \end{bmatrix} \quad \rightarrow \quad A^T = \begin{bmatrix} 1 & 4 \\ 2 & 5 \\ 3 & 6 \end{bmatrix}$$

Roughly speaking transposing is reflecting the matrix with respect to its main diagonal.

Scalar multiple of a matrix

Definition: Scalar multiple

Let $A = [a_{ij}]_{n\times m}$ be a matrix and λ is a scalar. The scalar multiple of A by λ is the matrix $\lambda A = [b_{ij}]_{n\times m}$ with $b_{ij} = \lambda a_{ij}$.

Example

$$A = \begin{bmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \end{bmatrix} \quad \rightarrow \quad -3 \cdot A = \begin{bmatrix} -3 & -6 & -9 \\ -12 & -15 & -18 \end{bmatrix}$$

In words, we multiply the matrix A elementwise by λ.

Adding matrices

Definition: Matrix addition

Let $A = [a_{ij}]_{n\times m}$ and $B = [b_{ij}]_{n\times m}$ be two matrices of the same size. The sum of A and B is defined by $A + B = C = [c_{ij}]_{n\times m}$ with $c_{ij} = a_{ij} + b_{ij}$ for all $i = 1, \ldots, n$ and $j = 1, \ldots, m$.

Example

$$\begin{bmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \end{bmatrix} + \begin{bmatrix} 1 & 0 & -4 \\ 5 & -1 & 7 \end{bmatrix} = \begin{bmatrix} 2 & 2 & -1 \\ 9 & 4 & 13 \end{bmatrix}$$

Multiplying matrices

Definition: Matrix multiplication

Let $A = (a_{ij})_{n\times m}$ and $B = (b_{ij})_{m\times k}$ be two matrices. The product of A and B is defined by $A \cdot B = C = (c_{i,j})_{n\times k}$ where for all $1 \leq j \leq n$ and for all $1 \leq \ell \leq k$ we have

$$c_{j,\ell} = \sum_{i=1}^{m} a_{j,i} \cdot b_{i,\ell}.$$

Example

$$\begin{bmatrix} 1 & 2 & 3 \end{bmatrix} \cdot \begin{bmatrix} 4 \\ 5 \\ 6 \end{bmatrix} = [1 \cdot 4 + 2 \cdot 5 + 3 \cdot 6] = [32]$$

Example

$$A = \begin{bmatrix} -1 & -2 & 0 \\ 0 & 1 & 0 \\ 1 & 1 & 1 \\ 3 & -4 & 2 \end{bmatrix}, \qquad B = \begin{bmatrix} 1 & 0 \\ 0 & -1 \\ 2 & 1 \end{bmatrix}$$

$$\begin{array}{ccc|cc} & & & 1 & 0 \\ & & & 0 & -1 \\ & & & 2 & 1 \\ \hline -1 & -2 & 0 & -1 & 2 \\ 0 & 1 & 0 & 0 & -1 \\ 1 & 1 & 1 & 3 & 0 \\ 3 & -4 & 2 & 7 & 6 \end{array} \qquad A \cdot B = \begin{bmatrix} -1 & 2 \\ 0 & -1 \\ 3 & 0 \\ 7 & 6 \end{bmatrix}$$

Basic properties of matrix operations

Proposition

- $A + B = B + A$
- $(A + B) + C = A + (B + C)$
- $(AB)C = A(BC)$
- $(AB)^T = B^T A^T$
- The product (if it exists) of upper (lower) triangular matrices is upper (lower) triangular.

Caution!

Matrix multiplication is not commutative!

Special matrices

Definition: Symmetric matrix

The matrix A is symmetric if $a_{ij} = a_{ji}$ for each i, j. In other words A is symmetric if $A^T = A$. A 3×3 example is

$$A = \begin{bmatrix} 3 & 4 & 6 \\ 4 & 1 & -2 \\ 6 & -2 & 0 \end{bmatrix}$$

Definition: Nonsigular matrix, inverse

The matrix A is nonsingular if there exists a matrix A^{-1} with $AA^{-1} = A^{-1}A = I_n$. The matrix A^{-1} is the inverse of A.

Definition: Positive-definite

The real matrix A is positive-definite if it is symmetric and $\mathbf{x}^T A\mathbf{x} > 0$ for all $\mathbf{x} \neq \mathbf{0}$. If we require $\mathbf{x}^T A\mathbf{x} \geq 0$ for all $\mathbf{x} \neq \mathbf{0}$, then we say A is positive-semidefinite.

Orthogonal matrices

Definition: Orthogonal matrix

The matrix A is orthogonal if $A^T A = AA^T = I_n$.

Let $\mathbf{c_i}$ be the ith column vector of the orthogonal matrix A. Then

$$\mathbf{c_i}^T \mathbf{c_j} = \sum_{k=1}^{n} a_{ki} a_{kj} = \sum_{k} (A^T)_{ik} (A)_{kj} = (I_n)_{ij} = \begin{cases} 0, & \text{if} \quad i \neq j \\ 1, & \text{otherwise} \end{cases}$$

Thus, the column vectors are of unit length, and are pairwise orthogonal. Similar can be shown for the row vectors.

Eigenvalues, eigenvectors

Definition: Eigenvalue, eigenvector

Let A be a complex $n \times n$ square matrix. The pair $(\lambda, \mathbf{v})$ $(\lambda \in \mathbb{C}, \mathbf{v} \in \mathbb{C}^n)$ is called an eigenvalue, eigenvector pair of A if $\lambda\mathbf{v} = A\mathbf{v}$.

Proposition

If A is a real symmetric matrix, then all eigenvalues of A are real numbers.

Proposition

If A is positive-definite, then all eigenvalues of A are positive real numbers. If A is positive-semidefinite, then all eigenvalues of A are nonnegative real numbers.

Subsection 3

Determinants

Definition

Let A be a square matrix. We associate a number to A called the determinant of A as follows.

Example

If A a 1×1 matrix, then the determinant is the only element of A.

$$\det[3] = 3 \quad , \quad \det[-4] = -4 \quad , \quad \det[a] = a.$$

For 2×2 matrices the determinant is the product of the elements in the main diagonal minus the product of the elements of the minor diagonal.

$$\det\begin{bmatrix} 1 & 2 \\ 3 & 4 \end{bmatrix} = 1 \cdot 4 - 2 \cdot 3 = -2 \quad , \quad \det\begin{bmatrix} a & b \\ c & d \end{bmatrix} = ad - bc.$$

Definition

Submatrix

Let A be an $N \times N$ matrix. The $(N-1) \times (N-1)$ matrix that we obtain by deleting the ith row and the jth column form A is denoted by A_{ij}.

Example

$$A = \begin{bmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \\ 7 & 8 & 9 \end{bmatrix}, \; A_{33} = \begin{bmatrix} 1 & 2 \\ 4 & 5 \end{bmatrix}, \; A_{12} = \begin{bmatrix} 4 & 6 \\ 7 & 9 \end{bmatrix}$$

Definition: Determinant

We define the determinant of 1×1 and 2×2 matrices as above.
Let A be an $N \times N$ matrix ($N \geq 3$). We define the determinant of A recursively as follows

$$\det A = \sum_{k=1}^{N} (-1)^{k+1} a_{1k} \det A_{1k}.$$

Example

Example

$$\det\begin{bmatrix} 2 & 3 & -4 \\ 1 & 0 & -2 \\ 2 & 5 & -1 \end{bmatrix} = (-1)^2 \cdot 2 \cdot \det\begin{bmatrix} 0 & -2 \\ 5 & -1 \end{bmatrix} +$$

$$+(-1)^3 \cdot 3 \cdot \det\begin{bmatrix} 1 & -2 \\ 2 & -1 \end{bmatrix} + (-1)^4 \cdot (-4) \cdot \det\begin{bmatrix} 1 & 0 \\ 2 & 5 \end{bmatrix} =$$

$$2 \cdot (0 \cdot (-1) - 5 \cdot (-2)) - 3 \cdot (1 \cdot (-1) - 2 \cdot (-2)) + (-4) \cdot (1 \cdot 5 - 2 \cdot 0) =$$

$$= 20 - 9 - 20 = -9$$

Properties of determinants

The following statement is fundamental in understanding determinants.

Proposation

If we swap two rows (or two columns, respectively) of a matrix, its determinant multiplies by -1.

Example

$$\det\begin{bmatrix} 2 & 3 & -4 \\ 1 & 0 & -2 \\ 2 & 5 & -1 \end{bmatrix} = (-1)\cdot\det\begin{bmatrix} 2 & 5 & -1 \\ 1 & 0 & -2 \\ 2 & 3 & -4 \end{bmatrix}$$

Corollary

If two rows (or two columns, respectively) of a matrix are identical, then its determinant is 0.

Properties of determinants

Theorem

Let A be a square matrix, and use the notation introduced above.

$$\sum_{j=1}^{N}(-1)^{(i+j)} \cdot a_{kj} \cdot \det A_{ij} = \begin{cases} \det A & \text{, if } k = i, \\ 0 & \text{, otherwise.} \end{cases}$$

In particular, we may develop a determinant with respect to any row (or column).

Corollary

The determinant won't change if we add a multiple of a row (or column, respectively) to another row (or column, respectively).

Properties of determinants

Lemma

1. $\det I_N = 1$
2. If A is upper or lower triangular, then $\det A = a_{11} \cdot a_{22} \cdot \ldots \cdot a_{NN}$.
3. $\det(A^T) = \det A$
4. $\det(A^{-1}) = (\det A)^{-1}$

Theorem

Let A be B two square matrices of the same size. Then

$$\det(AB) = \det A \cdot \det B.$$

Subsection 4

Vector and matrix norms

Vector norms

Definition: p-norm of a vector

Let $p \geq 1$, the p-norm (or L_p-norm) of a vector $\mathbf{x} = [x_1, \ldots, x_n]^T$ is defined by

$$\|\mathbf{x}\|_p = (|x_1|^p + \ldots + |x_n|^p)^{1/p}.$$

Example

Let $\mathbf{x} = [2, -3, 12]^T$, then

$$\|\mathbf{x}\|_1 = |2| + |-3| + |12| = 17,$$

and

$$\|\mathbf{x}\|_2 = \sqrt{2^2 + (-3)^2 + 12^2} = \sqrt{157}.$$

Vector norms

Let $\mathbf{x} = [x_1, \ldots, x_n]^T$, and $M = \max |x_i|$. Then

$$\lim_{p\to\infty} \|\mathbf{x}\|_p = \lim_{p\to\infty} \left[M \left(\frac{|x_1|^p}{M^p} + \ldots + \frac{|x_n|^p}{M^p} \right)^{1/p} \right] = \lim_{p\to\infty} [M(K)^{1/p}] = M,$$

thus the following definition is reasonable.

Definition: ∞-norm of a vector

For $\mathbf{x} = [x_1, \ldots, x_n]^T$ we define

$$\|\mathbf{x}\|_\infty = \max_{1\le i\le n} |x_i|.$$

Example

Let $\mathbf{x} = [2, -3, 12]^T$, then

$$\|\mathbf{x}\|_\infty = \max\{|2|, |-3|, |12|\} = 12.$$

Matrix norms

Definition: Matrix norm

For each vector norm ($p \in [1, \infty]$), we define an associated matrix norm as follows

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

Proposition

$$\|A\mathbf{x}\|_p \leq \|A\|_p \|\mathbf{x}\|_p.$$

Remark. The definition does not require that A be square, but we assume that throughout the course.

The cases $p = 1, 2, \infty$ are of particular interest. However, the definition is not feasible to calculate norms directly.

Matrix norms

Theorem

Let $A = [a_{ij}]_{n\times n}$ be a square matrix, then

a) $\|A\|_1 = \max_{1\le j\le n} \sum_{i=1}^{n} |a_{ij}|$,

b) $\|A\|_\infty = \max_{1\le i\le n} \sum_{j=1}^{n} |a_{ij}|$,

c) $\|A\|_2 = \mu_{max}^{1/2}$,

where μ_{max} is the largest eigenvalue of A^TA.

We remark that $(A^TA)^T = A^TA$, hence A^TA is symmetric, thus its eigenvalues are real. Also, $\mathbf{x}^T(A^TA)\mathbf{x} = (A\mathbf{x})^T(A\mathbf{x}) = \|A\mathbf{x}\|_2^2 \ge 0$, and so $\mu_{max} \ge 0$.

We only sketch the proof of part a). Part b) can be shown similarly, while we are going to come back to the eigenvalue problem later in the course.

Proof

Write $\|A\|_1^* = \max_{1\le j\le n} \sum_{i=1}^n |a_{ij}|$.

$$\|A\mathbf{x}\|_1 = \sum_{i=1}^n |(A\mathbf{x})_i| = \sum_{i=1}^n \left|\sum_{j=1}^n a_{ij}x_j\right| \le \sum_{i=1}^n \sum_{j=1}^n |a_{ij}||x_j|$$

$$= \sum_{j=1}^n \left(\sum_{i=1}^n |a_{ij}|\right)|x_j| \le \left(\max_{1\le j\le n} \sum_{i=1}^n |a_{ij}|\right) \sum_{j=1}^n |x_j| = \|A\|_1^*\|\mathbf{x}\|_1$$

Thus, $\|A\|_1^* \ge \|A\|_1$.

Now, let J be the value of j for which the column sum $\sum_{i=1}^n |a_{ij}|$ is maximal, and let $\mathbf{z}$ be a vector with all 0 elements except $z_J = 1$. Then $\|\mathbf{z}\|_1 = 1$, and $\|A\|_1\|\mathbf{z}\|_1 \ge \|A\mathbf{z}\|_1 = \|A\|_1^* = \|A\|_1^*\|\mathbf{z}\|_1$, hence $\|A\|_1^* \le \|A\|_1$. This completes the proof of part a).

Example

Example

Compute the $\|\cdot\|_1$, $\|\cdot\|_2$ and $\|\cdot\|_\infty$ norms of the following matrix

$$A = \begin{bmatrix} 2 & 3 & -4 \\ 1 & 0 & -2 \\ 2 & 5 & -1 \end{bmatrix}.$$

For the column sums $\sum_{i=1}^{3} |a_{ij}|$ we obtain 5, 8 and 7 for $j = 1, 2$ and 3 respectively. Thus $\|A\|_1 = \max\{5, 8, 7\} = 8$.

Similarly, for the row sums $\sum_{j=1}^{3} |a_{ij}|$ we obtain 9, 3 and 8 for $i = 1, 2$ and 3 respectively. Thus $\|A\|_\infty = \max\{9, 3, 8\} = 9$.

Example

To find the $\|\cdot\|_2$ norm, first we need to compute AA^T.

$$AA^T = \begin{bmatrix} 29 & 10 & 23 \\ 10 & 5 & 4 \\ 23 & 4 & 30 \end{bmatrix}.$$

The eigenvales of AA^T are (approximately) 54.483, 9.358 and 0.159. (We are going to study the problem of finding the eigenvalues of a matrix later during the course.)

Thus $\|A\|_2 = \sqrt{54.483} = 7.381$.

Section 2

Systems of linear equations

Subsection 1

Gaussian Elimination

Systems of linear equations

Example

$$\begin{cases} x_1 + x_2 + 2x_4 + x_5 = 1 \\ -x_1 - x_2 + x_3 - x_4 - x_5 = -2 \\ 2x_1 + 2x_2 + 2x_3 + 6x_4 + x_5 = 0 \end{cases}$$

Definition: System of linear equations

Let $A \in \mathbb{C}^{n\times m}$ and $\mathbf{b} \in \mathbb{C}^n$. The equation $A\mathbf{x} = \mathbf{b}$ is called a system of linear equations, where $\mathbf{x} = [x_1, x_2, \ldots, x_n]^T$ is the unknown, A is called the coefficient matrix, and $\mathbf{b}$ is the constant vector on the right hand side of the equation.

Matrix form

$$\left[\begin{array}{ccccc|c} 1 & 1 & 0 & 2 & 1 & 1 \\ -1 & -1 & 1 & -1 & -1 & -2 \\ 2 & 2 & 2 & 6 & 1 & 0 \end{array}\right] = [A|\mathbf{b}]$$

Solution of a system of linear equations

Definition: solution

Let $A\mathbf{x} = \mathbf{b}$ be a system of linear equations. The vector $\mathbf{x}_0$ is a solution of the system if $A\mathbf{x}_0 = \mathbf{b}$ holds true.

How to solve a system?

We say that a system of linear equations is solved if we found **all solutions**.

Example

Consider the following system over the real numbers:

$$\begin{cases} x_1 + x_2 = 1 \\ x_3 = -2 \end{cases}$$

Solution: $S = \{(1-t, t, -2)^T \mid t \in \mathbb{R}\}$.

Solution of a system of linear equations

Theorem

A system of linear equations can have either 0, 1 or infinitely many solutions.

As a start, we are going to deal with a system of linear equations $A\mathbf{x} = \mathbf{b}$ where A is a sqaure matrix, and the system has a unique solution. Later, we are going to examine the other cases as well.

Proposition

The system of linear equations $A\mathbf{x} = \mathbf{b}$ (A is a square matrix) has a unique solution if, and only if, A is nonsingular.

We don't prove this statement now, it will be trasparent later during the course.

Solving upper triangular systems

Example

Solve the following system of linear equations!

$$\begin{cases} x_1 + 2x_2 + x_3 = 1 \\ 2x_2 + x_3 = -2 \\ 4x_3 = 0 \end{cases}$$

This system can be solved easily, by substituting back the known values, starting from the last equation. First, we obtain $x_3 = 0/4 = 0$, then from the second equation we get $x_2 = -2 - x_3 = -2$. Finally we calculate $x_1 = 1 - 2x_2 - x_3 = 5$.

From this example one can see that solving a system where A is upper triangular is easy, and has a unique solution provided that the diagonal elements differ from 0. Thus, we shall transform our system to an upper triangular form with nonzero diagonals.

Homogeneous systems

Example

Consider the following system of linear equations:

$$\begin{cases} x_1 & + & 2x_2 & + & x_3 & = & 0 \\ -x_1 & + & 2x_2 & + & x_3 & = & 0 \\ 2x_1 & + & & & 4x_3 & = & 0 \end{cases}$$

In this case $\mathbf{b} = \mathbf{0}$. Observe, that $\mathbf{x} = \mathbf{0}$ is a solution. (It's not hard to check, that in this case $\mathbf{0}$ is the only solution.)

Definition: homogeneous systems of linear equations

A system of linear equations $A\mathbf{x} = \mathbf{b}$ is called homogenous if $\mathbf{b} = \mathbf{0}$.

As a consequence, we readily deduce the following

Theorem

A homogeneous system of linear equations can have either 1 or infinitely many solutions.

Gaussian elemination

Example

Solve the following system of linear equations!

$$\begin{cases} x_1 - 2x_2 - 3x_3 = 0 & (I.) \\ 2x_2 + x_3 = -8 & (II.) \\ -x_1 + x_2 + 2x_3 = 3 & (III.) \end{cases}$$

We use the matrix form to carry out the algorithm. The idea is that adding a multiple of an equation to another equation won't change the solution.

$$\left[\begin{array}{ccc|c} 1 & -2 & -3 & 0 \\ 0 & 2 & 1 & -8 \\ -1 & 1 & 2 & 3 \end{array}\right] \overset{III.+I.}{\sim} \left[\begin{array}{ccc|c} 1 & -2 & -3 & 0 \\ 0 & 2 & 1 & -8 \\ 0 & -1 & -1 & 3 \end{array}\right]$$

Gaussian elimination

$$\left[\begin{array}{ccc|c} 1 & -2 & -3 & 0 \\ 0 & 2 & 1 & -8 \\ 0 & -1 & -1 & 3 \end{array}\right] \overset{III.+II./2}{\sim} \left[\begin{array}{ccc|c} 1 & -2 & -3 & 0 \\ 0 & 2 & 1 & -8 \\ 0 & 0 & -1/2 & -1 \end{array}\right]$$

After substituting back, we obtain the unique solution:
$x_1 = -4, x_2 = -5, x_3 = 2$.

The main idea was to cancel every element under the main diagonal going column by column. First we used a_{11} to eliminate a_{21} and a_{31}. (In the example a_{21} was already zero, we didn't need to do anything with it.) Then we used a_{22} to eliminate a_{32}. Note that this second step cannot ruin the previously produced zeros in the first column. This idea can be generalized to larger systems.

Gaussian elimination

Consider a general system $A\mathbf{x} = \mathbf{b}$, where A is an $N \times N$ square matrix.

$$\left[\begin{array}{ccccc|c} a_{11} & a_{12} & a_{13} & \cdots & a_{1N} & b_1 \\ a_{21} & a_{22} & a_{23} & \cdots & a_{2N} & b_2 \\ a_{31} & a_{32} & a_{33} & \cdots & a_{3N} & b_3 \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ a_{N1} & a_{N2} & a_{N3} & \cdots & a_{NN} & b_N \end{array}\right]$$

Assume that $a_{11} \neq 0$. To eliminate the first column, for $j = 2, \ldots, N$ we take the $-a_{j1}/a_{11}$ multiple of the first row, and add it to the jth row, to make $a_{j1} = 0$.
If $a_{11} = 0$, then first we find a j with $a_{j1} \neq 0$, and swap the first row with the jth row, and then proceed as before.
If the first column of A contains 0's only, then the algorithm stops, and returns that the system doesn't have a unique solution.

Gaussian elimination

Assuming that the algorithm didn't stop, we obtained the following.

$$\left[\begin{array}{ccccc|c} a_{11} & a_{12} & a_{13} & \cdots & a_{1N} & b_1 \\ 0 & a'_{22} & a'_{23} & \cdots & a'_{2N} & b'_2 \\ 0 & a'_{32} & a'_{33} & \cdots & a'_{3N} & b'_3 \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & a'_{N2} & a'_{N3} & \cdots & a'_{NN} & b'_N \end{array}\right]$$

We may procced with a'_{22} as pivot element, and repeat the previous step to eliminate second column (under a'_{22}), and so on. Either the algorithm stops at some point, or after $N-1$ steps we arrive to an upper triangular form, with nonzero diagonal elements, that can be solved backward easily.

Gaussian elimination - via computer

Problem

If we use Gaussian algorithm on a computer, and the pivot element a_{ii} is nearly zero, the computer might produce serious arithmetic errors.

Thus the most important modification to the classical elimination scheme that must be made to produce a good computer algorithm is this: we interchange rows whenever $|a_{ii}|$ is too small, and not only when it is zero.

Several strategies are available for deciding when a pivot is too small to use. We shall see that row swaps require a negligible amount of work compared to actual elimination calculations, thus we will always switch row i with row l, where a_{li} is the largest (in absolute value) of all the potential pivots.

Gaussian elimination with partial pivoting

Gaussian elimination with partial pivoting

Assume that the first $i-1$ columns of the matrix are already eliminated.
Proceed as follows:

1. Search the potential pivots $a_{ii}, a_{i+1,i}, \ldots, a_{Ni}$ for the one that has the largest absolute value.
2. If all potential pivots are zero then stop, the system doesn't have a unique solution.
3. If a_{li} is the potential pivot of the largest absolute value, then switch rows i and l.
4. For all $j = i+1, \ldots, N$ add $-a_{ji}/a_{ii}$ times the ith row to the jth row.
5. Proceed to the $(i+1)$th column.

Running time

For the i th column we make $N - i$ row operations, and in each row operation we need to do $N - i$ multiplication. Thus altogether we need

$$\sum_{i=1}^{N}(N-i)^2 = \sum_{i=1}^{N}(N^2 - 2Ni + i^2) \approx N^3 - N^3 + \frac{N^3}{3} = \frac{N^3}{3}$$

multiplication.
It's not hard to see, that backward substitution and row swaps require $O(N^2)$ running time only.

Theorem

Solving a system of linear equations using Gaussian elimination with partial pivoting and back substitution requires $O(N^3)$ running time, where N is the number of equations (and unknowns).

Row operations revisited

The two row operations used to reduce A to upper triangular form can be thought of as resulting from premultiplications by certain elementary matrices. This idea leads to interesting new observations. We explain the idea through an example.

Example

Apply Gaussian elimination with partial pivoting to the following matrix!

$$A = \begin{bmatrix} 1 & 0 & 1 \\ 2 & 5 & -2 \\ 3 & 6 & 9 \end{bmatrix}$$

(Source of the numerical example: `http://www8.cs.umu.se/kurser/5DV005/HT10/gauss.pdf`)

Row operations revisited

Focus on the first column of $A_1 = A$. We decide to swap row 3 and row 1, because row 3 contains the element which has the largest absolute value. Observe that this change can be carried out by premultiplying with the matrix

$$P_1 = \begin{bmatrix} 0 & 0 & 1 \\ 0 & 1 & 0 \\ 1 & 0 & 0 \end{bmatrix}.$$

Hence

$$\tilde{A}_1 = P_1 A_1 = \begin{bmatrix} 0 & 0 & 1 \\ 0 & 1 & 0 \\ 1 & 0 & 0 \end{bmatrix} \begin{bmatrix} 1 & 0 & 1 \\ 2 & 5 & -2 \\ 3 & 6 & 9 \end{bmatrix} = \begin{bmatrix} 3 & 6 & 9 \\ 2 & 5 & -2 \\ 1 & 0 & 1 \end{bmatrix}.$$

Row operations revisited

Now we are ready to clear the first column of $\tilde{A}_1$. We define

$$M_1 = \begin{bmatrix} 1 & 0 & 0 \\ -\frac{2}{3} & 1 & 0 \\ -\frac{1}{3} & 0 & 1 \end{bmatrix},$$

and so

$$A_2 = M_1\tilde{A}_1 = \begin{bmatrix} 1 & 0 & 0 \\ -\frac{2}{3} & 1 & 0 \\ -\frac{1}{3} & 0 & 1 \end{bmatrix}\begin{bmatrix} 3 & 6 & 9 \\ 2 & 5 & -2 \\ 1 & 0 & 1 \end{bmatrix} = \begin{bmatrix} 3 & 6 & 9 \\ 0 & 1 & -8 \\ 0 & -2 & -2 \end{bmatrix}.$$

Row operations revisited

We continue with the second column of A_2. We swap row 2 and row 3, because row 3 contains the element which has the largest absolute value. This can be carried out by premultiplying with the matrix

$$P_2 = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 0 & 1 \\ 0 & 1 & 0 \end{bmatrix}.$$

We obtain

$$\tilde{A}_2 = P_2 A_2 = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 0 & 1 \\ 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} 3 & 6 & 9 \\ 0 & 1 & -8 \\ 0 & -2 & -2 \end{bmatrix} = \begin{bmatrix} 3 & 6 & 9 \\ 0 & -2 & -2 \\ 0 & 1 & -8 \end{bmatrix}.$$

Row operations revisited

Finally, we take care of the second column of $\tilde{A}_2$. To do this, we define

$$M_2 = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & \frac{1}{2} & 1 \end{bmatrix},$$

and so we get

$$A_3 = M_2\tilde{A}_2 = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & \frac{1}{2} & 1 \end{bmatrix} \begin{bmatrix} 3 & 6 & 9 \\ 0 & -2 & -2 \\ 0 & 1 & -8 \end{bmatrix} = \begin{bmatrix} 3 & 6 & 9 \\ 0 & -2 & -2 \\ 0 & 0 & -9 \end{bmatrix}.$$

Subsection 2

LU decomposition

The upper triangular form

We have now arrived at an upper triangular matrix

$$U = A_3 = \begin{bmatrix} 3 & 6 & 9 \\ 0 & -2 & -2 \\ 0 & 0 & -9 \end{bmatrix}.$$

By construction we have

$$U = A_3 = M_2\tilde{A}_2 = M_2P_2A_2 = M_2P_2M_1\tilde{A}_1 = M_2P_2M_1P_1A_1 = M_2P_2M_1P_1A,$$

or equivalently

$$A = P_1^{-1}M_1^{-1}P_2^{-1}M_2^{-1}U.$$

Inverses of the multipliers

Interchanging any two rows can be undone by interchanging the same two rows one more time, thus

$$P_1^{-1} = P_1 = \begin{bmatrix} 0 & 0 & 1 \\ 0 & 1 & 0 \\ 1 & 0 & 0 \end{bmatrix}, \quad P_2^{-1} = P_2 = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 0 & 1 \\ 0 & 1 & 0 \end{bmatrix}.$$

How can one undo adding a multiple of, say, row 1 to row 3? By subtracting the same multiple of row 1 from the new row 3. Thus

$$M_1^{-1} = \begin{bmatrix} 1 & 0 & 0 \\ \frac{2}{3} & 1 & 0 \\ \frac{1}{3} & 0 & 1 \end{bmatrix}, \quad M_2^{-1} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & -\frac{1}{2} & 1 \end{bmatrix}.$$

LU decomposition

Now, define

$$P = P_2 P_1 = \begin{bmatrix} 0 & 0 & 1 \\ 1 & 0 & 0 \\ 0 & 1 & 0 \end{bmatrix},$$

and multiply both sides of the equation by P! We obtain

$$P_2 P_1 A = P_2 P_1 (P_1^{-1} M_1^{-1} P_2^{-1} M_2^{-1} U) = P_2 M_1^{-1} P_2^{-1} M_2^{-1} U.$$

The good news is that we got rid of P_1 and P_1^{-1}.

We claim, that $P_2 M_1^{-1} P_2^{-1} M_2^{-1}$ is a lower a triangular matrix with all 1's in the main diagonal.

LU decomposition

We know that the effect of multiplying with P_2 from the left is to interchange rows 2 and 3. Similarly, the effect of multiplying with $P_2^{-1} = P_2$ from the right is to interchange columns 2 and 3. Therefore

$$P_2 M_1^{-1} P_2^{-1} = (P_2 M_1^{-1}) P_2 = \begin{bmatrix} 1 & 0 & 0 \\ \frac{1}{3} & 0 & 1 \\ \frac{2}{3} & 1 & 0 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 \\ 0 & 0 & 1 \\ 0 & 1 & 0 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 \\ \frac{1}{3} & 1 & 0 \\ \frac{2}{3} & 0 & 1 \end{bmatrix}.$$

Finally,

$$L = P_2 M_1^{-1} P_2^{-1} M_2^{-1} = \begin{bmatrix} 1 & 0 & 0 \\ \frac{1}{3} & 1 & 0 \\ \frac{2}{3} & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & -\frac{1}{2} & 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 \\ \frac{1}{3} & 1 & 0 \\ \frac{2}{3} & -\frac{1}{2} & 1 \end{bmatrix}.$$

LU decomposition

Using the method showed in the example, we can prove the following theorem.

Theorem

Let A be an $N \times N$ nonsingular matrix. Then there exists a permutation matrix P, an upper triangular matrix U, and a lower triangular matrix L with all 1's in the main diagonal, such that

$$PA = LU.$$

Notice that to calculate the matrices P, U and L we essentially have to do a Gaussian elimination with partial pivoting. In the example we obtained

$$PA = \begin{bmatrix} 0 & 0 & 1 \\ 1 & 0 & 0 \\ 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} 1 & 0 & 1 \\ 2 & 5 & -2 \\ 3 & 6 & 9 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 \\ \frac{1}{3} & 1 & 0 \\ \frac{2}{3} & -\frac{1}{2} & 1 \end{bmatrix} \begin{bmatrix} 3 & 6 & 9 \\ 0 & -2 & -2 \\ 0 & 0 & -9 \end{bmatrix} = LU.$$

Subsection 3

Applications of the Gaussian Elimination and the LU decomposition

Inverse matrix

Let A be an $N \times N$ matrix, and we would like to calculate A^{-1}. Consider the first column of A^{-1}, and denote it by $\mathbf{x}_1$. By definition $AA^{-1} = I_n$, hence

$$Ax_1 = [1, 0, 0, \ldots, 0]^T$$

Thus we can calculate the first column of A^{-1} by solving a system of linear equations with coefficient matrix A.

Similarly, if x_2 is the second column of A^{-1}, then

$$Ax_2 = [0, 1, 0, \ldots, 0]^T$$

As before, x_2 can be calculated by solving a system of linear equations with coefficient matrix A.

We can continue, and so each column of the inverse can be determined.

Inverse matrix

Observe, that we can take advantage of the fact that all systems have the same coefficient matrix A, thus we can perform the Gaussian elimination simoultaneously, as shown in the following example.

Example

Find the inverse of the following matrix.

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

$$\left[\begin{array}{ccc|ccc} 1 & 1 & -2 & 1 & 0 & 0 \\ -2 & -1 & 4 & 0 & 1 & 0 \\ -1 & -1 & 3 & 0 & 0 & 1 \end{array}\right] \overset{II.+2I.,III.+I.}{\sim} \left[\begin{array}{ccc|ccc} 1 & 1 & -2 & 1 & 0 & 0 \\ 0 & 1 & 0 & 2 & 1 & 0 \\ 0 & 0 & 1 & 1 & 0 & 1 \end{array}\right]$$

Inverse matrix

Notice that solving backward can also be performed in the matrix form, we need to eliminate backwards the elements above the diagonal.

$$\left[\begin{array}{ccc|ccc} 1 & 1 & -2 & 1 & 0 & 0 \\ 0 & 1 & 0 & 2 & 1 & 0 \\ 0 & 0 & 1 & 1 & 0 & 1 \end{array}\right] \overset{I.+2III.}{\sim} \left[\begin{array}{ccc|ccc} 1 & 1 & 0 & 3 & 0 & 2 \\ 0 & 1 & 0 & 2 & 1 & 0 \\ 0 & 0 & 1 & 1 & 0 & 1 \end{array}\right] \overset{I.-II.}{\sim}$$

$$\left[\begin{array}{ccc|ccc} 1 & 0 & 0 & 1 & -1 & 2 \\ 0 & 1 & 0 & 2 & 1 & 0 \\ 0 & 0 & 1 & 1 & 0 & 1 \end{array}\right]$$

Observe that what we obtained on the right handside is exactly the desired inverse of A

$$A^{-1} = \left[\begin{array}{ccc} 1 & -1 & 2 \\ 2 & 1 & 0 \\ 1 & 0 & 1 \end{array}\right].$$

Simoultaneous solving

Analyzing the algorithm one can prove the following simple proposition.

Proposition

The inverse (if it exists) of an upper (lower) triangular matrix is upper (lower) triangular.

Note that actually we just solved the matrix equation $AX = I_N$ for the square matrix X as unkonwn. Also we may observe that the role of I_N was unimportant, the same algorithm solves the matrix equation $AX = B$, where A, B are given square matrices, and X is the unknown square matrix.

Systems with the same matrix

Solving the matrix equation $AX = B$ was just a clever way of solving N systems of linear equations via the same coefficient matrix A simoultaneously, at the same.

However it may happen that we need to solve systems of linear equations with the same coefficient matrix one after another. One obvious way to deal with this problem is to calculate the inverse A^{-1}, and then each system can be solved very effectively in just $O(N^2)$ time.

We can do a bit better: when solving the first system we can also calculate the LU decomposition of A „for free”.

Systems with the same matrix

Assume that we already know the LU decomposition of a matrix A, and we would like to solve a system of linear equtions $A\mathbf{x} = \mathbf{b}$.

First we premultiply the equation by P: $PA\mathbf{x} = P\mathbf{b}$, which can be rewritten as $LU\mathbf{x} = P\mathbf{b}$. This can be solved in the form

$$L\mathbf{y} = P\mathbf{b}$$
$$U\mathbf{x} = \mathbf{y}$$

Observe that $L\mathbf{y} = P\mathbf{b}$ can be solved in $O(N^2)$ time by forward substitution, while $U\mathbf{x} = \mathbf{y}$ can be solved in $O(N^2)$ time by backward substitution.

Subsection 4

Banded systems, cubic spline interpolation

Sparse and banded systems

The large linear systems that arise in applications are usually sparse, that is, most of the matrix coefficients are zero. Many of these systems can, by properly ordering the equations and unknowns, be put into banded form, where all elements of the coefficient matrix are zero outside some relatively small band around the main diagonal.

Since zeros play a very important role in the elimination, it is possible to take advantage of the banded property, and one may find faster solving methods than simple Gaussian elimination. We also note here that if A is banded, so are the matrices L and U in its LU decomposition.

We are going to come back to the special algorithms and theoretical background for sparse and banded systems later in the course. Here we present a very important mathematical concept that naturally leads to banded systems.

An application

Definition: Interpolation

We are given the data points $(x_1, y_1), (x_2, y_2), \ldots, (x_N, y_N)$ in $\mathbb{R}^2$. The real function f interpolates to the given data points if $f(x_i) = y_i$ for all $i = 1, \ldots, N$.

Definition: Cubic spline

Let $s\colon (a, b) \subset \mathbb{R} \to \mathbb{R}$ be a function, and let $a = x_0 < x_1 < x_2 < \ldots < x_N < x_{N+1} = b$ be a partition of the interval (a, b). The function $s(x)$ is a cubic spline with respect to the given partition if $s(x)$, $s'(x)$ and $s''(x)$ are continuous on (a, b), and $s(x)$ is a cubic polinomial on each subinterval (x_i, x_{i+1}) $(i = 0, \ldots, N)$.

Cubic spline interpolation

Definition: cubic spline interpolation

We are given the data points $(x_1, y_1), (x_2, y_2), \ldots, (x_N, y_N)$ in $\mathbb{R}^2$. The real function $s\colon (a, b) \subset \mathbb{R} \to \mathbb{R}$ is a cubic spline interpolant to the data points, if $s(x)$ interpolates to the given data points and $s(x)$ is a cubic spline with respect to the partition $a = x_0 < x_1 < x_2 < \ldots < x_N < x_{N+1} = b$.

Remark. Usually we restrict the function $s(x)$ to $[x_1, x_N]$.

Cubic spline interpolations are interesting from both theoretical and practical point of view.

Calculating cubic spline interpolations

Proposition

The cubic polynomial

$$s_i(x) = y_i + \left[\frac{y_{i+1} - y_i}{x_{i+1} - x_i} - \frac{(x_{i+1} - x_i)(2\sigma_i + \sigma_{i+1})}{6}\right](x - x_i) + \frac{\sigma_i}{2}(x - x_i)^2 + \frac{\sigma_{i+1} - \sigma_i}{6(x_{i+1} - x_i)}(x - x_i)^3 \tag{1}$$

satisfies $s_i(x_i) = y_i$, $s_i(x_{i+1}) = y_{i+1}$, $s_i''(x_i) = \sigma_i$, and $s_i''(x_{i+1}) = \sigma_{i+1}$.

Thus, if we prescribe the value of the second derivative at x_i to be σ_i for all $i = 1, \ldots, N$, then we have a unique candidate for the cubic spline interpolant. It remains to ensure that the first derivative is continuous at the points $x_2, x_3, \ldots, x_{N-1}$.

Calculating cubic spline interpolations

Proposition

We define $s(x)$ on $[x_i, x_{i+1}]$ to be $s_i(x)$ from (1) ($i = 1, \ldots, N-1$). The first derivative $s'(x)$ is continuous on (x_1, x_N) if, and only if,

$$\begin{bmatrix} \frac{h_1+h_2}{3} & \frac{h_2}{6} & 0 & 0 & \ldots & 0 & 0 \\ \vdots & \ddots & & & & & \\ 0 & \ldots & \frac{h_i}{6} & \frac{h_i+h_{i+1}}{3} & \frac{h_{i+1}}{6} & \ldots & 0 \\ & & & & \ddots & & \vdots \\ 0 & 0 & \ldots & 0 & 0 & \frac{h_{N-2}}{6} & \frac{h_{N-2}+h_{N-1}}{3} \end{bmatrix} \begin{bmatrix} \sigma_2 \\ \vdots \\ \sigma_{i+1} \\ \vdots \\ \sigma_{N-1} \end{bmatrix} = \begin{bmatrix} r_1 - \frac{h_1}{6}\sigma_1 \\ \vdots \\ r_i \\ \vdots \\ r_{N-2} - \frac{h_{N-1}}{6}\sigma_N \end{bmatrix}$$

where $h_i = x_{i+1} - x_i$ and $r_i = (y_{i+2} - y_{i+1})/h_{i+1} - (y_{i+1} - y_i)/h_i$.

The cubic spline interpolation problem leads to a tridiagonal system of linear equations.

Cubic spline interpolations

Definition

If we set $\sigma_1 = \sigma_N = 0$, and there exists a unique cubic spline interpolant, then it is called the natural cubic spline interpolant.

The following theorem intuitively shows that natural cubic spline interpolant are the „least curved" interpolants.

Theorem

Among all functions that are continuous, with continuous first and second derivatives, which interpolate to the data points (x_i, y_i), $i = 1, \ldots, N$, the natural cubic spline interpolant $s(x)$ minimizes

$$\int_{x_1}^{x_N} [s''(x)]^2 dx.$$

Section 3

Least Square Problems

Subsection 1

Under and overdetermined systems

Problem setting

1. Let A be a matrix with n columns and m rows, and $\boldsymbol{b}$ be an m-dimensional column vector.
2. The system

$$A\boldsymbol{x} = \boldsymbol{b} \tag{2}$$

   of linear equations has m equations in n indeterminates.
3. (2) has a **unique solution** only if $n = m$, that is, if A is square. (And, then only if A is nonsingular.)
4. (2) may have (a) **infinitely many** solutions, or (b) **no solution at all.**
5. In case (a), we may be interested in *small* solutions: solutions with least 2-norms.
6. In case (b), we may be happy to find a vector $\boldsymbol{x}$ which *nearly* solves (2), that is, where $A\boldsymbol{x} - \boldsymbol{b}$ has least 2-norm.

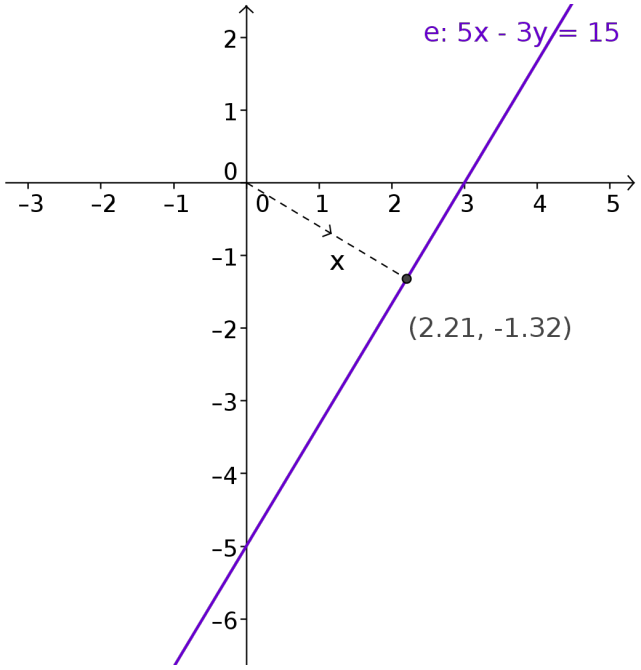
Least square problems for underdetermined systems: $m < n$

Assume that the number of equations is **less** than the number of indeterminates. Then:

1. We say that the system of linear equations is underdetermined.
2. The matrix A is *horizontally* stretched.
3. There are usually **infinitely many** solutions.
4. The problem we want to solve is

$$\text{minimize } \|\boldsymbol{x}\|_2 \text{ such that } A\boldsymbol{x} = \boldsymbol{b}. \qquad (3)$$

5. Example: Minimize $\sqrt{x^2 + y^2}$ such that $5x - 3y = 15$.



Least square problems for overdetermined systems: $m > n$

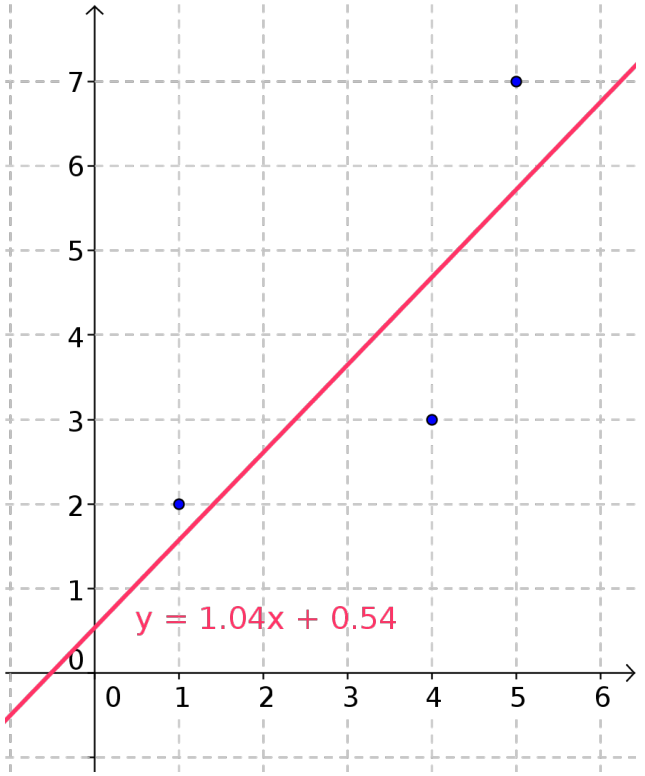
Assume that the number of equations is **more** than the number of indeterminates. Then:

1. We say that the system of linear equations is overdetermined.
2. The matrix A is *vertically* stretched.
3. There is usually **no solution.**
4. The problem we want to solve is

$$\text{minimize } \|A\boldsymbol{x} - \boldsymbol{b}\|_2. \tag{4}$$

5. Example: The *fitting line* to the points $(1, 2), (4, 3), (5, 7)$:

$$2 = m + b \tag{5}$$
$$3 = 4m + b \tag{6}$$
$$7 = 5m + b \tag{7}$$



Solution of overdetermined systems

Theorem

(a) The vector $\boldsymbol{x}$ solves the problem

$$\text{minimize } \|A\boldsymbol{x} - \boldsymbol{b}\|_2$$

if and only if $A^T A\boldsymbol{x} = A^T\boldsymbol{b}$.

(b) The system $A^T A\boldsymbol{x} = A^T\boldsymbol{b}$ has always a solution which is unique if the columns of A are linearly independent.

Proof. (a) Assume $A^T A\boldsymbol{x} = A^T\boldsymbol{b}$. Let $\boldsymbol{y}$ be arbitrary and $\boldsymbol{e} = \boldsymbol{y} - \boldsymbol{x}$.

$$\begin{aligned}
\|A(\boldsymbol{x}+\boldsymbol{e}) - \boldsymbol{b}\|_2^2 &= (A(\boldsymbol{x}+\boldsymbol{e}) - \boldsymbol{b})^T(A(\boldsymbol{x}+\boldsymbol{e}) - \boldsymbol{b}) \\
&= (A\boldsymbol{x} - \boldsymbol{b})^T(A\boldsymbol{x} - \boldsymbol{b}) + 2(A\boldsymbol{e})^T(A\boldsymbol{x} - \boldsymbol{b}) + (A\boldsymbol{e})^T(A\boldsymbol{e}) \\
&= \|A\boldsymbol{x} - \boldsymbol{b}\|_2^2 + \|A\boldsymbol{e}\|_2^2 + 2\boldsymbol{e}^T(A^T A\boldsymbol{x} - A^T\boldsymbol{b}) \\
&= \|A\boldsymbol{x} - \boldsymbol{b}\|_2^2 + \|A\boldsymbol{e}\|_2^2 \geq \|A\boldsymbol{x} - \boldsymbol{b}\|_2^2
\end{aligned}$$

This implies $\|A\boldsymbol{x} - \boldsymbol{b}\|_2$ to be minimal. For the converse, observe that if $A^T A\boldsymbol{x} \neq A^T\boldsymbol{b}$ then there is a vector $\boldsymbol{e}$ such that $\boldsymbol{e}^T(A^T A\boldsymbol{x} - A^T\boldsymbol{b}) > 0$.

Solution of overdetermined systems (cont.)

(b) Write

$$U = \{A^T A\boldsymbol{x} \mid \boldsymbol{x} \in \mathbb{R}^n\} \quad \text{and} \quad V = U^\perp = \{\boldsymbol{v} \mid \boldsymbol{v}^T\boldsymbol{u} = 0 \text{ for all } \boldsymbol{u} \in U\}.$$

By definition,

$$0 = \boldsymbol{v}^T A^T A\boldsymbol{v} = \|A\boldsymbol{v}\|_2^2$$

for any $\boldsymbol{v} \in V$, hence $A\boldsymbol{v} = \boldsymbol{0}$.
Thus, for any $\boldsymbol{v} \in V$,

$$(A^T\boldsymbol{b})^T\boldsymbol{v} = \boldsymbol{b}^T A\boldsymbol{v} = 0,$$

which implies $A^T\boldsymbol{b} \in V^\perp$.
A nontrivial fact of finite dimensional vector spaces is

$$V^\perp = U^{\perp\perp} = U.$$

Therefore $A^T\boldsymbol{b} \in U$, which shows the existence in (b).
The uniqueness follows from the observation if the columns of A are linearly independent then $\|A\boldsymbol{e}\|_2 = 0$ implies $A\boldsymbol{e} = \boldsymbol{0}$ and $\boldsymbol{e} = \boldsymbol{0}$. □

Example: Line fitting

Write the line fitting problem (5) as $A\boldsymbol{x} = \boldsymbol{b}$ with

$$A = \begin{bmatrix} 1 & 1 \\ 4 & 1 \\ 5 & 1 \end{bmatrix}, \quad \boldsymbol{x} = \begin{bmatrix} m \\ b \end{bmatrix}, \quad \boldsymbol{b} = \begin{bmatrix} 2 \\ 3 \\ 7 \end{bmatrix}.$$

Then

$$A^T A = \begin{bmatrix} 42 & 10 \\ 10 & 3 \end{bmatrix}, \quad A^T \boldsymbol{b} = \begin{bmatrix} 49 \\ 12 \end{bmatrix}.$$

The solution of

$$\begin{aligned} 42m + 10b &= 49 \\ 10m + 3b &= 12 \end{aligned}$$

is $m = 27/26 \approx 1.0385$ and $b = 7/13 \approx 0.5385$. Hence, the equation of the fitting line is

$$y = 1.0385\,x + 0.5385.$$

Solution of underdetermined systems

Theorem

(a) If $AA^T\boldsymbol{z} = \boldsymbol{b}$ and $\boldsymbol{x} = A^T\boldsymbol{z}$ then $\boldsymbol{x}$ is the unique solution for the underdetermined problem

$$\text{minimize } \|\boldsymbol{x}\|_2 \text{ such that } A\boldsymbol{x} = \boldsymbol{b}.$$

(b) The system $AA^T\boldsymbol{z} = \boldsymbol{b}$ has always a solution.

Proof. (a) Assume $AA^T\boldsymbol{z} = \boldsymbol{b}$ and $\boldsymbol{x} = A^T\boldsymbol{z}$. Let $\boldsymbol{y}$ be such that $A\boldsymbol{y} = \boldsymbol{b}$ and write $\boldsymbol{e} = \boldsymbol{y} - \boldsymbol{x}$. We have

$$A(\boldsymbol{x} + \boldsymbol{e}) = A\boldsymbol{x} = \boldsymbol{b} \Rightarrow A\boldsymbol{e} = \boldsymbol{0} \Rightarrow \boldsymbol{x}^T\boldsymbol{e} = (A^T\boldsymbol{z})^T\boldsymbol{e} = \boldsymbol{z}^T(A\boldsymbol{e}) = 0.$$

Then,

$$\|\boldsymbol{x} + \boldsymbol{e}\|_2^2 = (\boldsymbol{x} + \boldsymbol{e})^T(\boldsymbol{x} + \boldsymbol{e}) = \boldsymbol{x}^T\boldsymbol{x} + 2\boldsymbol{x}^T\boldsymbol{e} + \boldsymbol{e}^T\boldsymbol{e} = \|\boldsymbol{x}\|_2^2 + \|\boldsymbol{e}\|_2^2,$$

which implies $\|\boldsymbol{y}\|_2 \geq \|\boldsymbol{x}\|_2$ and equality holds if and only if $\boldsymbol{y} - \boldsymbol{x} = \boldsymbol{e} = \boldsymbol{0}$.

Solution of underdetermined systems (cont.)

(b) Write

$$U = \{AA^T z \mid z \in \mathbb{R}^n\} \quad \text{and} \quad V = U^\perp = \{v \mid v^T u = 0 \text{ for all } u \in U\}.$$

By definition,

$$0 = v^T AA^T v = \|A^T v\|_2^2$$

for any $v \in V$, hence $A^T v = \mathbf{0}$. As the system is underdetermined, there is a vector x_0 such that $Ax_0 = b$. Hence, for any $v \in V$

$$b^T v = (Ax_0)^T v = x_0^T (A^T v) = 0.$$

This means $b \in V^\perp = U^{\perp\perp} = U$, that is, $b = AA^T z$ for some vector z. □

Example: Closest point on a line

We want to minimize $\sqrt{x^2 + y^2}$ for the points of the line $5x - 3y = 15$. Then

$$A = \begin{bmatrix} 5 & -3 \end{bmatrix}, \quad \boldsymbol{x} = \begin{bmatrix} x \\ y \end{bmatrix}, \quad \boldsymbol{b} = \begin{bmatrix} 15 \end{bmatrix}.$$

Moreover, $AA^T = [34]$ and the solution of $AA^T\boldsymbol{z} = \boldsymbol{b}$ is $z = 15/34$. Thus, the optimum is

$$\begin{bmatrix} x \\ y \end{bmatrix} = A^T\boldsymbol{z} = \begin{bmatrix} 5 \\ -3 \end{bmatrix} \cdot \frac{15}{34} \approx \begin{bmatrix} 2.2059 \\ -1.3235 \end{bmatrix}.$$

Implementation and numerical stability questions

- The theorems above are important theoretical results for the solution of under- and overdetermined systems.
- In the practical applications, there are **two larger problems.**
- (1) Both methods require the solution of systems ($A^TA\boldsymbol{x} = A^T\boldsymbol{b}$ and $AA^T\boldsymbol{z} = \boldsymbol{b}$) which are well-determined but still may be singular.
- Many linear solvers have problems in dealing with such systems.
- (2) The numerical values in AA^T and A^TA have typically double length compared to the values in A.
- This may cause numerical instability.

Subsection 2

The QR decomposition

Orthogonal vectors, orthogonal matrices

We say that

1. the vectors $\boldsymbol{u}, \boldsymbol{v}$ are orthogonal, if their scalar product $\boldsymbol{u}^T\boldsymbol{v} = 0$.
2. the vector $\boldsymbol{v}$ is normed to 1, if its 2-norm is 1: $\|\boldsymbol{v}\|_2^2 = \boldsymbol{v}^T\boldsymbol{v} = 1$.
3. the vectors $\boldsymbol{v}_1, \ldots, \boldsymbol{v}_k$ are orthogonal, if they are pairwise orthogonal.
4. the vectors $\boldsymbol{v}_1, \ldots, \boldsymbol{v}_k$ form an orthonormal system if they are pairwise orthogonal and normed to 1.
5. the $n \times n$ matrix A is orthogonal, if $A^TA = AA^T = I$, where I is the $n \times n$ unit matrix.

Examples:

- The vectors $[1, -1, 0]$, $[1, 1, 1]$ and $[-3, -3, 6]$ are orthogonal.
- For any $\varphi \in \mathbb{R}$, the matrix

$$\begin{bmatrix} \cos\varphi & -\sin\varphi \\ \sin\varphi & \cos\varphi \end{bmatrix}$$

is orthogonal.

Properties of orthogonal matrices

- If $\boldsymbol{a}_i$ denotes the ith column of A, then
  the (i,j)-entry of A^TA is the scalar product $\boldsymbol{a}_i^T\boldsymbol{a}_j$.
- If $\boldsymbol{b}_j$ denotes the jth row of A, then
  the (i,j)-entry of AA^T is the scalar product $\boldsymbol{b}_i\boldsymbol{b}_j^T$.

Proposition: Rows and columns of orthogonal matrices

For an $n \times n$ matrix A the following are equivalent:

1. A is orthogonal.
2. The columns of A form an orthonormal system.
3. The rows of A form an orthonormal system.

Proposition: Orthogonal matrices preserve scalar product and 2-norm

Let Q be an orthogonal matrix. Then $(Q\boldsymbol{u})^T(Q\boldsymbol{v}) = \boldsymbol{u}^T\boldsymbol{v}$ and $\|Q\boldsymbol{u}\|_2 = \|\boldsymbol{u}\|_2$.

Proof. $(Q\boldsymbol{u})^T(Q\boldsymbol{v}) = u^TQ^TQ\boldsymbol{v} = \boldsymbol{u}^TI\boldsymbol{v} = \boldsymbol{u}^T\boldsymbol{v}$. □

Row echelon form

- The leading entry of a nonzero (row or column) vector is its first nonzero element.
- We say that the matrix $A = (a_{ij})$ is in row echelon form if the **leading entry of a nonzero row is always strictly to the right of the leading entry of the row above it.**
- Example:

$$\begin{bmatrix} 1 & a_{12} & a_{13} & a_{14} & a_{15} \\ 0 & 0 & 2 & a_{24} & a_{25} \\ 0 & 0 & 0 & -1 & a_{35} \\ 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

- In row echelon form, the last rows of the matrix may be all-zeros.
- Using Gaussian elimination, any matrix can be transformed into row echelon form by elementary row operations.
- Similarly, we can speak of matrices in column echelon form.

Overdetermined systems in row echelon form

1. Let A be an $m \times n$ matrix in row echelon form such that the last $m - k$ rows are all-zero and consider the system

$$\begin{bmatrix} a_{11} & a_{12} & a_{13} & \cdots & a_{1\ell} & \cdots & a_{1n} \\ 0 & 0 & a_{22} & \cdots & a_{2\ell} & \cdots & a_{2n} \\ \vdots & & & & \vdots & & \vdots \\ 0 & 0 & 0 & \cdots & a_{k\ell} & \cdots & a_{kn} \\ 0 & 0 & 0 & \cdots & 0 & \cdots & 0 \\ \vdots & & & & \vdots & & \vdots \\ 0 & 0 & 0 & \cdots & 0 & \cdots & 0 \end{bmatrix} \begin{bmatrix} x_1 \\ \vdots \\ x_\ell \\ \vdots \\ x_n \end{bmatrix} = \begin{bmatrix} b_1 \\ b_2 \\ \vdots \\ b_k \\ b_{k+1} \\ \vdots \\ b_m \end{bmatrix}$$

   of m equations in n variables.
2. As the leading entries in the first k rows are nonzero, there are values $x_1, \ldots, x_n$ such that the first k equations hold.
3. However, by any choice of the variables, the *error* in equations $k+1, \ldots, m$ is $b_{k+1}, \ldots, b_m$.
4. The minimum of $\|A\boldsymbol{x} - \boldsymbol{b}\|_2$ is $\sqrt{b_{k+1}^2 + \ldots + b_m^2}$.

The QR decomposition

Definition

We say that $A = QR$ is a QR decomposition of A, if A is an $m \times n$ matrix, Q is an $m \times m$ orthogonal matrix and R is an $m \times n$ matrix in row echelon form.

We will partly prove the following important result later.

Theorem: Existence of QR decompositions

Any real matrix A has a QR decomposition $A = QR$. If A is nonsingular then Q is unique up to the signs of its columns.

Example: Let $A = \begin{bmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{bmatrix}$ and assume that the first column is nonzero.

Define $c = \frac{a_{11}}{\sqrt{a_{11}^2+a_{21}^2}}, s = \frac{a_{21}}{\sqrt{a_{11}^2+a_{21}^2}}$. Then $A = QR$ is a QR decomposition with

$$Q = \begin{bmatrix} c & -s \\ s & c \end{bmatrix}, \quad R = \begin{bmatrix} \sqrt{a_{11}^2 + a_{21}^2} & \frac{a_{11}a_{12}+a_{21}a_{22}}{\sqrt{a_{11}^2+a_{21}^2}} \\ 0 & \frac{a_{11}a_{22}-a_{12}a_{21}}{\sqrt{a_{11}^2+a_{21}^2}} \end{bmatrix}.$$

Solving overdetermined systems with QR decomposition

We can solve the underdetermined system

$$\text{minimize } ||A\boldsymbol{x} - \boldsymbol{b}||_2$$

in the following way.

1. Let $A = QR$ be a QR-decomposition of A.
2. Put $\boldsymbol{c} = Q^T\boldsymbol{b}$.
3. Using the fact that the orthogonal matrix Q^T preserves the 2-norm, we have

$$||A\boldsymbol{x} - \boldsymbol{b}||_2 = ||QR\boldsymbol{x} - \boldsymbol{b}||_2 = ||Q^TQR\boldsymbol{x} - Q^T\boldsymbol{b}||_2 = ||R\boldsymbol{x} - \boldsymbol{c}||_2.$$

4. Since R has row echelon form, the underdetermined system

$$\text{minimize } ||R\boldsymbol{x} - \boldsymbol{c}||_2$$

can be solved in an obvious manner as explained before.

QR decomposition and orthogonalization

- Let A be a nonsingular matrix and $A = QR$ its QR decomposition.
- Let $\boldsymbol{a}_1, \ldots, \boldsymbol{a}_n$ be the column vectors of A, $\boldsymbol{q}_1, \ldots, \boldsymbol{q}_n$ the column vectors of Q and $R = (c_{ij})$ upper triangular.

$$\begin{bmatrix} \boldsymbol{a}_1 & \boldsymbol{a}_2 & \cdots & \boldsymbol{a}_n \end{bmatrix} = \begin{bmatrix} \boldsymbol{q}_1 & \boldsymbol{q}_2 & \cdots & \boldsymbol{q}_n \end{bmatrix} \begin{bmatrix} c_{11} & c_{12} & \cdots & c_{1n} \\ 0 & c_{22} & \cdots & c_{2n} \\ \vdots & & & \\ 0 & 0 & \cdots & c_{nn} \end{bmatrix}$$

$$= \begin{bmatrix} c_{11}\boldsymbol{q}_1 & c_{12}\boldsymbol{q}_1 + c_{22}\boldsymbol{q}_2 & \cdots & c_{1n}\boldsymbol{q}_1 + c_{2n}\boldsymbol{q}_2 + \cdots + c_{nn}\boldsymbol{q}_n \end{bmatrix}$$

Equivalently with $A = QR$:

$$\begin{cases} \boldsymbol{a}_1 = c_{11}\boldsymbol{q}_1 \\ \boldsymbol{a}_2 = c_{12}\boldsymbol{q}_1 + c_{22}\boldsymbol{q}_2 \\ \quad \vdots \\ \boldsymbol{a}_n = c_{1n}\boldsymbol{q}_1 + c_{2n}\boldsymbol{q}_2 + \cdots + c_{nn}\boldsymbol{q}_n \end{cases} \tag{8}$$

QR decomposition and orthogonalization (cont.)

- Since A is nonsingular, R is nonsingular and $c_{11} \cdots c_{nn} \neq 0$.
- We can therefore „solve" (8) for the $\boldsymbol{q}_i$'s by back substitution:

$$\begin{cases} \boldsymbol{q}_1 = d_{11}\boldsymbol{a}_1 \\ \boldsymbol{q}_2 = d_{12}\boldsymbol{a}_1 + d_{22}\boldsymbol{a}_2 \\ \quad \vdots \\ \boldsymbol{q}_n = d_{1n}\boldsymbol{a}_1 + d_{2n}\boldsymbol{a}_2 + \cdots + d_{nn}\boldsymbol{a}_n \end{cases} \tag{9}$$

- Thus, the QR decomposition of a nonsingular matrix is equivalent with transforming the $\boldsymbol{a}_i$'s into an orthonormal system as in (9).
- Transformations as in (9) are called orthogonalizations.
- The orthogonalization process consists of **two steps:**
- (1) [hard] **Transforming** the $\boldsymbol{a}_i$'s into an orthogonal system.
- (2) [easy] **Norming** the vectors to 1: $\boldsymbol{q}_i' = \dfrac{\boldsymbol{q}_i}{\|\boldsymbol{q}_i\|_2}$.

The orthogonal projection

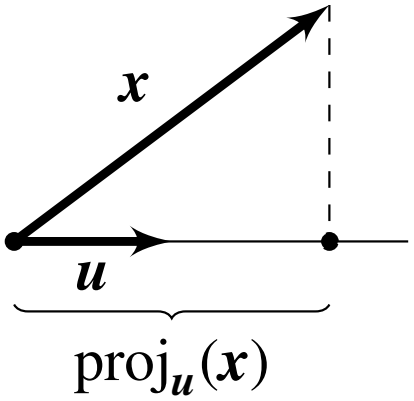
- For vectors $\boldsymbol{u}, \boldsymbol{x}$ define the map

$$\text{proj}_{\boldsymbol{u}}(\boldsymbol{x}) = \frac{\boldsymbol{u}^T\boldsymbol{x}}{\boldsymbol{u}^T\boldsymbol{u}}\boldsymbol{u}.$$

- On the one hand, the vectors $\boldsymbol{u}$ and $\text{proj}_{\boldsymbol{u}}(\boldsymbol{x})$ are parallel.
- On the other hand, $\boldsymbol{u}$ is perpendicular to $\boldsymbol{x} - \text{proj}_{\boldsymbol{u}}(\boldsymbol{x})$:

$$\boldsymbol{u}^T(\boldsymbol{x} - \text{proj}_{\boldsymbol{u}}(\boldsymbol{x})) = \boldsymbol{u}^T\boldsymbol{x} - \boldsymbol{u}^T\left(\frac{\boldsymbol{u}^T\boldsymbol{x}}{\boldsymbol{u}^T\boldsymbol{u}}\boldsymbol{u}\right) = \boldsymbol{u}^T\boldsymbol{x} - \left(\frac{\boldsymbol{u}^T\boldsymbol{x}}{\boldsymbol{u}^T\boldsymbol{u}}\right)(\boldsymbol{u}^T\boldsymbol{u}) = 0.$$

- This means that $\text{proj}_{\boldsymbol{u}}(\boldsymbol{x})$ is the orthogonal projection of the vector $\boldsymbol{x}$ to the 1-dimensional subspace spanned by $\boldsymbol{u}$:



The Gram-Schmidt orthogonalization

The Gram–Schmidt process works as follows. We are given the vectors $\boldsymbol{a}_1, \ldots, \boldsymbol{a}_n$ in $\mathbb{R}^n$ and define the vectors $\boldsymbol{q}_1, \ldots, \boldsymbol{q}_n$ recursively:

$$\begin{cases} \boldsymbol{q}_1 = \boldsymbol{a}_1 \\ \boldsymbol{q}_2 = \boldsymbol{a}_2 - \operatorname{proj}_{\boldsymbol{q}_1}(\boldsymbol{a}_2) \\ \boldsymbol{q}_3 = \boldsymbol{a}_3 - \operatorname{proj}_{\boldsymbol{q}_1}(\boldsymbol{a}_3) - \operatorname{proj}_{\boldsymbol{q}_2}(\boldsymbol{a}_3) \\ \quad \vdots \\ \boldsymbol{q}_n = \boldsymbol{a}_n - \operatorname{proj}_{\boldsymbol{q}_1}(\boldsymbol{a}_n) - \operatorname{proj}_{\boldsymbol{q}_2}(\boldsymbol{a}_n) - \cdots - \operatorname{proj}_{\boldsymbol{q}_{n-1}}(\boldsymbol{a}_n) \end{cases} \tag{10}$$

On the one hand, the $\boldsymbol{q}_i$'s are orthogonal. For example, we show that $\boldsymbol{q}_2$ is orthogonal to $\boldsymbol{q}_5$ by assuming that we have already shown $\boldsymbol{q}_2 \perp \boldsymbol{q}_1, \boldsymbol{q}_3, \boldsymbol{q}_4$.
Then, $\boldsymbol{q}_2 \perp \operatorname{proj}_{\boldsymbol{q}_1}(\boldsymbol{a}_5), \operatorname{proj}_{\boldsymbol{q}_3}(\boldsymbol{a}_5), \operatorname{proj}_{\boldsymbol{q}_4}(\boldsymbol{a}_5)$ too, since these are scalar multiples of the respective $\boldsymbol{q}_i$'s.
As before, we have $\boldsymbol{q}_2 \perp \boldsymbol{a}_5 - \operatorname{proj}_{\boldsymbol{q}_2}(\boldsymbol{a}_5)$. Therefore,

$$\boldsymbol{q}_2 \perp \boldsymbol{a}_5 - \operatorname{proj}_{\boldsymbol{q}_1}(\boldsymbol{a}_5) - \operatorname{proj}_{\boldsymbol{q}_2}(\boldsymbol{a}_5) - \operatorname{proj}_{\boldsymbol{q}_3}(\boldsymbol{a}_5) - \operatorname{proj}_{\boldsymbol{q}_4}(\boldsymbol{a}_5) = \boldsymbol{q}_5.$$

The Gram-Schmidt orthogonalization (cont.)

- On the other hand, we have to make clear that any $\boldsymbol{a}_i$ is a linear combination of $\boldsymbol{q}_1, \boldsymbol{q}_2, \ldots, \boldsymbol{q}_i$ as required in (8).
- Notice that (10) implies

$$\begin{aligned} \boldsymbol{a}_i &= \text{proj}_{\boldsymbol{q}_1}(\boldsymbol{a}_i) + \text{proj}_{\boldsymbol{q}_2}(\boldsymbol{a}_i) + \cdots + \text{proj}_{\boldsymbol{q}_{i-1}}(\boldsymbol{a}_i) + \boldsymbol{q}_i \\ &= c_{1i}\boldsymbol{q}_1 + c_{2i}\boldsymbol{q}_2 + \cdots + c_{i-1,i}\boldsymbol{q}_{i-1} + \boldsymbol{q}_i. \end{aligned} \tag{11}$$

- The coefficients $c_{ij} = (\boldsymbol{q}_i^T \boldsymbol{a}_j)/(\boldsymbol{q}_i^T \boldsymbol{q}_i)$ are well defined if and only if $\boldsymbol{q}_i \neq \boldsymbol{0}$.
- However, $\boldsymbol{q}_i \neq \boldsymbol{0}$ means that $\boldsymbol{a}_1, \boldsymbol{a}_2, \ldots, \boldsymbol{a}_i$ are linearly dependent, contradicting the fact that A is nonsingular.
- This proves that (10) indeed results an orthogonalization of the system $\boldsymbol{a}_1, \boldsymbol{a}_2, \ldots, \boldsymbol{a}_n$.

The Gram-Schmidt process

1. **Initialization:** Copy all a_i's to the q_i's.
2. **Step 1:** Finalize q_1 and subtract $\text{proj}_{q_1}(a_2), \text{proj}_{q_1}(a_3), \ldots$ from $q_2, q_3, \ldots$.
3. **Step 2:** Finalize q_2 and subtract $\text{proj}_{q_2}(a_3), \text{proj}_{q_2}(a_4), \ldots$ from $q_3, q_4, \ldots$.
4. *and so on...*
5. **Step n:** Finalize q_{n-1} and substract $\text{proj}_{q_{n-1}}(a_n)$ from q_n.
6. **Last step:** Finalize q_n and **quit.**

$$
\begin{aligned}
q_1 &\leftarrow a_1 \\
q_2 &\leftarrow a_2 - \text{proj}_{q_1}(a_2) \\
q_3 &\leftarrow a_3 - \text{proj}_{q_1}(a_3) - \text{proj}_{q_2}(a_3) \\
&\vdots \\
q_n &\leftarrow a_n - \text{proj}_{q_1}(a_n) - \text{proj}_{q_2}(a_n) - \cdots - \text{proj}_{q_{n-1}}(a_n)
\end{aligned}
$$

The Gram-Schmidt process

1. **Initialization:** Copy all $\boldsymbol{a}_i$'s to the $\boldsymbol{q}_i$'s.
2. **Step 1:** Finalize $\boldsymbol{q}_1$ and subtract $\text{proj}_{\boldsymbol{q}_1}(\boldsymbol{a}_2), \text{proj}_{\boldsymbol{q}_1}(\boldsymbol{a}_3), \ldots$ from $\boldsymbol{q}_2, \boldsymbol{q}_3, \ldots$.
3. **Step 2:** Finalize $\boldsymbol{q}_2$ and subtract $\text{proj}_{\boldsymbol{q}_2}(\boldsymbol{a}_3), \text{proj}_{\boldsymbol{q}_2}(\boldsymbol{a}_4), \ldots$ from $\boldsymbol{q}_3, \boldsymbol{q}_4, \ldots$.
4. *and so on...*
5. **Step n:** Finalize $\boldsymbol{q}_{n-1}$ and substract $\text{proj}_{\boldsymbol{q}_{n-1}}(\boldsymbol{a}_n)$ from $\boldsymbol{q}_n$.
6. **Last step:** Finalize $\boldsymbol{q}_n$ and **quit.**

$$
\begin{aligned}
\boldsymbol{q}_1 &\leftarrow \boldsymbol{a}_1 \\
\boldsymbol{q}_2 &\leftarrow \boldsymbol{a}_2 - \text{proj}_{\boldsymbol{q}_1}(\boldsymbol{a}_2) \\
\boldsymbol{q}_3 &\leftarrow \boldsymbol{a}_3 - \text{proj}_{\boldsymbol{q}_1}(\boldsymbol{a}_3) - \text{proj}_{\boldsymbol{q}_2}(\boldsymbol{a}_3) \\
&\vdots \\
\boldsymbol{q}_n &\leftarrow \boldsymbol{a}_n - \text{proj}_{\boldsymbol{q}_1}(\boldsymbol{a}_n) - \text{proj}_{\boldsymbol{q}_2}(\boldsymbol{a}_n) - \cdots - \text{proj}_{\boldsymbol{q}_{n-1}}(\boldsymbol{a}_n)
\end{aligned}
$$

The Gram-Schmidt process

1. **Initialization:** Copy all $\boldsymbol{a}_i$'s to the $\boldsymbol{q}_i$'s.
2. **Step 1:** Finalize $\boldsymbol{q}_1$ and subtract $\text{proj}_{\boldsymbol{q}_1}(\boldsymbol{a}_2), \text{proj}_{\boldsymbol{q}_1}(\boldsymbol{a}_3), \ldots$ from $\boldsymbol{q}_2, \boldsymbol{q}_3, \ldots$.
3. **Step 2:** Finalize $\boldsymbol{q}_2$ and subtract $\text{proj}_{\boldsymbol{q}_2}(\boldsymbol{a}_3), \text{proj}_{\boldsymbol{q}_2}(\boldsymbol{a}_4), \ldots$ from $\boldsymbol{q}_3, \boldsymbol{q}_4, \ldots$.
4. *and so on...*
5. **Step n:** Finalize $\boldsymbol{q}_{n-1}$ and substract $\text{proj}_{\boldsymbol{q}_{n-1}}(\boldsymbol{a}_n)$ from $\boldsymbol{q}_n$.
6. **Last step:** Finalize $\boldsymbol{q}_n$ and **quit.**

$$
\begin{aligned}
\boldsymbol{q}_1 &\leftarrow \boldsymbol{a}_1 \\
\boldsymbol{q}_2 &\leftarrow \boldsymbol{a}_2 - \text{proj}_{\boldsymbol{q}_1}(\boldsymbol{a}_2) \\
\boldsymbol{q}_3 &\leftarrow \boldsymbol{a}_3 - \text{proj}_{\boldsymbol{q}_1}(\boldsymbol{a}_3) - \text{proj}_{\boldsymbol{q}_2}(\boldsymbol{a}_3) \\
&\vdots \\
\boldsymbol{q}_n &\leftarrow \boldsymbol{a}_n - \text{proj}_{\boldsymbol{q}_1}(\boldsymbol{a}_n) - \text{proj}_{\boldsymbol{q}_2}(\boldsymbol{a}_n) - \cdots - \text{proj}_{\boldsymbol{q}_{n-1}}(\boldsymbol{a}_n)
\end{aligned}
$$

The Gram-Schmidt process

1. **Initialization:** Copy all $\boldsymbol{a}_i$'s to the $\boldsymbol{q}_i$'s.
2. **Step 1:** Finalize $\boldsymbol{q}_1$ and subtract $\text{proj}_{\boldsymbol{q}_1}(\boldsymbol{a}_2), \text{proj}_{\boldsymbol{q}_1}(\boldsymbol{a}_3), \ldots$ from $\boldsymbol{q}_2, \boldsymbol{q}_3, \ldots$.
3. **Step 2:** Finalize $\boldsymbol{q}_2$ and subtract $\text{proj}_{\boldsymbol{q}_2}(\boldsymbol{a}_3), \text{proj}_{\boldsymbol{q}_2}(\boldsymbol{a}_4), \ldots$ from $\boldsymbol{q}_3, \boldsymbol{q}_4, \ldots$.
4. *and so on...*
5. **Step n:** Finalize $\boldsymbol{q}_{n-1}$ and substract $\text{proj}_{\boldsymbol{q}_{n-1}}(\boldsymbol{a}_n)$ from $\boldsymbol{q}_n$.
6. **Last step:** Finalize $\boldsymbol{q}_n$ and **quit.**

$$
\begin{aligned}
\boldsymbol{q}_1 &\leftarrow \boldsymbol{a}_1 \\
\boldsymbol{q}_2 &\leftarrow \boldsymbol{a}_2 - \text{proj}_{\boldsymbol{q}_1}(\boldsymbol{a}_2) \\
\boldsymbol{q}_3 &\leftarrow \boldsymbol{a}_3 - \text{proj}_{\boldsymbol{q}_1}(\boldsymbol{a}_3) - \text{proj}_{\boldsymbol{q}_2}(\boldsymbol{a}_3) \\
&\vdots \\
\boldsymbol{q}_n &\leftarrow \boldsymbol{a}_n - \text{proj}_{\boldsymbol{q}_1}(\boldsymbol{a}_n) - \text{proj}_{\boldsymbol{q}_2}(\boldsymbol{a}_n) - \cdots - \text{proj}_{\boldsymbol{q}_{n-1}}(\boldsymbol{a}_n)
\end{aligned}
$$

The Gram-Schmidt process

1. **Initialization:** Copy all $\boldsymbol{a}_i$'s to the $\boldsymbol{q}_i$'s.
2. **Step 1:** Finalize $\boldsymbol{q}_1$ and subtract $\text{proj}_{\boldsymbol{q}_1}(\boldsymbol{a}_2), \text{proj}_{\boldsymbol{q}_1}(\boldsymbol{a}_3), \ldots$ from $\boldsymbol{q}_2, \boldsymbol{q}_3, \ldots$.
3. **Step 2:** Finalize $\boldsymbol{q}_2$ and subtract $\text{proj}_{\boldsymbol{q}_2}(\boldsymbol{a}_3), \text{proj}_{\boldsymbol{q}_2}(\boldsymbol{a}_4), \ldots$ from $\boldsymbol{q}_3, \boldsymbol{q}_4, \ldots$.
4. *and so on...*
5. **Step** n**:** Finalize $\boldsymbol{q}_{n-1}$ and substract $\text{proj}_{\boldsymbol{q}_{n-1}}(\boldsymbol{a}_n)$ from $\boldsymbol{q}_n$.
6. **Last step:** Finalize $\boldsymbol{q}_n$ and **quit.**

$$
\begin{aligned}
\boldsymbol{q}_1 &\leftarrow \boldsymbol{a}_1 \\
\boldsymbol{q}_2 &\leftarrow \boldsymbol{a}_2 - \text{proj}_{\boldsymbol{q}_1}(\boldsymbol{a}_2) \\
\boldsymbol{q}_3 &\leftarrow \boldsymbol{a}_3 - \text{proj}_{\boldsymbol{q}_1}(\boldsymbol{a}_3) - \text{proj}_{\boldsymbol{q}_2}(\boldsymbol{a}_3) \\
&\vdots \\
\boldsymbol{q}_n &\leftarrow \boldsymbol{a}_n - \text{proj}_{\boldsymbol{q}_1}(\boldsymbol{a}_n) - \text{proj}_{\boldsymbol{q}_2}(\boldsymbol{a}_n) - \cdots - \text{proj}_{\boldsymbol{q}_{n-1}}(\boldsymbol{a}_n)
\end{aligned}
$$

Demostration of the Gram-Schmidt process: The orthogonalization

$$A = \begin{bmatrix} 4.000 & 2.000 & -5.000 & -9.000 \\ 1.000 & -7.000 & -5.000 & -6.000 \\ 6.000 & -3.000 & -6.000 & -9.000 \\ 7.000 & 9.000 & 0.000 & 8.000 \end{bmatrix}$$

$$Q = \begin{bmatrix} 4.000 & 2.000 & -5.000 & -9.000 \\ 1.000 & -7.000 & -5.000 & -6.000 \\ 6.000 & -3.000 & -6.000 & -9.000 \\ 7.000 & 9.000 & 0.000 & 8.000 \end{bmatrix}$$

$$R = \begin{bmatrix} 1.000 & \mathbf{0.451} & 0.000 & 0.000 \\ 0.000 & 1.000 & 0.000 & 0.000 \\ 0.000 & 0.000 & 1.000 & 0.000 \\ 0.000 & 0.000 & 0.000 & 1.000 \end{bmatrix}$$

Demostration of the Gram-Schmidt process: The orthogonalization

$$A = \begin{bmatrix} 4.000 & 2.000 & -5.000 & -9.000 \\ 1.000 & -7.000 & -5.000 & -6.000 \\ 6.000 & -3.000 & -6.000 & -9.000 \\ 7.000 & 9.000 & 0.000 & 8.000 \end{bmatrix}$$

$$Q = \begin{bmatrix} 4.000 & 0.196 & -5.000 & -9.000 \\ 1.000 & -7.451 & -5.000 & -6.000 \\ 6.000 & -5.706 & -6.000 & -9.000 \\ 7.000 & 5.843 & 0.000 & 8.000 \end{bmatrix}$$

$$R = \begin{bmatrix} 1.000 & 0.451 & 0.000 & 0.000 \\ 0.000 & 1.000 & 0.000 & 0.000 \\ 0.000 & 0.000 & 1.000 & 0.000 \\ 0.000 & 0.000 & 0.000 & 1.000 \end{bmatrix}$$

Demostration of the Gram-Schmidt process: The orthogonalization

$$A = \begin{bmatrix} 4.000 & 2.000 & -5.000 & -9.000 \\ 1.000 & -7.000 & -5.000 & -6.000 \\ 6.000 & -3.000 & -6.000 & -9.000 \\ 7.000 & 9.000 & 0.000 & 8.000 \end{bmatrix}$$

$$Q = \begin{bmatrix} 4.000 & 0.196 & -5.000 & -9.000 \\ 1.000 & -7.451 & -5.000 & -6.000 \\ 6.000 & -5.706 & -6.000 & -9.000 \\ 7.000 & 5.843 & 0.000 & 8.000 \end{bmatrix}$$

$$R = \begin{bmatrix} 1.000 & 0.451 & \mathbf{-0.598} & 0.000 \\ 0.000 & 1.000 & 0.000 & 0.000 \\ 0.000 & 0.000 & 1.000 & 0.000 \\ 0.000 & 0.000 & 0.000 & 1.000 \end{bmatrix}$$

Demostration of the Gram-Schmidt process: The orthogonalization

$$A = \begin{bmatrix} 4.000 & 2.000 & -5.000 & -9.000 \\ 1.000 & -7.000 & -5.000 & -6.000 \\ 6.000 & -3.000 & -6.000 & -9.000 \\ 7.000 & 9.000 & 0.000 & 8.000 \end{bmatrix}$$

$$Q = \begin{bmatrix} 4.000 & 0.196 & -2.608 & -9.000 \\ 1.000 & -7.451 & -4.402 & -6.000 \\ 6.000 & -5.706 & -2.412 & -9.000 \\ 7.000 & 5.843 & 4.186 & 8.000 \end{bmatrix}$$

$$R = \begin{bmatrix} 1.000 & 0.451 & -0.598 & 0.000 \\ 0.000 & 1.000 & 0.000 & 0.000 \\ 0.000 & 0.000 & 1.000 & 0.000 \\ 0.000 & 0.000 & 0.000 & 1.000 \end{bmatrix}$$

Demostration of the Gram-Schmidt process: The orthogonalization

$$A = \begin{bmatrix} 4.000 & 2.000 & -5.000 & -9.000 \\ 1.000 & -7.000 & -5.000 & -6.000 \\ 6.000 & -3.000 & -6.000 & -9.000 \\ 7.000 & 9.000 & 0.000 & 8.000 \end{bmatrix}$$

$$Q = \begin{bmatrix} 4.000 & 0.196 & -2.608 & -9.000 \\ 1.000 & -7.451 & -4.402 & -6.000 \\ 6.000 & -5.706 & -2.412 & -9.000 \\ 7.000 & 5.843 & 4.186 & 8.000 \end{bmatrix}$$

$$R = \begin{bmatrix} 1.000 & 0.451 & -0.598 & \mathbf{-0.392} \\ 0.000 & 1.000 & 0.000 & 0.000 \\ 0.000 & 0.000 & 1.000 & 0.000 \\ 0.000 & 0.000 & 0.000 & 1.000 \end{bmatrix}$$

Demostration of the Gram-Schmidt process: The orthogonalization

$$A = \begin{bmatrix} 4.000 & 2.000 & -5.000 & -9.000 \\ 1.000 & -7.000 & -5.000 & -6.000 \\ 6.000 & -3.000 & -6.000 & -9.000 \\ 7.000 & 9.000 & 0.000 & 8.000 \end{bmatrix}$$

$$Q = \begin{bmatrix} 4.000 & 0.196 & -2.608 & -7.431 \\ 1.000 & -7.451 & -4.402 & -5.608 \\ 6.000 & -5.706 & -2.412 & -6.647 \\ 7.000 & 5.843 & 4.186 & 10.745 \end{bmatrix}$$

$$R = \begin{bmatrix} 1.000 & 0.451 & -0.598 & -0.392 \\ 0.000 & 1.000 & 0.000 & 0.000 \\ 0.000 & 0.000 & 1.000 & 0.000 \\ 0.000 & 0.000 & 0.000 & 1.000 \end{bmatrix}$$

Demostration of the Gram-Schmidt process: The orthogonalization

$$A = \begin{bmatrix} 4.000 & 2.000 & -5.000 & -9.000 \\ 1.000 & -7.000 & -5.000 & -6.000 \\ 6.000 & -3.000 & -6.000 & -9.000 \\ 7.000 & 9.000 & 0.000 & 8.000 \end{bmatrix}$$

$$Q = \begin{bmatrix} 4.000 & 0.196 & -2.608 & -7.431 \\ 1.000 & -7.451 & -4.402 & -5.608 \\ 6.000 & -5.706 & -2.412 & -6.647 \\ 7.000 & 5.843 & 4.186 & 10.745 \end{bmatrix}$$

$$R = \begin{bmatrix} 1.000 & 0.451 & -0.598 & -0.392 \\ 0.000 & 1.000 & \mathbf{0.577} & 0.000 \\ 0.000 & 0.000 & 1.000 & 0.000 \\ 0.000 & 0.000 & 0.000 & 1.000 \end{bmatrix}$$

Demostration of the Gram-Schmidt process: The orthogonalization

$$A = \begin{bmatrix} 4.000 & 2.000 & -5.000 & -9.000 \\ 1.000 & -7.000 & -5.000 & -6.000 \\ 6.000 & -3.000 & -6.000 & -9.000 \\ 7.000 & 9.000 & 0.000 & 8.000 \end{bmatrix}$$

$$Q = \begin{bmatrix} 4.000 & 0.196 & -2.721 & -7.431 \\ 1.000 & -7.451 & -0.105 & -5.608 \\ 6.000 & -5.706 & 0.879 & -6.647 \\ 7.000 & 5.843 & 0.816 & 10.745 \end{bmatrix}$$

$$R = \begin{bmatrix} 1.000 & 0.451 & -0.598 & -0.392 \\ 0.000 & 1.000 & 0.577 & 0.000 \\ 0.000 & 0.000 & 1.000 & 0.000 \\ 0.000 & 0.000 & 0.000 & 1.000 \end{bmatrix}$$

Demostration of the Gram-Schmidt process: The orthogonalization

$$A = \begin{bmatrix} 4.000 & 2.000 & -5.000 & -9.000 \\ 1.000 & -7.000 & -5.000 & -6.000 \\ 6.000 & -3.000 & -6.000 & -9.000 \\ 7.000 & 9.000 & 0.000 & 8.000 \end{bmatrix}$$

$$Q = \begin{bmatrix} 4.000 & 0.196 & -2.721 & -7.431 \\ 1.000 & -7.451 & -0.105 & -5.608 \\ 6.000 & -5.706 & 0.879 & -6.647 \\ 7.000 & 5.843 & 0.816 & 10.745 \end{bmatrix}$$

$$R = \begin{bmatrix} 1.000 & 0.451 & -0.598 & -0.392 \\ 0.000 & 1.000 & 0.577 & \mathbf{1.154} \\ 0.000 & 0.000 & 1.000 & 0.000 \\ 0.000 & 0.000 & 0.000 & 1.000 \end{bmatrix}$$

Demostration of the Gram-Schmidt process: The orthogonalization

$$A = \begin{bmatrix} 4.000 & 2.000 & -5.000 & -9.000 \\ 1.000 & -7.000 & -5.000 & -6.000 \\ 6.000 & -3.000 & -6.000 & -9.000 \\ 7.000 & 9.000 & 0.000 & 8.000 \end{bmatrix}$$

$$Q = \begin{bmatrix} 4.000 & 0.196 & -2.721 & -7.658 \\ 1.000 & -7.451 & -0.105 & 2.988 \\ 6.000 & -5.706 & 0.879 & -0.064 \\ 7.000 & 5.843 & 0.816 & 4.004 \end{bmatrix}$$

$$R = \begin{bmatrix} 1.000 & 0.451 & -0.598 & -0.392 \\ 0.000 & 1.000 & 0.577 & 1.154 \\ 0.000 & 0.000 & 1.000 & 0.000 \\ 0.000 & 0.000 & 0.000 & 1.000 \end{bmatrix}$$

Demostration of the Gram-Schmidt process: The orthogonalization

$$A = \begin{bmatrix} 4.000 & 2.000 & -5.000 & -9.000 \\ 1.000 & -7.000 & -5.000 & -6.000 \\ 6.000 & -3.000 & -6.000 & -9.000 \\ 7.000 & 9.000 & 0.000 & 8.000 \end{bmatrix}$$

$$Q = \begin{bmatrix} 4.000 & 0.196 & -2.721 & -7.658 \\ 1.000 & -7.451 & -0.105 & 2.988 \\ 6.000 & -5.706 & 0.879 & -0.064 \\ 7.000 & 5.843 & 0.816 & 4.004 \end{bmatrix}$$

$$R = \begin{bmatrix} 1.000 & 0.451 & -0.598 & -0.392 \\ 0.000 & 1.000 & 0.577 & 1.154 \\ 0.000 & 0.000 & 1.000 & \mathbf{2.681} \\ 0.000 & 0.000 & 0.000 & 1.000 \end{bmatrix}$$

Demostration of the Gram-Schmidt process: The orthogonalization

$$A = \begin{bmatrix} 4.000 & 2.000 & -5.000 & -9.000 \\ 1.000 & -7.000 & -5.000 & -6.000 \\ 6.000 & -3.000 & -6.000 & -9.000 \\ 7.000 & 9.000 & 0.000 & 8.000 \end{bmatrix}$$

$$Q = \begin{bmatrix} 4.000 & 0.196 & -2.721 & -0.363 \\ 1.000 & -7.451 & -0.105 & 3.269 \\ 6.000 & -5.706 & 0.879 & -2.421 \\ 7.000 & 5.843 & 0.816 & 1.816 \end{bmatrix}$$

$$R = \begin{bmatrix} 1.000 & 0.451 & -0.598 & -0.392 \\ 0.000 & 1.000 & 0.577 & 1.154 \\ 0.000 & 0.000 & 1.000 & 2.681 \\ 0.000 & 0.000 & 0.000 & 1.000 \end{bmatrix}$$

Demostration of the Gram-Schmidt process: The orthogonalization

$$A = \begin{bmatrix} 4.000 & 2.000 & -5.000 & -9.000 \\ 1.000 & -7.000 & -5.000 & -6.000 \\ 6.000 & -3.000 & -6.000 & -9.000 \\ 7.000 & 9.000 & 0.000 & 8.000 \end{bmatrix}$$

$$Q = \begin{bmatrix} 4.000 & 0.196 & -2.721 & -0.363 \\ 1.000 & -7.451 & -0.105 & 3.269 \\ 6.000 & -5.706 & 0.879 & -2.421 \\ 7.000 & 5.843 & 0.816 & 1.816 \end{bmatrix}$$

$$R = \begin{bmatrix} 1.000 & 0.451 & -0.598 & -0.392 \\ 0.000 & 1.000 & 0.577 & 1.154 \\ 0.000 & 0.000 & 1.000 & 2.681 \\ 0.000 & 0.000 & 0.000 & 1.000 \end{bmatrix}$$

Demonstration of the Gram-Schmidt process: The normalization

$$Q^T Q = \begin{bmatrix} 102.000 & 0.000 & 0.000 & 0.000 \\ 0.000 & 122.255 & 0.000 & 0.000 \\ 0.000 & 0.000 & 8.853 & 0.000 \\ 0.000 & 0.000 & 0.000 & 19.974 \end{bmatrix}$$

$$Q_{\text{normed}} = \begin{bmatrix} 0.396 & 0.018 & -0.914 & -0.081 \\ 0.099 & -0.674 & -0.035 & 0.731 \\ 0.594 & -0.516 & 0.295 & 0.542 \\ 0.693 & 0.528 & 0.274 & 0.406 \end{bmatrix}$$

$$R_{\text{normed}} = \begin{bmatrix} 10.100 & 4.555 & -6.040 & -3.961 \\ 0.000 & 11.057 & 6.377 & 12.756 \\ 0.000 & 0.000 & 2.975 & 7.977 \\ 0.000 & 0.000 & 0.000 & 4.469 \end{bmatrix}$$

Implementation of the Gram-Schmidt process

Arguments for:

- After the kth step we have the first k elements of orthogonal system.
- It works for singular and/or nonsquare matrices as well.
- However, one must deal with the case when one of the $\boldsymbol{q}_i$'s is zero.
- Then, one defines $\text{proj}_{\mathbf{0}}(\boldsymbol{x}) = \mathbf{0}$ for all $\boldsymbol{x}$.

Arguments against:

- Numerically unstable, slight modifications are needed.
- Other orthogonalization algorithms use Householder transformations or Givens rotations.

Section 4

The Eigenvalue Problem

Subsection 1

Introduction

Eigenvalues, eigenvectors

Definition: Eigenvalue, eigenvector

Let A be a complex $N \times N$ square matrix. The pair $(\lambda, \mathbf{v})$ $(\lambda \in \mathbb{C}, \mathbf{v} \in \mathbb{C}^N)$ is called an eigenvalue, eigenvector pair of A if $\lambda\mathbf{v} = A\mathbf{v}$ and $\mathbf{v} \neq \mathbf{0}$.

Example

The pair $\lambda = 2$ and $\mathbf{v} = [4, -1]^T$ is an eigenvalue, eigenvector pair of the matrix

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

since

$$2\begin{bmatrix} 4 \\ -1 \end{bmatrix} = \begin{bmatrix} 3 & 4 \\ 0 & 2 \end{bmatrix} \cdot \begin{bmatrix} 4 \\ -1 \end{bmatrix}.$$

Motivation

Finding the eigenvalues of matrices is very important in numerous applications.

Applications

1. Solving the Schrödinger equation in quantum mechanics
2. Molecular orbitals can be defined by the eigenvectors of the Fock operator
3. Geology - study of glacial till
4. Principal components analysis
5. Vibration analysis of mechanical structures (with many degrees of freedom)
6. Image processing

Tacoma Narrows Bridge

Image source: `http://www.answers.com/topic/galloping-gertie-large-image`

Characteristic polynomial

Proposition

λ is an eigenvalue of A if, and only if, $\det(A - \lambda I_N) = 0$, where I_N is the identity matrix of size N.

Proof. We observe that $\lambda \mathbf{v} = A\mathbf{v} \Leftrightarrow (A - \lambda I)\mathbf{v} = \mathbf{0}$. The latter system of linear equations is homogenius, and has a non-trivial solution if, and only if, it is singular. □

Definition

The Nth-degree polynomial $p(\lambda) = \det(A - \lambda I)$ is called the characteristic polynomial of A.

Characteristic polynomial

Example

Consider the matrix

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

The characteristic polynomial of A is

$$\det(A - \lambda I) = \det \begin{bmatrix} 2-\lambda & 4 & -1 \\ 0 & 1-\lambda & 1 \\ 0 & 0 & 1-\lambda \end{bmatrix} =$$

$$= (2-\lambda)(1-\lambda)^2 = -\lambda^3 + 4\lambda^2 - 5\lambda + 2.$$

The problem of finding the eigenvalues of an $N \times N$ matrix is equivalent to solving a polynomial equation of degree N.

Example

Consider the matrix

$$B = \begin{bmatrix} -\alpha_1 & -\alpha_2 & \cdots & -\alpha_{N-1} & -\alpha_N \\ 1 & 0 & \cdots & 0 & 0 \\ 0 & 1 & \cdots & 0 & 0 \\ \vdots & \vdots & & \vdots & \vdots \\ 0 & 0 & \cdots & 1 & 0 \end{bmatrix}.$$

An inductive argument shows that the characteristic polynomial of B is $p(\lambda) = (-1)^N \cdot (\lambda^N + \alpha_1 \lambda^{N-1} + \alpha_2 \lambda^{N-2} + \ldots + \alpha_{N-1}\lambda + \alpha_N)$.

The above example shows, that every polynomial is the characteristic polynomial of some matrix.

The Abel-Ruffini theorem

Abel-Ruffini theorem

There is no general algebraic solution – that is, solution in radicals – to polynomial equations of degree five or higher.

This theorem together with the previous example shows us, that we cannot hope for an exact solution for the eigenvalue problem in general, if $N > 4$.

Thus, we are interested - as usual - in iterative methods, that produce approximate solutions, and also we might be interested in solving special cases.

Subsection 2

The Jacobi Method

Symmetric matrices

First, we are going to present an algorithm, that iteratively approximates the eigenvalues of a real symmetric matrix.

Proposition

If A is a real symmetric matrix, then all eigenvalues of A are real numbers.

Proposition

If A is positive-definite, then all eigenvalues of A are positive real numbers. If A is positive-semidefinite, then all eigenvalues of A are nonnegative real numbers.

The idea of Jacobi

First we note that the eigenvalues of a diagonal (even upper triangular) matrix are exactly the diagonal elements.

Thus the idea is to transform the matrix (in many steps) into (an almost) diagonal form without changing the eigenvalues.

$$\begin{bmatrix} a_{11} & * & * & \dots & * \\ * & a_{22} & * & \dots & * \\ * & * & a_{33} & \dots & * \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ * & * & * & \dots & a_{NN} \end{bmatrix} \overset{e.p.t.}{\rightsquigarrow} \begin{bmatrix} \lambda_1 & 0 & 0 & \dots & 0 \\ 0 & \lambda_2 & 0 & \dots & 0 \\ 0 & 0 & \lambda_3 & \dots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & \dots & \lambda_N \end{bmatrix}$$

Eigenvalues and similarities

Proposition

Let A and X be $N \times N$ matrices, and assume $\det X \neq 0$. λ is an eigenvalue of A if, and only if, λ is an eigenvalue of $X^{-1}AX$.

Proof. Observe, that

$$\det(X^{-1}AX - \lambda I) = \det(X^{-1}(A - \lambda I)X) = \det X^{-1} \det(A - \lambda I) \det X.$$

Thus, $\det(A - \lambda I) = 0 \Leftrightarrow \det(X^{-1}AX - \lambda I) = 0$. □

Corollary

Let A and X be $N \times N$ matrices, and assume that A is symmetric and X is orthogonal. λ is an eigenvalue of A if, and only if, λ is an eigenvalue of the symmetric matrix $X^T AX$.

Rotation matrices

We introduce Givens rotation matrices as follows.

$$Q_{ij} = \begin{bmatrix} 1 & & & & & & & & \\ & 1 & & & & & & & \\ & & 1 & & & & & & \\ & & & c & & -s & & & \\ & & & & 1 & & & & \\ & & & & & 1 & & & \\ & & & s & & c & & & \\ & & & & & & 1 & & \\ & & & & & & & 1 & \\ & & & & & & & & 1 \end{bmatrix}$$

Here $c^2 + s^2 = 1$, and $[Q_{ij}]_{ii} = c$, $[Q_{ij}]_{jj} = c$, $[Q_{ij}]_{ij} = -s$, $[Q_{ij}]_{ji} = s$, all other diagonal elements are 1, and we have zeros everywhere else.

Rotation matrices

Proposition

Q_{ij} is orthogonal.

We would like to achieve a diagonal form by succesively transforming the original matrix A via Givens rotations matrices. With the proper choice of c and s we can zero out symmetric pairs of non-diagonal elements.

Example

Let $i = 1, j = 3, c = 0.973249$ and $s = 0.229753$. Then

$$Q_{ij}^T \cdot \begin{bmatrix} 1 & 2 & -1 & 1 \\ 2 & 1 & 0 & 5 \\ -1 & 0 & 5 & 3 \\ 1 & 5 & 3 & 2 \end{bmatrix} \cdot Q_{ij} = \begin{bmatrix} 0.764 & 1.946 & 0 & 1.663 \\ 1.946 & 1 & -0.460 & 5 \\ 0 & -0.460 & 5.236 & 2.690 \\ 1.663 & 5 & 2.690 & 2 \end{bmatrix}$$

Creating zeros

A tedious, but straightforward calculation gives the following, general lemma.

Lemma

Let A be a symmetric matrix, and assume $a_{ij} = a_{ji} \neq 0$. Let $B = Q_{ij}^T A Q_{ij}$. Then B is a symmetric matrix with

$$\begin{aligned} b_{ii} &= c^2 a_{ii} + s^2 a_{jj} + 2sca_{ij}, \\ b_{jj} &= s^2 a_{ii} + c^2 a_{jj} - 2sca_{ij}, \\ b_{ij} &= b_{ji} = cs(a_{jj} - a_{ii}) + (c^2 - s^2)a_{ji}. \end{aligned}$$

In particular, if

$$c = \left(\frac{1}{2} + \frac{\beta}{2 \cdot (1 + \beta^2)^{1/2}} \right)^{1/2}; \qquad s = \left(\frac{1}{2} - \frac{\beta}{2 \cdot (1 + \beta^2)^{1/2}} \right)^{1/2}$$

with $2\beta = (a_{ii} - a_{jj})/a_{ij}$, then $b_{ij} = b_{ji} = 0$.

Approching a diagonal matrix

With the help of the previous lemma we can zero out any non-diagonal element with just one conjugation. The problem is, that while creating new zeros, we might ruin some previously done work, as it was shown in the example.
Thus, instead of reaching a diagonal form, we try to minimize the sum of squares of the non-diagonal elements. The following theorem makes this idea precise.

Definition

Let X be a symmetric matrix. We introduce

$$\operatorname{sqsum}(X) = \sum_{i,j=1}^{N} x_{ij}^2; \qquad \operatorname{diagsqsum}(X) = \sum_{i=1}^{N} x_{ii}^2$$

for the sum of squares of all elements, and for the sum of squares of the diagonal elements, respectively.

Main theorem

Theorem

Assume that the symmetric matrix A has been transformed into the symmetric matrix $B = Q_{ij}^T A Q_{ij}$ such that $b_{ij} = b_{ji} = 0$. Then the sum of squares of all elements remains unchanged, while the sum of squares of the diagonal elements increases. More precisely

$$\text{sqsum}(A) = \text{sqsum}(B); \qquad \text{diagsqsum}(B) = \text{diagsqsum}(A) + 2a_{ij}^2.$$

Proof. Using $c^2 + s^2 = 1$, from the previous lemma we may deduce by straigthforward calculation the following identity:

$$b_{ii}^2 + b_{jj}^2 + 2b_{ij}^2 = a_{ii}^2 + a_{jj}^2 + 2a_{ij}^2.$$

Since we assumed $b_{ij} = 0$, and obviously $b_{kk} = a_{kk}$ if $k \neq i, j$, it follows that $\text{diagsqsum}(B) = \text{diagsqsum}(A) + 2a_{ij}^2$.

Proof of Main Theorem

Introduce $P = AQ_{ij}$, and denote the kth column of A by $\mathbf{a}_k$, and the kth column of P by $\mathbf{p}_k$. We claim that sqsum(P) = sqsum(A).

Observe, that $\mathbf{p}_i = c\mathbf{a}_i + s\mathbf{a}_j$, $\mathbf{p}_j = -s\mathbf{a}_i + c\mathbf{a}_j$, while $\mathbf{p}_k = \mathbf{a}_k$ if $k \neq i, j$. Thus

$$\begin{aligned} \|\mathbf{p}_i\|_2^2 + \|\mathbf{p}_j\|_2^2 &= \mathbf{p}_i^T\mathbf{p}_i + \mathbf{p}_j^T\mathbf{p}_j = c^2\mathbf{a}_i^T\mathbf{a}_i + 2cs\mathbf{a}_i^T\mathbf{a}_j + s^2\mathbf{a}_j^T\mathbf{a}_j \\ &+ s^2\mathbf{a}_i^T\mathbf{a}_i - 2cs\mathbf{a}_i^T\mathbf{a}_j + c^2\mathbf{a}_j^T\mathbf{a}_j = \mathbf{a}_i^T\mathbf{a}_i + \mathbf{a}_j^T\mathbf{a}_j = \|\mathbf{a}_i\|_2^2 + \|\mathbf{a}_j\|_2^2 \end{aligned}$$

and hence

$$\text{sqsum}(P) = \sum_{k=1}^{N} \|\mathbf{p}_k\|_2^2 = \sum_{k=1}^{N} \|\mathbf{a}_k\|_2^2 = \text{sqsum}(A).$$

We may show similarly, that sqsum(P) = sqsum$(Q_{ij}^T P)$, which implies sqsum(A) = sqsum(B). □

One Jacobi iteration

We knock out non-diagonal elements iteratively conjugating with Givens rotation matrices. To perform one Jacobi iteration first we need to pick an a_{ij} we would like to knock out, then calculate the values of c and s (see Lemma), finally perform the calculation $Q_{ij}^T A Q_{ij}$. Since only the ith and jth rows and columns of A change actually, one iteration needs only $O(N)$ time.

We need to find a strategy to pick a_{ij} effectively.

- If we systematically knock out every non-zero element in some prescribed order until each non-diagonal element is small, we may spend much time with knocking out already small elements.
- It seems feasible to find the largest non-diagonal element to knock out, however it takes $O(N^2)$ time.

Strategy to pick a_{ij}

Instead of these, we choose a strategy somewhere in between: we check all non-diagonal elements in a prescribed cyclic order, zeroing every element that is „larger than half-average”. More precisely, we knock out an element a_{ij} if

$$a_{ij}^2 > \frac{\text{sqsum}(A) - \text{diagsqsum}(A)}{2N(N-1)}.$$

Theorem

If in the Jacobi method we follow the strategy above, then the convergence criterion

$$\text{sqsum}(A) - \text{diagsqsum}(A) \leq \varepsilon \cdot \text{sqsum}(A)$$

will be satisfied after at most $N^2 \ln(1/\varepsilon)$ iterations.

Speed of convergence

Proof. Denote sqsum(A) – diagsqsum(A) after k iterations by e_k. By the Main Theorem and by the strategy it follows that

$$e_{k+1} = e_k - 2a_{ij}^2 \leq e_k - \frac{e_k}{N(N-1)} < e_k \left(1 - \frac{1}{N^2}\right) \leq e_k \exp\left(\frac{-1}{N^2}\right).$$

Thus after $L = N^2 \ln(1/\varepsilon)$ iterations

$$e_L \leq e_0 \left[\exp\left(\frac{-1}{N^2}\right)\right]^L \leq e_0 \exp\left[-\ln\left(\frac{1}{\varepsilon}\right)\right] = e_0\varepsilon.$$

Since sqsum(A) remains unchanged, the statement of the Theorem readily follows. □

Demonstration via example

We start with the matrix

Example

$$A = A_0 = \begin{bmatrix} 1 & 5* & -1 & 1 \\ 5 & 1 & 0 & 2 \\ -1 & 0 & 5 & 3 \\ 1 & 2 & 3 & 2 \end{bmatrix}$$

and show the effect of a couple of iterations.

Before we start the Jacobi method we have $\text{sqsum}(A) = 111$ and $\text{diagsqsum}(A) = 31$. Thus the critical value is $(111 - 31)/24 = 3.33$. We prescribe the natural order on the upper triangle, so we choose $i = 1$ and $j = 2$ (see starred element). We use 3 digits accuracy during the calculation.

Demonstration via example

After one iteration

$$A_1 = \begin{bmatrix} -4 & 0 & -0.707 & -0.707 \\ 0 & 6 & -0.707 & 2.121* \\ -0.707 & -0.707 & 5 & 3 \\ -0.707 & 2.121 & 3 & 2 \end{bmatrix}$$

- sqsum(A_1) = 111
- diagsqsum(A_1) = 81
- critical value 1.250

For the next iteration we pick $i = 2$ and $j = 4$ (see starred element).

Demonstration via example

After two iterations

$$A_2 = \begin{bmatrix} -4 & -0.28 & -0.707 & -0.649 \\ -0.28 & 6.915 & 0.539 & 0 \\ -0.707 & 0.539 & 5 & 3.035* \\ -0.649 & 0 & 3.035 & 1.085 \end{bmatrix}$$

- sqsum(A_1) = 111
- diagsqsum(A_1) = 90
- critical value 0.875

For the next iteration we pick $i = 3$ and $j = 4$ (see starred element).

Demonstration via example

After three iterations

$$A_3 = \begin{bmatrix} -4 & -0.28 & -0.931* & -0.232 \\ -0.28 & 6.915 & 0.473 & -0.258 \\ -0.931 & 0.473 & 6.654 & 0 \\ -0.232 & -0.258 & 0 & -0.569 \end{bmatrix}$$

- $\text{sqsum}(A_1) = 111$
- $\text{diagsqsum}(A_1) = 108,416$
- critical value 0.107

For the next iteration we pick $i = 1$ and $j = 3$ (see starred element).

Demonstration via example

After four iterations

$$A_4 = \begin{bmatrix} -4.081 & -0.238 & 0 & -0.231* \\ -0.238 & 6.915 & 0.495 & -0.258 \\ 0 & 0.495 & 6.735 & 0.02 \\ -0.231 & -0.258 & 0.02 & -0.569 \end{bmatrix}$$

- $\text{sqsum}(A_1) = 111$
- $\text{diagsqsum}(A_1) = 110, 155$
- critical value 0.035

For the next iteration we pick $i = 1$ and $j = 4$ (see starred element).

Demonstration via example

After five iterations

$$A_5 = \begin{bmatrix} -4.096 & -0.254 & 0,001 & 0 \\ -0.254 & 6.915 & 0.495* & -0.242 \\ 0.001 & 0.495 & 6.735 & 0.02 \\ 0 & -0.242 & 0.02 & -0.554 \end{bmatrix}$$

- $\text{sqsum}(A_1) = 111$
- $\text{diagsqsum}(A_1) = 110,262$
- critical value 0.031

For the next iteration we pick $i = 2$ and $j = 3$ (see starred element).

Demonstration via example

After six iterations

$$A_6 = \begin{bmatrix} -4.096 & -0.194 & 0,164 & 0 \\ -0.194 & 7,328 & 0 & -0.173* \\ 0.164 & 0 & 6.322 & 0.17 \\ 0 & -0.173 & 0.17 & -0.554 \end{bmatrix}$$

- $\text{sqsum}(A_1) = 111$
- $\text{diagsqsum}(A_1) = 110,751$
- critical value 0.010

For the next iteration we would pick $i = 2$ and $j = 4$ (see starred element). We stop here. The eigenvalues of the original matrix A are $\lambda_1 = -4.102$, $\lambda_2 = 7.336$, $\lambda_3 = 6.322$ and $\lambda_4 = -0.562$. Thus our estimation is already 0.01 exact.

The Jacobi method

We summarize the Jacobi method. We assume that $\varepsilon > 0$ and a symmetric matrix $A = A_0$ are given.

1. **Initialization** We compute sqsum(A) and dss = diagsqsum(A), and set $k = 0$. Then repeat the following steps until

$$\text{sqsum}(A) - dss \leq \varepsilon \cdot \text{sqsum}(A).$$

2. Consider the elements of A_k above its diagonal in the natural cyclical order and find the next a_{ij} with $a_{ij}^2 > \frac{\text{sqsum}(A) - \text{diagsqsum}(A_k)}{2N(N-1)}$.
3. Compute c and s according to the lemma, and construct Q_{ij}. Compute $A_{k+1} = Q_{ij}^T A_k Q_{ij}$.
4. Put $dss = dss + 2a_{ij}^2 (= \text{diagsqsum}(A_{k+1}))$, $k = k + 1$, and repeat the cycle.

When the algorithm stops, A_k is almost diagonal, and the eigenvalues of A are listed in the main diagonal.

Subsection 3

QR method for general matrices

General matrices

The Jacobi method works only for real symmetric matrices. We cannot hope to modify or fix it, since obviously the main trick always produces real numbers into the diagonal, while a general real matrix can have complex eigenvalues.

We sketch an algorithm that effectively finds good approximations of the eigenvalues of general matrices. We apply an itaretive method that is based on the QR decomposition of the matrices. It turns out, that this method converges pretty slowly for general matrices, and to perform one iteration step, we need $O(N^3)$ time. However, if we apply the method for upper Hessenberg matrices, then one iteration takes only $O(N^2)$, and the upper Hessenberg structure is preserved.

Review: the QR decomposition

QR decomposition

Let A be a real square matrix. Then there is a Q orthogonal matrix and there is an R matrix in row echelon form such that $A = QR$.

Example

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

As we saw earlier, the QR decomposition of a matrix can be calculated via the Gram-Schmidt process in $O(N^3)$ steps.

The QR method

We start with a square matrix $A = A_0$. Our goal is to transform A into upper triangular from without changing the eigenvalues.

The QR method

1. Set $k = 0$.
2. Consider A_k, and compute its QR decompostion $A_k = Q_k R_k$.
3. We define $A_{k+1} = R_k Q_k$.
4. Put $k = k + 1$, and go back to Step 2.

Note that for any k we have $Q_k^{-1} = Q_k^T$ since Q_k is orthogonal, and so $R_k = Q_k^T A_k$. Hence $A_{k+1} = R_k Q_k = Q_k^T A_k Q_k$, which shows that A_k and A_{k+1} have the same eigenvalues.

The QR method

Theorem

Assume that A has N eigenvalues satisfying

$$|\lambda_1| > |\lambda_2| > \ldots > |\lambda_N| > 0.$$

Then A_k defined above approaches upper triangular form.

Thus, after a finite number of iterations we get a good approximations of the eigenvalues of the original matrix A. The following, more precise statement shows the speed of the conergence.

Theorem

We use the notation above, and let $a_{ij}^{(k)}$ be the jth element in the ith row of A_k. For $i > j$

$$|a_{ij}^{(k)}| = O\left(\left|\frac{\lambda_i}{\lambda_j}\right|^k\right).$$

Demonstartion of the QR method

We demonstrate the speed of the convergence through numerical examples. (Source: `http://people.inf.ethz.ch/arbenz/ewp/Lnotes/chapter3.pdf`)

Initial value

$$A = A_0 = \begin{bmatrix} -0.445 & 4.906 & -0.879 & 6.304 \\ -6.394 & 13.354 & 1.667 & 11.945 \\ 3.684 & -6.662 & -0.06 & -7.004 \\ 3.121 & -5.205 & -1.413 & -2.848 \end{bmatrix}$$

The eigenvalues of the matrix A are approximately 1, 2, 3 and 4.

Demonstartion of the QR method

After 5 iterations we obtain the following.

After 5 iterations

$$A_5 = \begin{bmatrix} 4.076 & 0.529 & -6.013 & -22.323 \\ -0.054 & 2.904 & 1.338 & -2.536 \\ 0.018 & 0.077 & 1.883 & 3.248 \\ 0.001 & 0.003 & 0.037 & 1.137 \end{bmatrix}$$

[The eigenvalues of the matrix A and A_5 are approximately 1, 2, 3 and 4.]

Demonstartion of the QR method

After 10 iterations we obtain the following.

After 10 iterations

$$A_{10} = \begin{bmatrix} 4.002 & 0.088 & -7.002 & -21.931 \\ -0.007 & 2.990 & 0.937 & 3.087 \\ 0.001 & 0.011 & 2.002 & 3.618 \\ 0.000 & 0.000 & -0.001 & 1.137 \end{bmatrix}$$

[The eigenvalues of the matrix A and A_{10} are approximately 1, 2, 3 and 4.]

Demonstartion of the QR method

After 20 iterations we obtain the following.

After 20 iterations

$$A_{20} = \begin{bmatrix} 4.000 & 0.021 & -7.043 & -21.898 \\ 0.000 & 3.000 & 0.873 & 3.202 \\ 0.000 & 0.000 & 2.000 & -3.642 \\ 0.000 & 0.000 & 0.000 & 1.000 \end{bmatrix}$$

[The eigenvalues of the matrix A and A_{20} are approximately 1, 2, 3 and 4.]

On the QR method

Remarks on the QR method:

- The convergence of the algorith can be very slow, if the eigenvalues are very close to each other.
- The algorithm is expensive. Each iteration step requires $O(N^3)$ time as we showed earlier.

Both issues can be improved. We only sketch a method that reduces the running time of one iteration step. We recall the following definition.

Definition: Upper (lower) Hessenberg matrix

The matrix A is upper (lower) Hessenberg if $a_{ij} = 0$ when $i - 1 > j$ ($i < j - 1$).
A 4×4 example of an upper Hessenberg matrix is

$$A = \begin{bmatrix} 1 & 2 & 5 & 9 \\ -1 & 5 & 2 & -1 \\ 0 & 3 & -2 & 3 \\ 0 & 0 & 1 & 7 \end{bmatrix}$$

Upper Hessenberg matrices

Proposition

Let A be an upper Hessenberg matrix. To perform one iteration of the QR method on A takes $O(N^2)$ time.

Lemma

Let A be an upper Hessenberg matrix with QR decomposition $A = QR$. Then the matrix RQ is also upper Hessenberg.

These two statements ensure that if we start with a matrix A that is in upper Hessenberg form, the QR method can be applied much more effectively.

Similarly, as in the Jacobi method, using Givens rotation matrices, we can transform an arbitrary matrix A into upper Hessenberg form. The algorithm is called Householder reduction, and it has running time $O(N^3)$. (We do not cover this method in details.)

Summary

The extended QR method to find the eigenvalues of a general square matrix A goes as follows.

Extended QR method

1. We transform A into upper Hessenberg form using Householder reduction in $O(N^3)$ steps.
2. We iterate $A_{k+1} = R_k Q_k$ until the desired accuracy is reached. Every iteration step requires $O(N^2)$ time.

A_k approaches an upper triangular form, the eigenvalues of the original matrix A will appear in the main diagonal.

Remark. The speed of the convergence in the QR method can be greatly improved by introducing spectral shifts in the algorithm.

Section 5

A sparse iterative model: Poisson's Equation

Subsection 1

The Jacobi Iteration

Sparse matrices

- A sparse matrix can be vaguely defined as a matrix with few nonzeros. It is important to take advantage of the sparsity.
- Sparsity can be structured or unstructured.
- The fraction of zero elements (resp. non-zero elements) in a matrix is called the sparsity (resp. density).
- An important special type of sparse matrices is that of band matrices.
- The concept of sparsity is useful in combinatorics and application areas such as network theory, which have a low density of significant data or connections.
- Huge sparse matrices often appear in science or engineering when solving partial differential equations.

Basic idea of the iterative method

We want to solve the system

$$A\boldsymbol{x} = \boldsymbol{b} \tag{12}$$

of linear equations. We split A in two parts, B and $A - B$, and write (12) in the form

$$B\boldsymbol{x} = (B - A)\boldsymbol{x} + \boldsymbol{b}. \tag{13}$$

Now we define an iterative method based on this formula:

$$B\boldsymbol{x}_{n+1} = (B - A)\boldsymbol{x}_n + \boldsymbol{b}. \tag{14}$$

1. It is clear that we must choose B so that we can easily solve (13).
2. Typically, one chooses B diagonal or triangular.
3. If $\boldsymbol{x}_n$ converges to to a vector $\boldsymbol{x}_\infty$ then $\boldsymbol{x}_\infty$ satisfies (13), and thus will be a solution for $A\boldsymbol{x} = \boldsymbol{b}$.

Convergence of the iterative method

To determine when the iteration (14) will converge, we subtract (14) from (13) and get

$$B\boldsymbol{e}_{n+1} = (B - A)\boldsymbol{e}_n,$$

where $\boldsymbol{e}_n = \boldsymbol{x} - \boldsymbol{x}_n$ is the error after n iteration. Then

$$\boldsymbol{e}_{n+1} = (I - B^{-1}A)\boldsymbol{e}_n = (I - B^{-1}A)^{n+1}\boldsymbol{e}_0. \tag{15}$$

Theorem: Convergence of the iterative method

If A and B are both nonsingular, and the initial guess $\boldsymbol{x}_0$ is not exactly equal to the solution $\boldsymbol{x}$ of $A\boldsymbol{x} = \boldsymbol{b}$, and if $B\boldsymbol{x}_{n+1} = (B - A)\boldsymbol{x}_n + \boldsymbol{b}$, then $\boldsymbol{x}_n$ converges to $\boldsymbol{x}$ if and only if all eigenvalues of $I - B^{-1}A$ are less than 1 in absolute value.

The proof relies on the following

Proposition

$H^n \to 0$ as $n \to \infty$ if and only if all eigenvalues of H are < 1 in absolute value.

The simplest choice for B: The Jacobi iteration

- One obvious choice for B is the diagonal matrix consisting of the elements of the diagonal of A.
- This choice defines the Jacobi iteration

$$D\boldsymbol{x}_{n+1} = (D - A)\boldsymbol{x}_n + \boldsymbol{b},$$

or

$$(x_i)_{n+1} = \frac{1}{a_{ii}}\left(b_i - \sum_{i \neq j} a_{ij}(x_j)_n\right), \qquad i = 1, \ldots, N. \tag{16}$$

- In practice, either we give estimates on the true eigenvalues of $I - D^{-1}A$, or,
- we apply more general results giving such bounds.

Convergence for diagonal-dominant matrices

Theorem: Diagonal-dominant matrices

If A is diagonal-dominant, that is, if for each i

$$|a_{ii}| > \sum_{j \neq i} |a_{ij}|,$$

then the Jacobi iteration (16) will converge.

Proof. The ith row of $H = I - D^{-1}A$ has the form

$$[-a_{i1}/a_{ii} \ \cdots \ -a_{i,i-1}/a_{ii} \ 0 \ -a_{i,i+1}/a_{ii} \ \cdots \ -a_{iN}/a_{ii}];$$

the sum of the absolute values of the elements is less than. Hence, the ∞-norm of H is less than 1. If $\boldsymbol{z}$ is an eigenvector of H with eigenvalue λ then

$$|\lambda| \, \|\boldsymbol{z}\|_\infty = \|\lambda \boldsymbol{z}\|_\infty = \|H\boldsymbol{z}\|_\infty \leq \|H\|_\infty \|\boldsymbol{z}\|_\infty.$$

Since the eigenvector $\boldsymbol{z} \neq \boldsymbol{0}$, we see that $|\lambda| \leq \|H\|_\infty < 1$ holds for all eigenvalues of H. The convergence follows from the appropriate theorem. □

Subsection 2

Poisson's Equation in one dimension

Poisson's equation in one dimension

We begin with a one-dimensional version of Poisson's equation:

$$-\frac{d^2v(x)}{dx^2} = f(x), \qquad 0 < x < 1, \tag{17}$$

where $f(x)$ is a given function and $v(x)$ is the unknown function that we want to compute. $v(x)$ must also satisfy the *boundary conditions* $v(0) = v(1) = 0$.

Proposition: Properties of $-\frac{d^2}{dx^2}$

Let C_0 be the space of analytic functions on $[0, 1]$ satisfying the boundary condition. The eigenvalues of the linear operator $-\frac{d^2}{dx^2}$ on C_0 are $\hat{\lambda}_i = i^2\pi^2$ with corresponding eigenvectors $\hat{z}_i = \sin(i\pi x)$.

Proof. It is easy to see that for a fixed scalar k, the solutions of the differential equation $-\frac{d^2u}{dx^2} = ku$ have the form $u(x) = \alpha \sin(\sqrt{k}x) + \beta \cos(\sqrt{k}x)$. $u(0) = 0$ implies $\beta = 0$ and $u(1) = 0$ implies $\sqrt{k} = i\pi$ for an integer i. □

Discretization of Poisson's equation

1. We compute an approximate solution at $N+2$ evenly spaced points x_i between 0 and 1: $x_i = ih$, where $h = \frac{1}{N+1}$ and $0 \le i \le N+1$.
2. We abbreviate $v_i = v(x_i)$ and $f_i = f(x_i)$.
3. To convert (17) into a linear equation for the unknowns $v_1, \ldots, v_N$, we use *finite differences*

$$\left.\frac{dv(x)}{dx}\right|_{x=(i-1/2)h} \approx \frac{v_i - v_{i-1}}{h},$$
$$\left.\frac{dv(x)}{dx}\right|_{x=(i+1/2)h} \approx \frac{v_{i+1} - v_i}{h}.$$

4. Subtracting these approximations and dividing by h yield the *centered difference approximation*

$$-\left.\frac{d^2v(x)}{dx^2}\right|_{x=x_i} \approx \frac{2v_i - v_{i-1} - v_{i+1}}{h^2}. \tag{18}$$

Linear form of the discrete Poisson's equation

1. From now on we will not distinguish between v and its approximation $(v_0, \ldots, v_{N+1})$. The *truncation error* can be shown to be $O(h^2 \cdot \|\frac{d^4 v}{dx^4}\|_\infty)$.
2. We may rewrite (18) at $x = x_i$ as

$$-v_{i-1} + 2v_i - v_{i+1} = h^2 f_i,$$

   where $0 < i < N + 1$.
3. Since the boundary conditions imply $v_0 = v_{N+1} = 0$, we have N equations in N unknowns $v_1, \ldots, v_N$:

$$T_N \cdot \begin{bmatrix} v_1 \\ \vdots \\ \vdots \\ v_N \end{bmatrix} = \begin{bmatrix} 2 & -1 & & 0 \\ -1 & \ddots & \ddots & \\ & \ddots & \ddots & -1 \\ 0 & & -1 & 2 \end{bmatrix} \cdot \begin{bmatrix} v_1 \\ \vdots \\ \vdots \\ v_N \end{bmatrix} = h^2 \begin{bmatrix} f_1 \\ \vdots \\ \vdots \\ f_N \end{bmatrix} \tag{19}$$

Eigenvalues and eigenvectors of T_N

The coefficient matrix T_N plays a central role in all that follows.

Lemma: Eigenvalues and eigenvectors of T_N

(a) The eigenvalues of T_N are $\lambda_j = 2\left(1 - \cos\frac{j\pi}{N+1}\right)$. (b) The eigenvectors are z_j, where $z_j(k) = \sqrt{\frac{2}{N+1}}\sin(jk\pi/(N+1))$. (c) z_j has unit 2-norm.

Proof. We use the trigonometric identity $\sin(\alpha+\beta)+\sin(\alpha-\beta) = 2\sin\alpha\cos\beta$:

$$\sin\left(\frac{j(k-1)\pi}{N-1}\right) + \sin\left(\frac{j(k+1)\pi}{N-1}\right) = 2\sin\left(\frac{jk\pi}{N-1}\right)\cos\left(\frac{j\pi}{N-1}\right),$$

which implies

$$\begin{aligned}(T_N z_j)(k) &= -\sin\left(\frac{j(k-1)\pi}{N-1}\right) + 2\sin\left(\frac{jk\pi}{N-1}\right) - \sin\left(\frac{j(k+1)\pi}{N-1}\right)\\ &= \left(2 - 2\cos\frac{j\pi}{N+1}\right)\sin\left(\frac{jk\pi}{N-1}\right) = \lambda_j z_j(k).\end{aligned}$$

This proves (a) and (b).

Eigenvalues and eigenvectors of T_N (cont.)

By $\cos(\alpha+\beta)-\cos(\alpha-\beta)=2\cos\alpha\cos\beta$:

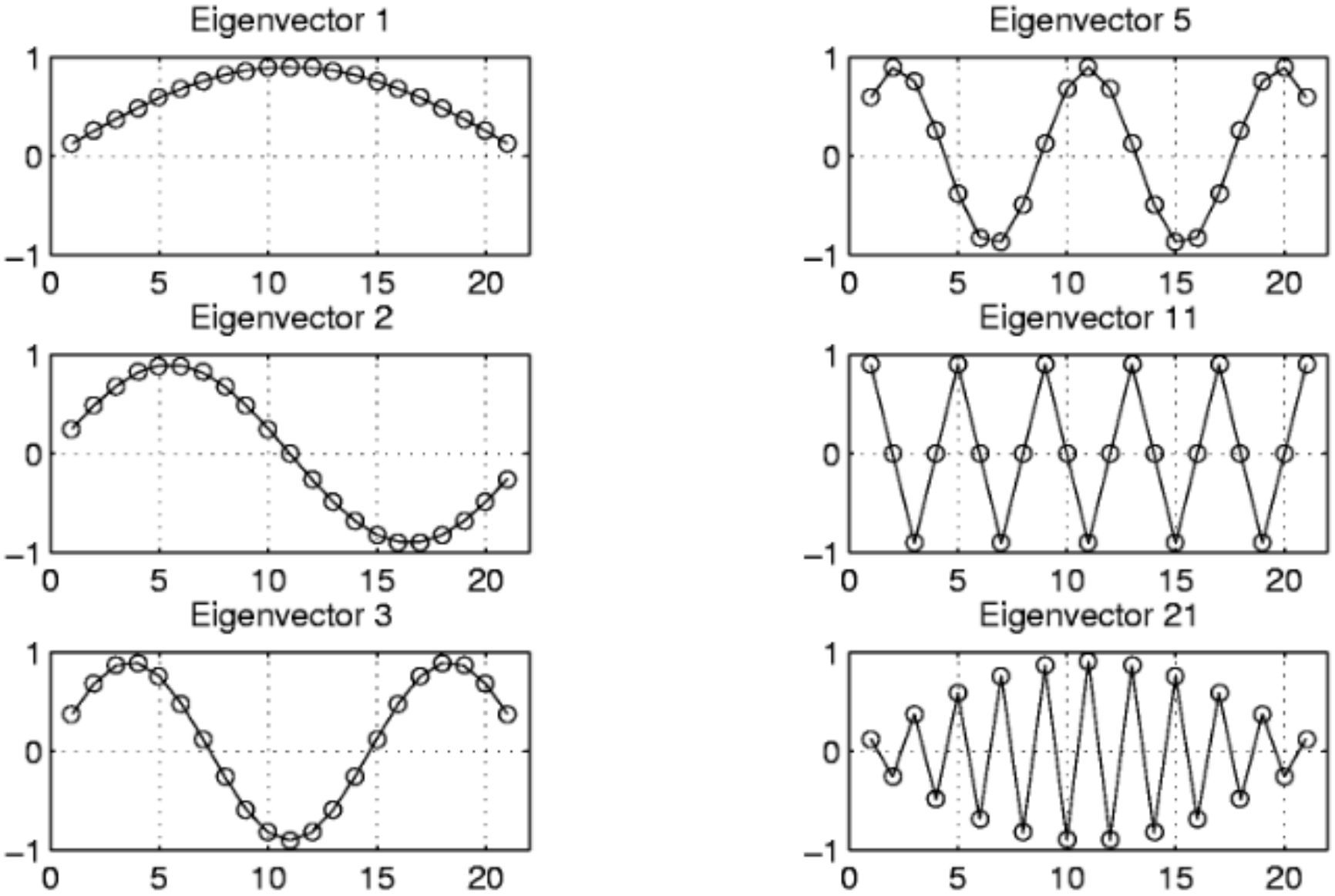
$$2\left(\sum_{k=0}^{N+1}\cos\frac{2jk\pi}{N+1}\right)\cos\frac{j\pi}{N+1}=\sum_{k=0}^{N+1}\left(\cos\frac{(2k+1)j\pi}{N+1}-\cos\frac{(2k-1)j\pi}{N+1}\right)$$
$$=\cos\frac{(2(N+1)+1)j\pi}{N+1}-\cos\frac{-j\pi}{N+1}=0.$$

Thus,

$$0=\sum_{k=0}^{N+1}\cos\frac{2jk\pi}{N+1}=\sum_{k=0}^{N+1}\left(1-2\sin^2\frac{jk\pi}{N+1}\right)=N+1-2\sum_{k=0}^{N+1}\sin^2\frac{jk\pi}{N+1}.$$

This shows $\|z_j\|_2=1$. □

Eigenvectors of T_N with $N = 21$



Eigenvalues and eigenvectors of $-\frac{d^2}{dx^2}$

- The eigenvector

$$z_j(k) = \sqrt{\frac{2}{N+1}} \sin(jk\pi/(N+1)),$$

is *precisely* equal to the eigenfunction $\hat{z}_i(x)$ evaluated at the sample points $x_j = jh$, when scaled by $\sqrt{\frac{2}{N+1}}$.

- When i is small compared to N, $\hat{\lambda}_i = i^2\pi^2$ is *well approximated* by

$$h^{-2}\lambda_i = (N+1)^2 \cdot 2\left(1 - \cos\frac{i\pi}{N+1}\right) = i^2\pi^2 + O((N+1)^{-2}).$$

- Here,

$$\lambda_i = 2\left(1 - \cos\frac{j\pi}{N+1}\right), \quad i = 1, \ldots, N$$

are the eigenvalues of T_N.

Poisson's equations and the Jacobi iteration

- Let $Z = [z_1, \ldots, z_N]$ be the orthogonal matrix whose columns are the eigenvectors of T_N.
- With $\Lambda = \operatorname{diag}(\lambda_1, \ldots, \lambda_N)$ we can write $T_N = Z\Lambda Z^T$.
- The smallest and the largest eigenvalues of T_N are

$$\lambda_1 = 2\left(1 - \cos\frac{\pi}{N+1}\right) \approx \frac{\pi^2}{(N+1)^2},$$

$$\lambda_N = 2\left(1 - \cos\frac{N\pi}{N+1}\right) \approx 4 - \frac{\pi^2}{(N+1)^2}$$

- For the eigenvalues of $I - D^{-1}T_N = I - \frac{1}{2}T_N$ yields

$$-1 < 1 - \lambda_i/2 < 1.$$

Theorem

The Jacobi iteration converges for T_N.

Subsection 3

Poisson's Equations in higher dimensions

Poisson's equations in two dimensions

Poisson's equation in two dimensions is

$$-\frac{\partial^2 v(x,y)}{\partial x^2} - \frac{\partial^2 v(x,y)}{\partial y^2} = f(x,y) \tag{20}$$

on the unit square

$$\{(x,y) \mid 0 < x, y < 1\},$$

with boundary conditions

$$v(x,0) = v(x,1) = v(0,y) = v(1,y) = 0, \quad 0 \le x, y \le 1.$$

We discretize at the grid points in the square which are at

$$(x_i, y_i), \quad \text{with} \quad x_i = ih,\ y_j = jh,\ h = \frac{1}{N+1}.$$

Discretization in two dimensions

From (18) we know that we can approximate

$$-\frac{\partial^2 v(x,y)}{\partial x^2}\bigg|_{x=x_i,y=y_j} \approx \frac{2v_{i,j} - v_{i-1,j} - v_{i+1,j}}{h^2},$$
$$-\frac{\partial^2 v(x,y)}{\partial y^2}\bigg|_{x=x_i,y=y_j} \approx \frac{2v_{i,j} - v_{i,j-1} - v_{i,j+1}}{h^2}.$$

Adding these approximations we have

$$-\frac{\partial^2 v(x,y)}{\partial x^2} - \frac{\partial^2 v(x,y)}{\partial y^2}\bigg|_{x=x_i,y=y_j} \approx \frac{4v_{i,j} - v_{i-1,j} - v_{i+1,j} - v_{i,j-1} - v_{i,j+1}}{h^2}. \quad (21)$$

The *truncation error* is bounded by $O(h^2)$. From the boundary conditions we know $v_{0j} = vN + 1, j = vi0 = vi, N + 1 = 0$. Thus, the discretization of (20) defines a set of $n = N^2$ linear equations in the n unknown v_{ij} for $1 \leq i, j \leq N$:

$$4v_{i,j} - v_{i-1,j} - v_{i+1,j} - v_{i,j-1} - v_{i,j+1} = h^2 f_{ij}. \quad (22)$$

Matrix equations

- Think of the unknowns v_{ij} as entries of an $N \times N$ matrix $V = (v_{ij})$ and similarly, the right hand sides $h^2 f_{ij}$ as an $T \times N$ matrix $h^2 F$.
- Notice that the ij-entries of the product matrices are

$$(T_N \cdot V)_{ij} = 2v_{ij} - v_{i-1,j} - v_{i+1,j},$$
$$(V \cdot T_N)_{ij} = 2v_{ij} - v_{i,j-1} - v_{i,j+1}.$$

- By adding these two equations and using $(h^2 F)_{ij} = h^2 f_{ij}$ equation (22) becomes the matrix equation

$$T_N \cdot V + V \cdot T_N = h^2 F. \tag{23}$$

- This is a linear system of equations for the entries of V.
- Analogously to eigenvectors, we say that V is an *eigenmatrix* with eigenvalue λ for the linear map $V \mapsto T_N V + V T_N$, if

$$T_N V + V T_N = \lambda V.$$

Eigenvectors and eigenvalues of the 2-dimensional system

- Let z_i, z_j be eigenvectors of T_N with eigenvalues λ_i, λ_j and put $V = z_i z_j^T$. Then,

$$\begin{aligned} T_N V + V T_N &= (T_N z_i) z_j^T + z_i (z_j^T T_N) \\ &= (\lambda_i z_i) z_j^T + z_i (z_j^T \lambda_j) \\ &= (\lambda_i + \lambda_j)(z_i z_j^T) \\ &= (\lambda_i + \lambda_j) V. \end{aligned}$$

- So for any $1 \le i, j \le N$, $V = z_i z_j^T$ is an eigenmatrix and $\lambda_i + \lambda_j$ the corresponding eigenvalue.
- As the system (22) is $n = N^2$ dimensional, we obtained all eigenvalues.
- The eigenfuctions of the 2-dimensional Poisson's equations are

$$\left(-\frac{\partial^2}{\partial x^2} - \frac{\partial^2}{\partial y^2} \right) \sin(i\pi x) \sin(j\pi y) = (i^2\pi^2 + j^2\pi^2) \sin(i\pi x) \sin(j\pi y).$$

Matrix form of the matrix equations

1. In order to represent the system (22) in usual matrix form, we write the unknowns v_{ij} in a single long $N^2 \times 1$ (column) vector $\boldsymbol{v}$.
2. We number the entries of $\boldsymbol{v}$ columnwise from the upper left to the lower right: $\boldsymbol{v}_1 = v_{11},\ \boldsymbol{v}_2 = v_{21},\ \ldots,\ \boldsymbol{v}_{(i-1)N+j} = v_{ij},\ \ldots,\ \boldsymbol{v}_{N^2} = v_{NN}$.
3. Let $T_{N\times N}$ be the matrix of $V \mapsto T_N V + V T_N$ in this coordinate frame.
4. For example, when $N = 3$ we can transform (22) to get

$$T_{3\times 3}\boldsymbol{v} = \left[\begin{array}{ccc|ccc|ccc} 4 & -1 & & -1 & & & & & \\ -1 & 4 & -1 & & -1 & & & & \\ & -1 & 4 & & & -1 & & & \\ \hline -1 & & & 4 & -1 & & -1 & & \\ & -1 & & -1 & 4 & -1 & & -1 & \\ & & -1 & & -1 & 4 & & & -1 \\ \hline & & & -1 & & & 4 & -1 & \\ & & & & -1 & & -1 & 4 & -1 \\ & & & & & -1 & & -1 & 4 \end{array}\right] \begin{bmatrix} \boldsymbol{v}_1 \\ \boldsymbol{v}_2 \\ \vdots \\ \vdots \\ \vdots \\ \boldsymbol{v}_9 \end{bmatrix} = h^2 \begin{bmatrix} f_1 \\ f_2 \\ \vdots \\ \vdots \\ \vdots \\ f_9 \end{bmatrix}.$$

$T_{N\times N}$ in block matrix form

We can rewrite $T_{N\times N}$ as a block matrix having N $N\times N$ blocks of the form $T_N + 2I$ on its diagonal and $-I_N$ blocks on its offdiagonals:

$$T_{N\times N} = \begin{bmatrix} T_N + 2I_N & -I_N & & \\ -I_N & \ddots & \ddots & \\ & \ddots & \ddots & -I_N \\ & & -I_N & T_N + 2T_N \end{bmatrix} \tag{24}$$

$$= \begin{bmatrix} T_N & & & \\ & \ddots & & \\ & & \ddots & \\ & & & T_N \end{bmatrix} + \begin{bmatrix} 2I_N & -I_N & & \\ -I_N & \ddots & \ddots & \\ & \ddots & \ddots & -I_N \\ & & -I_N & T_N \end{bmatrix}.$$

The Kronecker product of matrices

Definition: The Kronecker product of matrices

Let $A = (a_{ij})$ be an $m \times n$ matrix and B a $p \times q$ matrix. Then $A \otimes B$, the Kronecker product of A and B, is the $mp \times nq$ matrix

$$\begin{bmatrix} a_{11} \cdot B & \cdots & a_{1n} \cdot B \\ \vdots & & \vdots \\ a_{m1} \cdot B & \cdots & a_{mn} \cdot B \end{bmatrix}.$$

Proposition: Properties of the Kronecker product

1. Assume that the product AC and BD of matrices are well defined. Then

$$(A \otimes B) \cdot (C \otimes D) = (AC) \otimes (BD).$$

2. If A and B are invertible then $(A \otimes B)^{-1} = A^{-1} \otimes B^{-1}$.
3. $(A \otimes B)^T = A^T \otimes B^T$.

Poisson's equations with Kronecker products

Definition

Let X be $m \times n$. Then $\mathrm{vec}(X)$ is defined to be a column vector of size mn made of the columns of X stacked atop one another from left to right.

Lemma

1. $\mathrm{vec}(AX) = (I_n \otimes A) \cdot \mathrm{vec}(X)$.
2. $\mathrm{vec}(XB) = (B^T \otimes I_m) \cdot \mathrm{vec}(X)$.
3. The Poisson equation $T_N V + V T_N = h^2 F$ is equivalent to

$$T_{N\times N} \cdot \mathrm{vec}(V) \equiv (I_N \otimes T_N + T_N \otimes I_N) \cdot \mathrm{vec}(V) = h^2 \cdot \mathrm{vec}(F). \tag{25}$$

Proof. We only prove (3). The Poisson equation is clearly equivalent to

$$\mathrm{vec}(T_N V + V T_N) = \mathrm{vec}(T_N V) + \mathrm{vec}(V T_N) = \mathrm{vec}(h^2 F).$$

By (1), $\mathrm{vec}(T_N V) = (I_N \otimes T_N) \cdot \mathrm{vec}(V)$. By (2) and the symmetry of T_N, $\mathrm{vec}(V T_N) = (T_N^T \otimes I_N)\,\mathrm{vec}(V) = (T_N \otimes I_N)\,\mathrm{vec}(V)$. Adding these implies (3). □

The eigendecomposition of $T_{N\times N}$

Theorem: The eigendecomposition of $T_{N\times N}$

Let $T_N = Z\Lambda Z^T$ be the eigendecomposition of T_N, with $Z = [z_1, \ldots, z_N]$ the orthogonal matrix whose columns are eigenvectors, and $\Lambda = \text{diag}(\lambda_1, \ldots, \lambda_N)$. Then:

1. The eigendecomposition of $T_{N\times N} = I_N \otimes T_N + T_N \otimes I_N$ is

$$I \otimes T_N + T_N \otimes I = (Z \otimes Z) \cdot (I \otimes \Lambda + \Lambda \otimes I) \cdot (Z \otimes Z)^T. \tag{26}$$

2. $I \otimes \Lambda + \Lambda \otimes I$ is a diagonal matrix whose $((i-1)N + j)$th diagonal entry, the (i,j)th eigenvalue of $T_{N\times N}$, is $\lambda_{i,j} = \lambda_i + \lambda_j$.
3. $Z \otimes Z$ is an orthogonal matrix whose $((i-1)N + j)$th column, the corresponding eigenvector, is $z_i \otimes z_j$.

Proof. We use (1) and (3) from the previous Lemma. It is easy to see that $Z \otimes Z$ is orthogonal:

$$(Z \otimes Z)(Z \otimes Z)^T = (Z \otimes Z)(Z^T \otimes Z^T) = ZZ^T \otimes ZZ^T = I \otimes I = I.$$

The eigendecomposition of $T_{N\times N}$ (cont.)

Also, it is easy to verify that $I \otimes \Lambda + \Lambda \otimes I$ is diagonal, with entries $\lambda_i + \lambda_j$. (26) follows from

$$\begin{aligned}(Z \otimes Z) \cdot (I \otimes \Lambda + \Lambda \otimes I) \cdot (Z \otimes Z)^T &= (Z \otimes Z) \cdot (I \otimes \Lambda + \Lambda \otimes I) \cdot (Z^T \otimes Z^T) \\ &= (ZIZ^T) \otimes (Z\Lambda Z^T) + (Z\Lambda Z^T) \otimes ZIZ^T \\ &= I \otimes T_N + T_N \otimes I \\ &= T_{N\times N}\end{aligned}$$

Finally, from the definition of the Kronecker product, one can see that column $(i-1)N + j$ of $Z \otimes Z$ is $z_i \otimes z_j$. □

Remark. Similarly, Poisson's equation in three dimensions leads to

$$T_{N\times N\times N} = T_N \otimes I_N \otimes I_N + I_N \otimes T_N \otimes I_N + I_N \otimes I_N \otimes T_N,$$

with eigenvalues all possible triple sums of the λ_i's, and eigenvector matrix $Z \otimes Z \otimes Z$. Poisson's equation in higher dimensions work analogously.

Demo: Maple program with $N = 21$

It takes a few seconds to make 1000 iterations and obtain a 4-digit exact solution:

```
T_NxN:=KroneckerProduct(T_N,I_N)+KroneckerProduct(I_N,T_N):
F_NxN := N^2 vector discretizing f(x,y)=x^2+y^2

H:=1.0-T_NxN/T_NxN[1,1]:
F:=F_NxN/T_NxN[1,1]:

V := Vector(N^2):      # The solution vector
for i from 1 to 1000 do
    V0 := copy(V):
    V := H.V0 + F:      # The Jacobi iteration step
    print(i,Norm(V-V0),Norm(T_NxN.V-F_NxN)):
end do:
```

Other solvers in Maple

We get more information by setting higher infolevel.

```
infolevel[LinearAlgebra]:=3;
```

The standard method uses **LU decomposition.** Very fast.

```
LinearSolve(T_NxN,F_NxN):
```

We can force Maple to use **sparse iteration method.** Very fast.

```
LinearSolve(T_NxN,F_NxN,method=SparseIterative):
```

We can **invert** a matrix of this size using floating number arithmetic.
However, the inversion in rational arithemtic is very slow.

```
MatrixInverse(1.0*T_NxN).F_NxN:          # fast
MatrixInverse(T_NxN).F_NxN:              # very slow
```

Section 6

Linear Programming

Subsection 1

Linear Inequalities

Resource Allocation Problem

A company produces two products: chairs and tables. They make a profit of \$40 on each chair and \$50 on each table. A chair requres the following resources to produce: 2 man-hours, 3 hours of machine time, and 1 unit of wood. The table requires 2 man-hours, 1 hour of machine time, and 4 units of wood. The factory has 60 man-hours, 75 machine hours, and 84 units of wood available each day.

How should the resources (man-hours, machine-hours, and wood) be allocated between the two products in order to maximize the factory's profit?

$$\text{maximize} \quad 40c + 50t \qquad \textit{(objective function)}$$

$$\text{such that} \quad \begin{cases} 2c + 2t \leq 60 \\ 3c + t \leq 75 \\ c + 4t \leq 84 \\ c, t \geq 0 \end{cases} \qquad \textit{(constraints or bounds)}$$

Blending Problem

A feed company wants each feed bag that they produce to contain a minimum of 120 units of protein and 80 units of calcium. Corn contains 10 units of protein and 5 units of calcium per pound, and bone-meal contains 2 units of protein and 5 units of calcium per pound.
If corn costs 8 cents per pound and bone-meal costs 4 cents per pound, *how much of each should they put in each bag, in order to minimize costs?*

$$\begin{array}{lll} \text{minimize} & 8c + 4b & \textit{(objective function)} \\ \text{such that} & \begin{cases} 10c + 2b \geq 120 \\ 5c + 5b \geq 80 \\ c, b \geq 0 \end{cases} & \textit{(constraints or bounds)} \end{array}$$

Transportation Problem

1. A bulldozer company has two warehouses (A and B) and three stores (1, 2 and 3).
2. The first warehouse has 40 bulldozers in stock and the second has 20.
3. The three stores have 25, 10, and 22 bulldozers, respectively, on order.
4. If C_{WS} is used to represent the cost to transport a bulldozer from warehouse W to store S, we know that

$$C_{A1} = 550,\ C_{A2} = 300,\ C_{A3} = 400,\ C_{B1} = 350,\ C_{B2} = 300,\ C_{B3} = 100.$$

5. *Determine the routing that will satisfy the needs of the stores, at minimum cost.*

Transportation Problem: The variables

1. Let X_{WS} be the number of bulldozers transported from warehouse W to store S.
2. We want to minimize the objective function

$$C_{A1}X_{A1} + C_{A2}X_{A2} + C_{A3}X_{A3} + C_{B1}X_{B1} + C_{B2}X_{B2} + C_{B3}X_{B3}.$$

3. In praxis, X_{WN} can only take **nonnegative integer values**.
4. The nonnegativity constraints are clearly

$$X_{A1}, X_{A2}, X_{A3}, X_{B1}, X_{B2}, X_{B3} \geq 0. \qquad \text{(T1)}$$

5. We **forget** about the condition of being integers. This would cause enormous difficulties.
6. Remark. This technique of „forgetting” is called *constraint relaxation.*

Transportation Problem: The constraints

1. The constraints

$$\begin{aligned} X_{A1} + X_{A2} + X_{A3} &\leq 40 \\ X_{B1} + X_{B2} + X_{B3} &\leq 20 \end{aligned} \tag{T2}$$

   state that the number of bulldozers leaving each warehouse cannot exceed the warehouse capacity.

2. The constraints

$$\begin{aligned} X_{A1} + X_{B1} &= 25 \\ X_{A2} + X_{B2} &= 10 \\ X_{A3} + X_{B3} &= 22 \end{aligned} \tag{T3}$$

   state that the number of bulldozers arriving at the store must be equal to the number ordered.

3. Actually, the number arriving must be *at least* as many as ordered. However, the minimum cost will clearly not specify that we deliver more bulldozers than ordered. *(No relaxation.)*

Curve Fitting in L_∞-norm

Definition: L_∞-norm

The L_∞-norm of the vector $\boldsymbol{x} = (x_1, \ldots, x_N) \in \mathbb{R}^N$ is $\|\boldsymbol{x}\|_\infty = \max\{|x_1|, \ldots, |x_N|\}$.

- We want to find the straight line $y = mx + b$ that best fits the data points

$$(u_1, v_1), \ldots, (u_N, v_N)$$

  in the L_∞-norm.
- In other words, we want to find m, b, ε such that

$$|mu_i + b - v_i| \leq \varepsilon \qquad \text{for all } i = 1, \ldots, N,$$

  and, ε is as small as possible.
- Using objective function and constraints:

$$\varepsilon \to \min, \text{ where } -\varepsilon \leq mu_i + b - v_i \leq \varepsilon \text{ for all } i = 1, \ldots, N.$$

Curve Fitting in L_1-norm

Definition: L_1-norm

The L_1-norm of the vector $\boldsymbol{x} = (x_1, \ldots, x_N) \in \mathbb{R}^N$ is $\|\boldsymbol{x}\|_1 = |x_1| + \cdots + |x_N|$.

- We want to find the best fitting line $y = mx + b$ in the L_1-norm.
- That is, we want to find $m, b, \varepsilon_1, \ldots, \varepsilon_N$ such that

$$|mu_i + b - v_i| \leq \varepsilon_i \qquad \text{for all } i = 1, \ldots, N,$$

  and, $\varepsilon_1 + \cdots + \varepsilon_N$ is as small as possible.
- Using objective function and constraints:

$$\varepsilon_1 + \cdots + \varepsilon_N \to \min, \text{ where } -\varepsilon_i \leq mu_i + b - v_i \leq \varepsilon_i \text{ for all } i = 1, \ldots, N.$$

- Sometimes in practice, the „errors" $mu_i + b - v_i$ cannot take negative values. Then the constraints become $0 \leq mu_i + b - v_i \leq \varepsilon_i$.

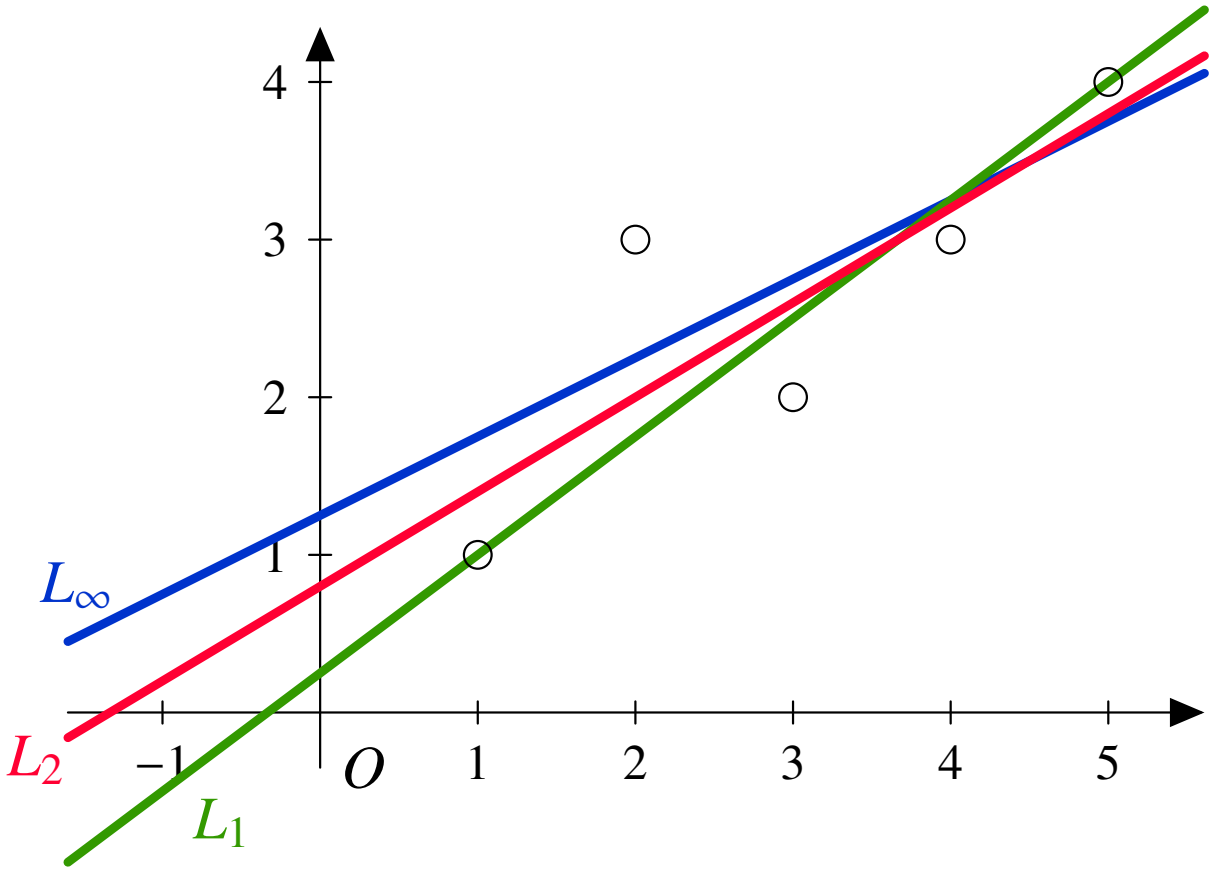
Example of curve fitting

The points are:
$(1, 1)$, $(2, 3)$, $(3, 2)$,
$(4, 3)$, $(5, 4)$.
The best fit lines are:

(L_2) $\quad y = 0.6\,x + 0.8$

(L_∞) $\quad y = 0.5\,x + 1.25$

(L_1) $\quad y = 0.75\,x + 0.25$



Linear Programming (LP) problems

The inequality formulation for LP problems is the following:

$$\begin{aligned} \text{minimize/maximize} \quad & c_1 x_1 + \cdots + c_n x_n \\ \text{subject to} \quad & a_{11} x_1 + \cdots + a_{1n} x_n \geq b_1 \\ & \vdots \\ & a_{m1} x_1 + \cdots + a_{mn} x_n \geq b_m \end{aligned}$$

Using summation notation:

$$\begin{aligned} \text{minimize/maximize} \quad & \sum_{j=1}^{n} c_j x_j \\ \text{subject to} \quad & \sum_{j=1}^{n} a_{ij} x_j \geq b_i \qquad i = 1, \ldots, m \end{aligned}$$

Remark. Equality constraints are replaced by two inequalities.

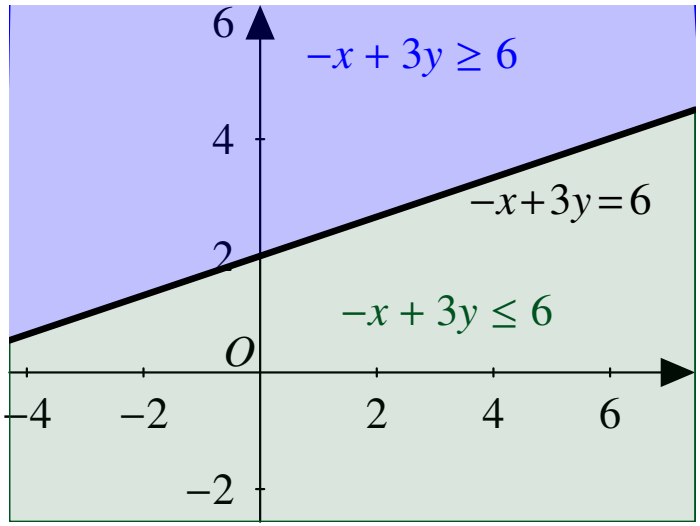
Advantages of Linear Programming

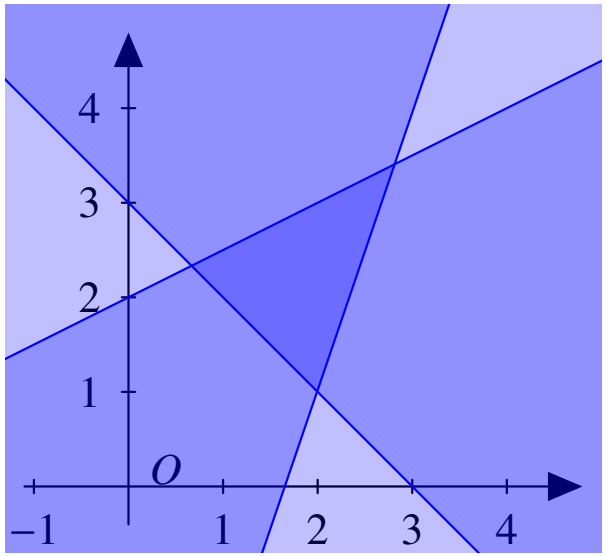
1. LP problems can be solved very efficiently (several 10,000 variables and constraints)
2. LP has an extensive, useful mathematical theory (optimality conditions, sensitivity analysis, ...)
3. Widely available high-quality software:
   - Computational environments (MATLAB, MAPLE, Octave, Mathematica) have optimization tools
   - Free software: GLPK, lpsolve, CVXOPT
   - Built-in solvers for MS Office, LibreOffice and Google Spreadsheets

Subsection 2

Geometric solutions of LP problems

Geometry of systems of linear inequalities





1. In a Cartesian coordinate system, the equation $a_1x + a_2y = b$ determines a line ℓ.
2. The inequalities

$$a_1x + a_2y \leq b$$
$$a_1x + a_2y \geq b$$

determine the halfplanes ℓ^+, ℓ^- of ℓ.

3. The system of linear inequalities

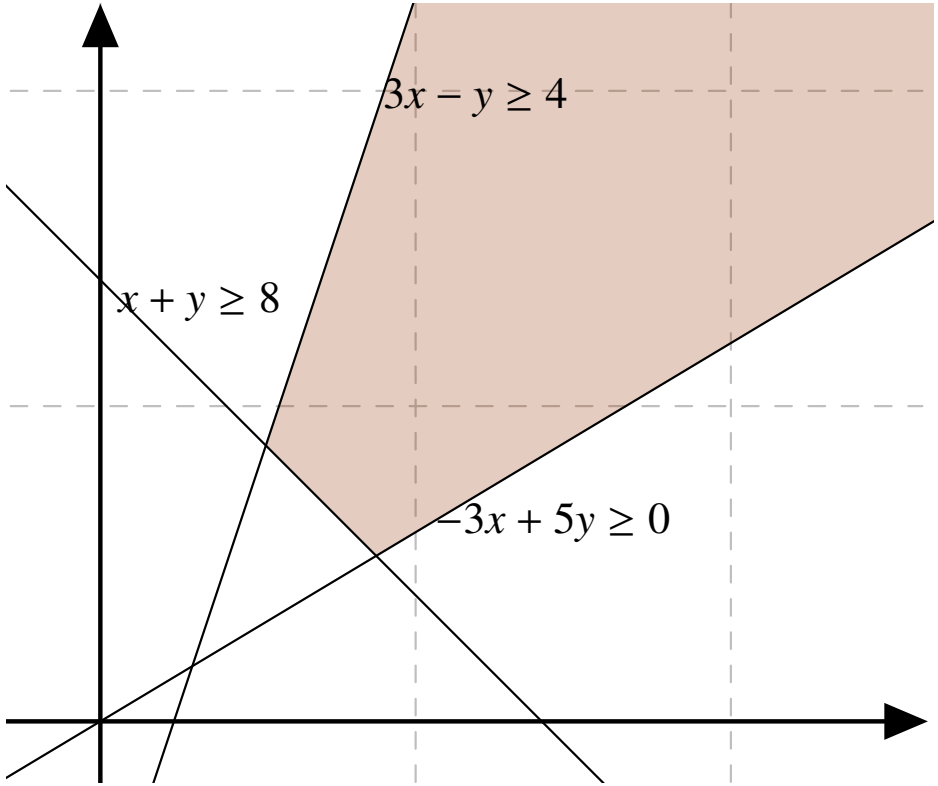
$$\begin{cases} a_{11}x + a_{12}y \geq b_1 \\ \vdots \\ a_{m1}x + a_{m2}y \geq b_m \end{cases}$$

determine the intersection of halfplanes.

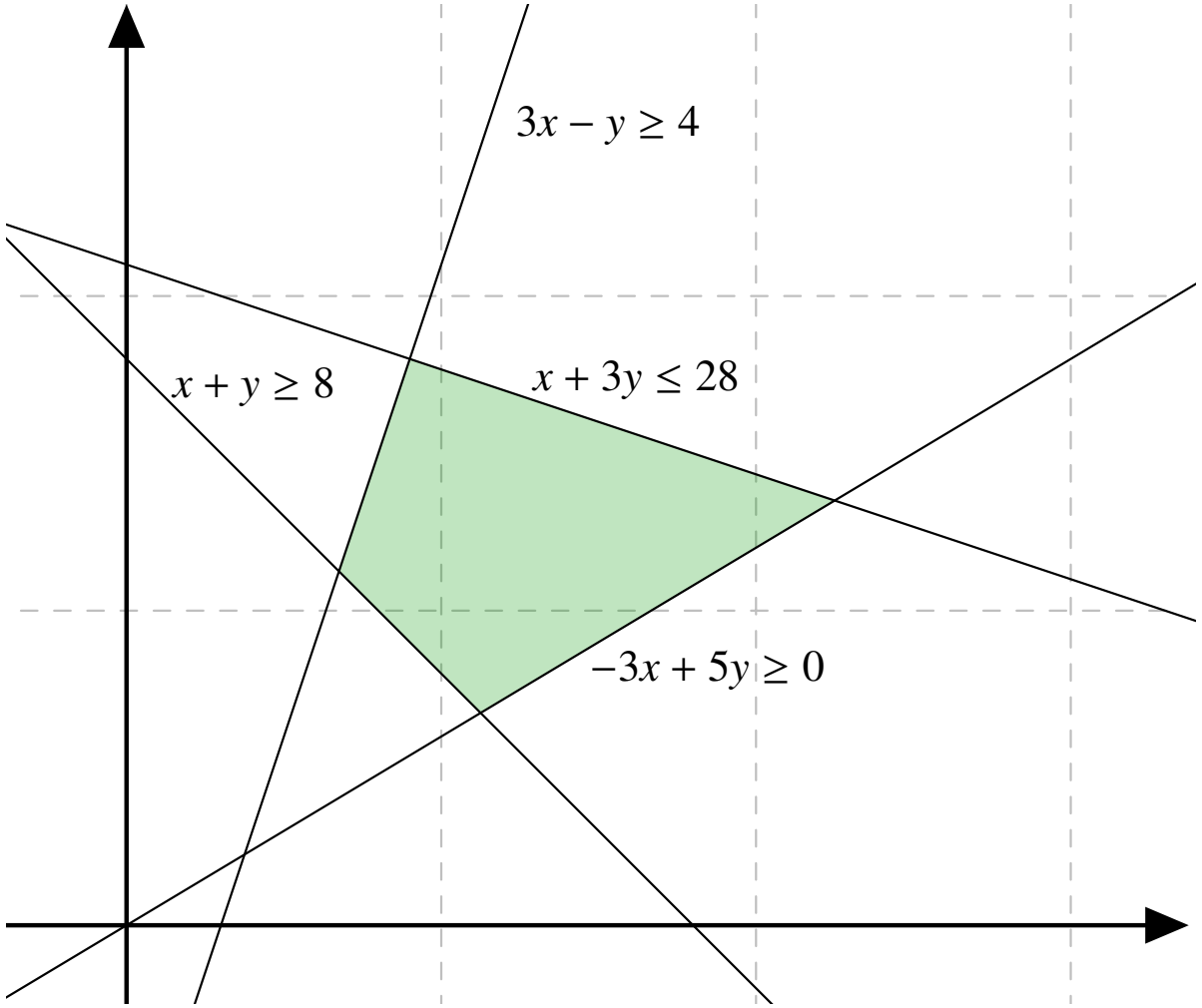
Example of infinite feasibility domain

Definition: Feasibility domain/Feasible region

A vector $(x_1, \ldots, x_n)$ satisfying the constraints of an LP problem is called a *feasible point*. The set of feasible points is the *feasibility domain* or *feasible region*.

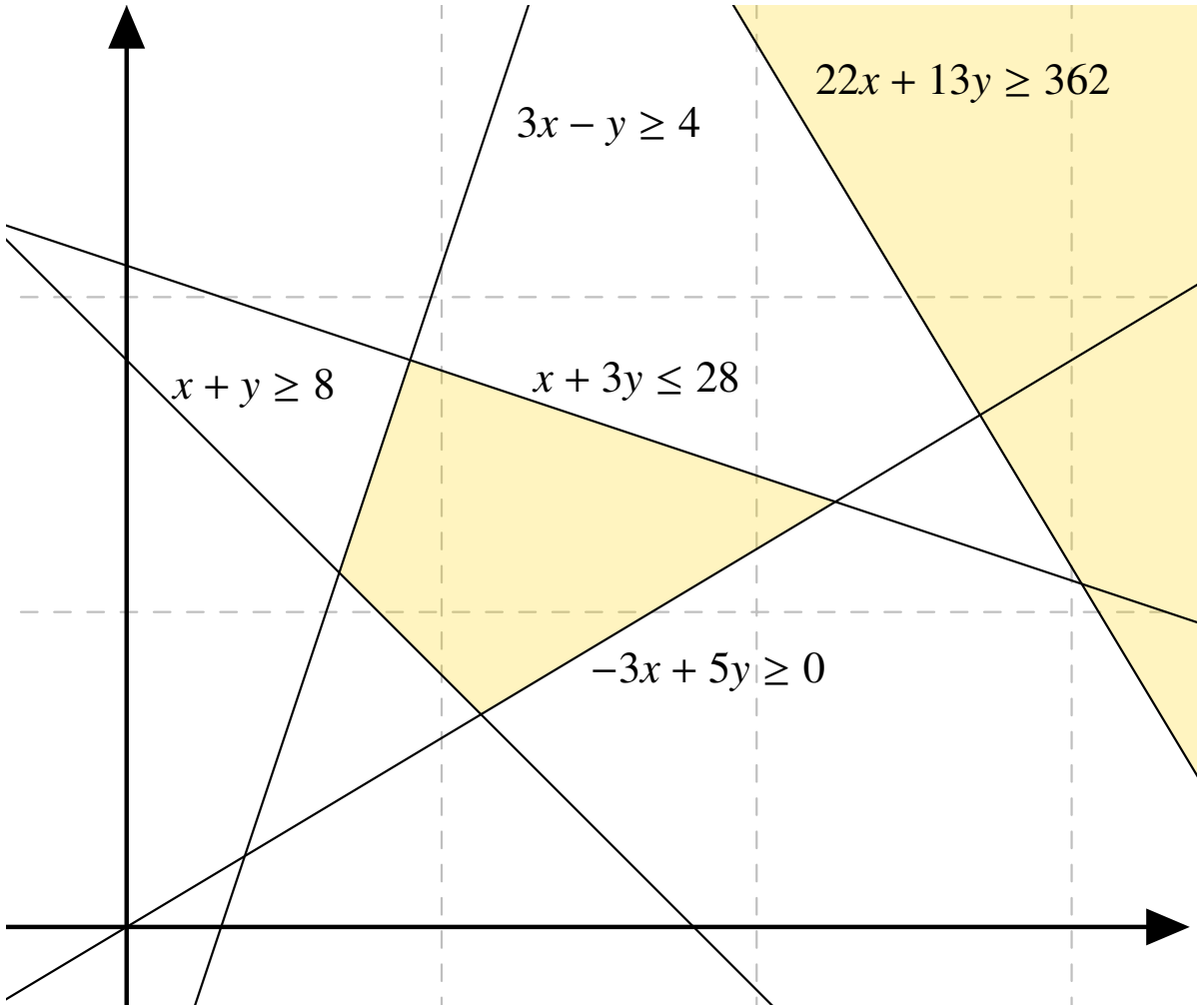


Example of finite feasibility domain



Example of empty feasibility domain (infeasible)

The intersection of the two yellow domains is empty:



Geometry of objective functions

1. The equations

$$c_1x + c_2y = d$$
$$c_1x + c_2y = d'$$

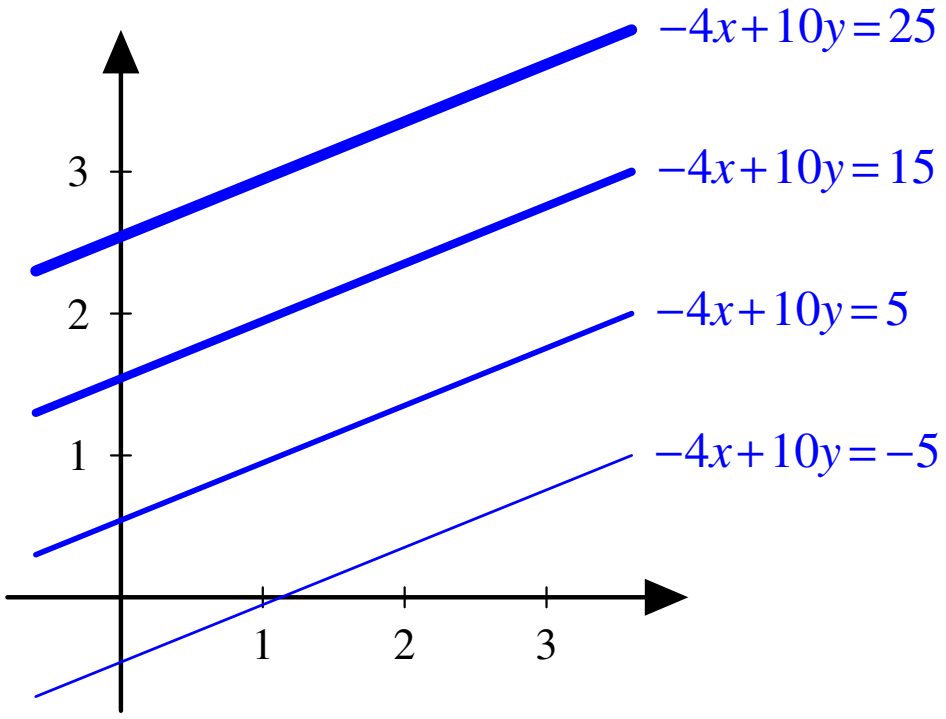
determine parallel lines.

2. The objective function

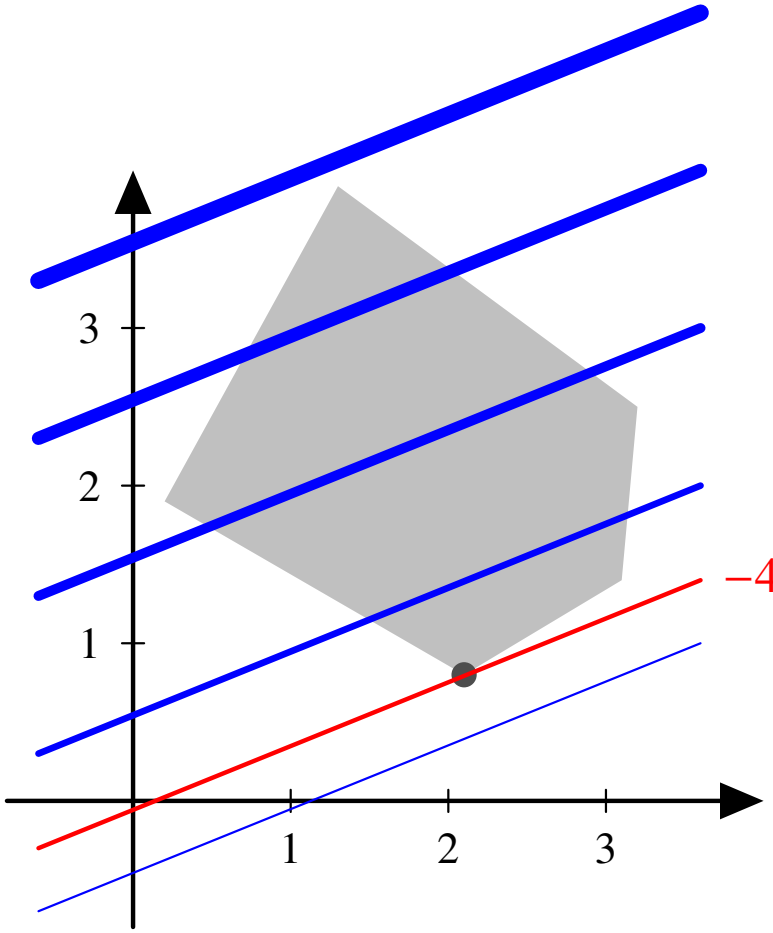
$$f(x, y) = c_1x + c_2y$$

corresponds to a parallel class of lines.

3. The evaluation in P corresponds to the element of the class through P.



LP problems in two variables

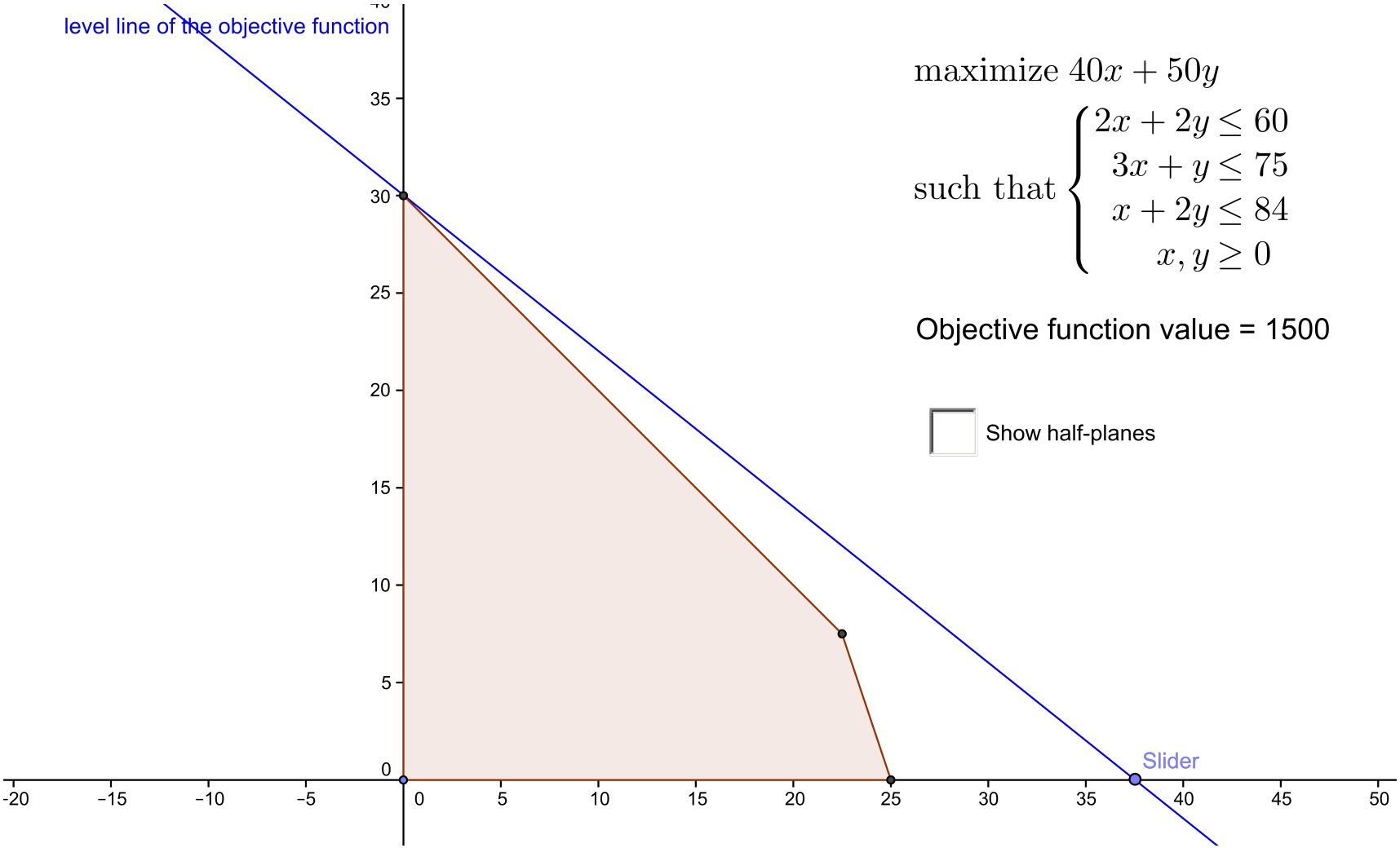


Task: Minimize $-4x + 10y$ on the grey domain of feasibility.

Solution: Among the parallel lines $-4x + 10y = d$, find the one which „supports" the gray domain „from below".

Observe: The optimum must be a vertex (or side) of the gray polygon!

Geometric solution of the Resource Allocation Problem



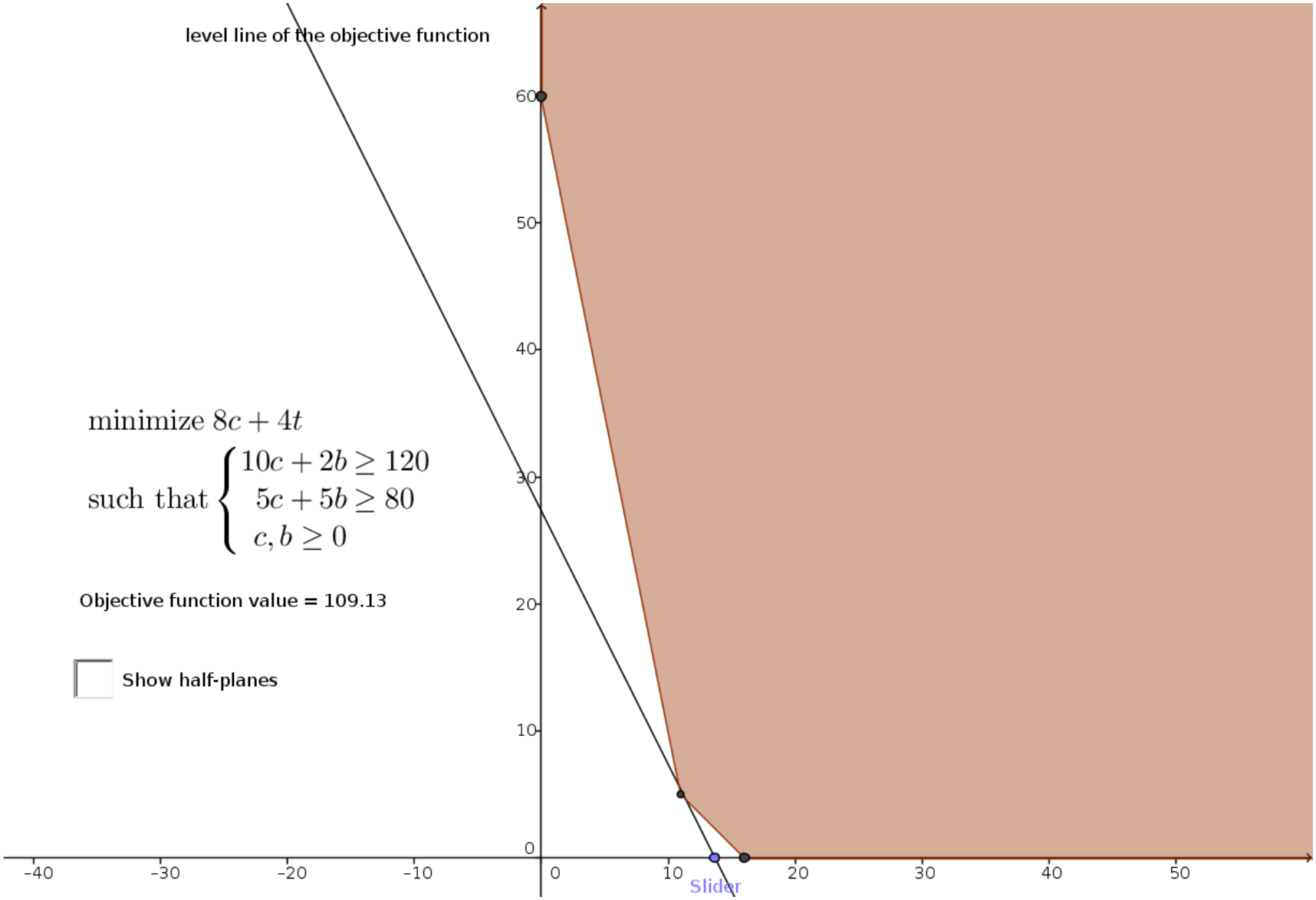
$$\text{maximize } 40x + 50y$$

$$\text{such that } \begin{cases} 2x + 2y \leq 60 \\ 3x + y \leq 75 \\ x + 2y \leq 84 \\ x, y \geq 0 \end{cases}$$

Objective function value = 1500

☐ Show half-planes

Geometric solution of the Blending Problem



Terminology of LP problems

1. It its most general form, an LP problem can be written as follows:

$$\begin{aligned} \text{minimize/maximize} \quad & c_1 x_1 + \cdots + c_n x_n \\ \text{subject to} \quad & a_{11} x_1 + \cdots + a_{1n} x_n \geq b_1 \\ & \vdots \\ & a_{m1} x_1 + \cdots + a_{mn} x_n \geq b_m \end{aligned}$$

2. By multiplying the objective function with -1, one can swich between minimization and maximization.
3. $\boldsymbol{x}^* = (x_1^*, \ldots, x_n^*)$ is an *optimal solution,* if it is feasible and the objective function is minimal/maximal.
4. For a *minimization problem* the latter means

$$c_1 x_1^* + \cdots + c_n x_n^* \leq c_1 x_1 + \cdots + c_n x_n$$

   for all feasible points $(x_1, \ldots, x_n)$.
5. The value $c_1 x_1^* + \cdots + c_n x_n^*$ is called the *optimum* of the LP problem.

Feasibility

Concerning the solvability of LP problems, we have the following possibilities:

Infeasible: The feasibility domain is **empty.**

Feasible: The feasibility domain is **not empty** (finite or infinite), and, the objective function **does take its extremum** on the feasibility domain.

Unbounded: The feasibility domain is **infinite,** and the objective function **does not take its extremum** on the feasibility domain.

Example of unbounded problem

$$\begin{aligned} \text{maximize} \quad & x_1 - 4x_2 \\ \text{subject to} \quad & -2x_1 + x_2 \leq -1 \\ & -x_1 - 2x_2 \leq -2 \\ & x_1, x_2 \geq 0 \end{aligned}$$

For any $x_1 \geq 2$, $(x_1, 0)$ is a feasible point and as x_1 gets large the objective function does too.

Hence, the problem is unbounded.

Subsection 3

Duality Theory

Equivalence of LP problems

Definition: Equivalence of LP problems

Let us consider two LP problems with variables $x_1, \ldots, x_n$ and $y_1, \ldots, y_m$. We say that the two LP problems are *equivalent,* if their optima are equal, and, there is a „nice" one-to-one mapping between their optimal solutions.

The standard form of LP problems

Primal standard form:

$$\begin{cases} \text{maximize} & \displaystyle\sum_{j=1}^{n} c_j x_j \\ \text{subject to} & \displaystyle\sum_{j=1}^{n} a_{ij} x_j \le b_i \quad i = 1, \ldots, m \\ & x_j \ge 0 \quad j = 1, \ldots, n \end{cases} \tag{PS}$$

Dual standard form:

$$\begin{cases} \text{minimize} & \displaystyle\sum_{i=1}^{m} b_i y_i \\ \text{subject to} & \displaystyle\sum_{i=1}^{m} y_i a_{ij} \ge c_j \quad j = 1, \ldots, n \\ & y_i \ge 0 \quad i = 1, \ldots, m \end{cases} \tag{DS}$$

The general form of LP problems

Primal general form:

$$\begin{cases} \text{maximize} & \displaystyle\sum_{j=1}^{n} c_j x_j \\ \text{subject to} & \displaystyle\sum_{j=1}^{n} a_{ij} x_j = b_i \quad i = 1, \ldots, m \\ & x_j \geq 0 \quad j = 1, \ldots, n \end{cases} \tag{PG}$$

Dual general form:

$$\begin{cases} \text{minimize} & \displaystyle\sum_{i=1}^{m} b_i y_i \\ \text{subject to} & \displaystyle\sum_{i=1}^{m} y_i a_{ij} \geq c_j \quad j = 1, \ldots, n \end{cases} \tag{DG}$$

Equivalence of primal standard and general forms

1. If a LP problem is given in the primal general form, then it can be transformed to primal standard form by *replacing the constraint* $\sum_j a_{ij}x_j = b_i$ with the two inequalities

$$\sum_j a_{ij}x_j \le b_i \quad \text{and} \quad \sum_j -a_{ij}x_j \le -b_i.$$

2. If a LP problem is given in the primal standard form (PS), then it can be transformed to primal general form by *adding the slack variables*

$$w_i = b_i - \sum_j a_{ij}x_j \qquad (i = 1, \ldots, m).$$

Then:

$$\begin{cases} \text{maximize} & \sum_j c_j x_j & \\ \text{subject to} & \sum_j a_{ij}x_j + w_i = b_i & i = 1, \ldots, m \\ & w_i, x_j \ge 0 & i = 1, \ldots, m, \\ & & j = 1, \ldots, n \end{cases}$$

Duality

Definition: Duality of LP problems

We say that the LP problem (DS) is the *dual* of the standard LP problem (PS).

- One can show that the dual of the dual of a standard LP problem is the problem itself.
- In order to achieve this, one has to write the dual problem (DS) in primal standard form.
- Example of dual problems:

$$(1)\begin{cases} \max. & x_1 + 2x_2 \\ \text{sub. to} & 4x_1 + 3x_2 \leq 1 \\ x_1, x_2 \geq 0 \end{cases} \quad \text{and} \quad (2)\begin{cases} \min. & y_1 \\ \text{sub. to} & 4y_1 \geq 1;\ 3y_1 \geq 2 \\ y_1 \geq 0 \end{cases}$$

Weak Duality

Weak Duality Theorem

If $\boldsymbol{x} = (x_1, \ldots, x_n)$ is a feasible point of the standard primal LP problem (PS), and $\boldsymbol{y} = (y_1, \ldots, y_m)$ is a feasible point for its dual problem (DS), then

$$\sum_j c_j x_j \leq \sum_i b_i y_i.$$

Proof. (Easy.) Using $x_j, y_i \geq 0$ and the constraints on c_j, b_i's, we have

$$\sum_j c_j x_j \leq \sum_j \left(\sum_i y_i a_{ij} \right) x_j = \sum_{i,j} y_i a_{ij} x_j = \sum_i \left(\sum_j a_{ij} x_j \right) y_i \leq \sum_i b_i y_i. \quad \square$$

Strong Duality Theorem

The main result of the Duality Theory of Linear Programming is the

Strong Duality Theorem

If the primal standard problem has an optimal solution $\boldsymbol{x}^* = (x_1^*, \ldots, x_n^*)$, then the dual also has an optimal solution $\boldsymbol{y}^* = (y_1^*, \ldots, y_m^*)$ such that

$$\sum_j c_j x_j^* = \sum_i b_i y_i^*.$$

Proof. Hard. □

We will present two of the many applications of the Strong Duality Theorem.

Optimality of solutions

1. For general optimization problems, it is almost as difficult to *prove the optimality* of a given solution then to *find* an optimal solution.
2. Most of the LP solving methods produce the optimal solutions for the primal and dual problems at the same time.

Application

Assume that the solutions $(x_1^*, \ldots, x_n^*)$, $(y_1^*, \ldots, y_m^*)$ of the standard primal and dual problems are given. In order to prove their optimality, one has to check that both are feasible, and show

$$\sum_j c_j x_j^* = \sum_i b_i y_i^*.$$

Feasibility of dual problems

1. The Weak Duality Theorem implies that any feasible solution of the standard dual problem implies a bound for the primal problem.
2. Therefore, if the primal problem is unbounded, then the dual must be infeasible.
3. In fact, only 4 possibilities can occur:

Corollary of the Strong Duality Theorem

One of the following holds:

1. Both (PS) and (DS) are feasible.
2. (PS) is unbounded and (DS) is infeasible.
3. (PS) is infeasible and (DS) is unbounded.
4. Both (PS) and (DS) are infeasible.

The general form of LP problems (repetitorium)

We have already seen that LP problems in *primal standard* form are equivalent with problems in *primal general* form:

Primal general form:

$$\begin{cases} \text{maximize} & \displaystyle\sum_{j=1}^{n} c_j x_j \\ \text{subject to} & \displaystyle\sum_{j=1}^{n} a_{ij} x_j = b_i \quad i = 1, \ldots, m \\ & x_j \geq 0 \quad j = 1, \ldots, n \end{cases}$$

Dual general form:

$$\begin{cases} \text{minimize} & \displaystyle\sum_{i=1}^{m} b_i y_i \\ \text{subject to} & \displaystyle\sum_{i=1}^{m} y_i a_{ij} \geq c_j \quad j = 1, \ldots, n \end{cases}$$

We now show that (PG) and (DG) are *dual* to each other.

Duals of general form LP problems

Proposition

The general form LP problems (PG) and (DG) are *dual* to each other.

Proof. First we replace the equality constraints of (PG) by inequalities:

$$\begin{aligned}
\text{maximize} \quad & \sum_{j=1}^{n} c_j x_j \\
\text{subject to} \quad & \sum_{j=1}^{n} a_{ij} x_j \le b_i \qquad i = 1, \ldots, m \\
& \sum_{j=1}^{n} -a_{ij} x_j \le -b_i \quad i = 1, \ldots, m \\
& x_j \ge 0 \qquad j = 1, \ldots, n
\end{aligned} \tag{27}$$

Now, the problem is in standard form with n variables and $2m + n$ constraints.

Duals of general form LP problems (cont.)

Let us denote the dual variables associated with the first set of m constraints by y_i^+, and the remaining dual variables by y_i^-, $i = 1, \ldots, m$.
Then the dual problem is

$$\begin{aligned} \text{minimize} \quad & \sum_{i=1}^{m} b_i y_i^+ - \sum_{i=1}^{m} b_i y_i^- \\ \text{subject to} \quad & \sum_{i=1}^{m} y_i^+ a_{ij} - \sum_{i=1}^{m} y_i^- a_{ij} \geq c_j \quad j = 1, \ldots, n \\ & y_i^+, y_i^- \geq 0 \quad i = 1, \ldots, m \end{aligned} \tag{28}$$

Duals of general form LP problems (cont.)

If we put

$$y_i = y_i^+ - y_i^-, \qquad i = 1, \ldots, m, \tag{29}$$

the dual problem reduces to the dual general form

$$\begin{aligned} \text{minimize} \quad & \sum_{i=1}^{m} b_i y_i \\ \text{subject to} \quad & \sum_{i=1}^{m} y_i a_{ij} \geq c_j \quad j = 1, \ldots, n \end{aligned} \tag{30}$$

In fact, (28) and (30) are equivalent:

1. If $(\boldsymbol{y}^+, \boldsymbol{y}^-)$ is feasible for (28), then (29) defines a feasible point of (30).
2. Conversely, let $(y_1, \ldots, y_m)$ be feasible for (30) and define

$$y_i^+ = \begin{cases} y_i & \text{if } y_i \geq 0 \\ 0 & \text{otherwise} \end{cases} \qquad y_i^- = \begin{cases} y_i & \text{if } y_i < 0 \\ 0 & \text{otherwise.} \end{cases}$$

Then, $(\boldsymbol{y}^+, \boldsymbol{y}^-)$ is feasible for (28). *The objective function values are equal.*

Subsection 4

The Simplex Method

Brief history of the Simplex Method

1. **1940s** (Dantzig, Kantorovich, Koopmans, von Neumann, ...): foundations of LP, motivated by economic and logistics problems of WWII
2. **1947** (Dantzig): simplex algorithm, published in 1949
3. **1950s–60s** applications in other fields (structural optimization, control theory, filter design, ...)
4. **1979** (Khachiyan) ellipsoid algorithm: more efficient (polynomial-time) than simplex in worst case, much slower in practice
5. **1984** (Karmarkar): projective (interior-point) algorithm: polynomial-time worst-case complexity, and efficient in practice
6. **1984–today** variations of interior-point methods (improved complexity or efficiency in practice), software for large-scale problems

Overview

1. We have seen that the four formulations (PS), (DS), (PG), (DG) of LP problems are equivalent.
2. The input of the simplex algorithm is the standard primal formulation.

$$\textbf{(PS)} \begin{cases} \text{maximize} & \displaystyle\sum_{j=1}^{n} c_j x_j \\ \text{subject to} & \displaystyle\sum_{j=1}^{n} a_{ij} x_j \le b_i \quad i = 1, \ldots, m \\ & x_j \ge 0 \quad j = 1, \ldots, n \end{cases}$$

The algorithm proceeds in two steps:

3. **Phase I:** (Initialization.) We transfer the problem into another LP problem which is in standard primal formulation and the constants $b_1, \ldots, b_m$ are nonnegative.
4. **Phase II:** By *iteration,* we solve the problem with nonnegative constants.

Phase II: Illustration on an example

We illustrate how the simplex method works on a specific example:

$$\begin{aligned} \text{maximize} \quad & 5x_1 + 4x_2 + 3x_3 \\ \text{subject to} \quad & 2x_1 + 3x_2 + x_3 \leq 5 \\ & 4x_1 + x_2 + 2x_3 \leq 11 \\ & 3x_1 + 4x_2 + 2x_3 \leq 8 \\ & x_1, x_2, x_3 \geq 0 \end{aligned} \tag{31}$$

We start by adding so-called *slack variables*

$$\begin{aligned} w_1 &= 5 - 2x_1 - 3x_2 - x_3 \geq 0 \\ w_2 &= 11 - 4x_1 - x_2 - 2x_3 \geq 0 \\ w_3 &= 8 - 3x_1 - 4x_2 - 2x_3 \geq 0 \end{aligned}$$

Initial feasible solution

We get the equivalent formulation of (31)

$$\begin{aligned} \text{maximize} \quad & \zeta = 5x_1 + 4x_2 + 3x_3 \\ \text{subject to} \quad & w_1 = 5 - 2x_1 - 3x_2 - x_3 \\ & w_2 = 11 - 4x_1 - x_2 - 2x_3 \\ & w_3 = 8 - 3x_1 - 4x_2 - 2x_3 \\ & x_1, x_2, x_3, w_1, w_2, w_3 \geq 0 \end{aligned} \tag{32}$$

1. The simplex method is an **iterative process** in which we start with a solution $x_1, x_2, \ldots, w_3$ that satisfies the equations in (32)
2. and then look for a **new solution** $\bar{x}_1, \bar{x}_2, \ldots, \bar{w}_3$,
3. which is better in the sense that it has a **larger objective function value:**

$$5\bar{x}_1 + 4\bar{x}_2 + 3\bar{x}_3 > 5x_1 + 4x_2 + 3x_3.$$

4. To start the iteration, we need the **initial feasible solution**

$$(x_1, x_2, x_3, w_1, w_2, w_3) = (0, 0, 0, 5, 11, 8).$$

First improvement

1. We rewrite the equations in (32) as

$$\begin{aligned} \zeta &= 5x_1 + 4x_2 + 3x_3 \\ \hline w_1 &= 5 - 2x_1 - 3x_2 - x_3 \\ w_2 &= 11 - 4x_1 - x_2 - 2x_3 \\ w_3 &= 8 - 3x_1 - 4x_2 - 2x_3 \end{aligned} \tag{33}$$

2. Note that we introduced ζ for the value of the objective function.
3. We ask whether the initial solution can be improved.
4. Since the coefficient of x_1 is $5 > 0$, if we increase x_1, we will increase ζ.
5. As $x_2 = x_3 = 0$, the limits on the increment are

$$w_1 = 5 - 2x_1 \geq 0, \qquad w_2 = 11 - 4x_1 \geq 0, \qquad w_3 = 8 - 3x_1 \geq 0.$$

6. In other words, $x_1 \leq 2.5$, $x_1 \leq 2.75$ and $x_1 \leq 2.66$ must hold.
7. Our new, improved bound is $(x_1, x_2, x_3, w_1, w_2, w_3) = (2.5, 0, 0, 0, 1, 0.5)$.

Row operations on the equations

1. Our new, improved bound is $(x_1, x_2, x_3, w_1, w_2, w_3) = (2.5, 0, 0, 0, 1, 0.5)$.
2. We observe that we still have 3 *zero and* 3 *nonzero valiables.*
3. The „zero" variables are called the **independent,** the others **dependent.**
4. Indeed, w_1 **„entered"** and x_1 **„left"** the *set of independent variables*
5. We express x_1 by the *independent* variables w_1, x_2, x_3:

$$x_1 = 2.5 - 0.5w_1 - 1.5x_2 - 0.5x_3.$$

6. The new equations are

$$\begin{aligned} \zeta &= 12.5 - 2.5w_1 - 3.5x_2 + 0.5x_3 \\ \hline x_1 &= 2.5 - 0.5w_1 - 1.5x_2 - 0.5x_3 \\ w_2 &= 1 + 2w_1 + 5x_2 \\ w_3 &= 0.5 + 1.5w_1 + 0.5x_2 - 0.5x_3 \end{aligned}$$

7. Remark. We can recover our current solution by setting the independent variables to 0.

Second iteration

1. We repeat the present form of the LP problem:

$$\begin{aligned} \zeta &= 12.5 - 2.5w_1 - 3.5x_2 + 0.5x_3 \\ \hline x_1 &= 2.5 - 0.5w_1 - 1.5x_2 - 0.5x_3 \\ w_2 &= 1 + 2w_1 + 5x_2 \\ w_3 &= 0.5 + 1.5w_1 + 0.5x_2 - 0.5x_3 \end{aligned} \tag{34}$$

2. The only variable of the objective function with positive coefficient is x_3.
3. The nonnegativity of the dependent variables implies

$$2.5 - 0.5x_3 \geq 0, \qquad 1 \geq 0, \qquad 0.5 - 0.5x_3 \geq 0.$$

4. Therefore, $x_3 = 1$. The new feasible solution is

$$(x_1, x_2, x_3, w_1, w_2, w_3) = (2, 0, 1, 0, 1, 0).$$

5. The „entering” variable is w_3 and the „leaving” variable is x_3.

End of the algorithm

1. We use the last equation of (35) to express x_3 as
$$x_3 = 1 + 3w_1 + x_2 - 2w_3.$$
2. The result of the appropriate row operations is the system
$$\begin{aligned} \zeta &= 13 \quad - w_1 - 3x_2 \quad - w_3 \\ \hline x_1 &= \quad 2 - 2w_1 - 2x_2 \quad + w_3 \\ w_2 &= \quad 1 + 2w_1 + 5x_2 \\ x_3 &= \quad 1 + 3w_1 \quad + x_2 - 2w_3 \end{aligned} \tag{35}$$
3. Here, **there is no** variable in the objective function, for which an increase would produce a corresponding increase in ζ.
4. **The iteration ends.**
5. Since (35) is completely equivalent to (31), the current solution $(x_1, x_2, x_3, w_1, w_2, w_3) = (2, 0, 1, 0, 1, 0)$ is **optimal.**
6. The optimum is $\zeta = 13$.

Terminology

1. The systems of equation (33)–(35) are called *dictionaries*. (Chvátal 1983)
2. Dependent variables are also called *basic* variables. Independent variables are *nonbasic*.
3. The solutions we have obtained by setting the nonbasic variables to 0 are called *basic feasible solutions*.
4. The step from one dictionary to the next is called a *pivot*.
5. There is often more than one choice for the entering and the leaving variables. Particular rules that make the choice unambiguous are called *pivot rules*.
6. Most texts describe the Simplex Method as a sequence of pivots on a table of numbers called the *simplex tableau*.

Phase I: Initialization

1. In order to start the previously presented Simplex Method, we needed an **initial feasible solution.**
2. This was easy to find provided **the right-hand sides** of the problem were all **nonnegative.**
3. **If not,** then we introduce an *auxiliary problem*

$$\text{original} \begin{cases} \max & \sum_{j=1}^{n} c_j x_j \\ \text{s. t.} & \sum_{j=1}^{n} a_{ij} x_j \le b_i \\ & x_j \ge 0 \end{cases} \qquad \text{auxiliary} \begin{cases} \max & -x_0 \\ \text{s. t.} & \sum_{j=1}^{n} a_{ij} x_j - x_0 \le b_i \\ & x_j \ge 0 \end{cases}$$

4. A initial feasible solution is $x_0 = \max_i b_i$ and $x_j = 0$ $(j = 1, \ldots, n)$.
5. The original problem **has a feasible solution** if and only if the auxiliary problem has **objective value** 0**.**

Example of an auxiliary problem

1. We illustrate with an example the use of the auxiliary problem:

$$\begin{cases} \text{maximize} & -2x_1 - x_2 \\ \text{subject to} & -x_1 + x_2 \leq -1 \\ & -x_1 - 2x_2 \leq -2 \\ & x_2 \leq 1 \\ & x_1, x_2 \geq 0. \end{cases} \qquad \begin{cases} \text{maximize} & -x_0 \\ \text{subject to} & -x_1 + x_2 - x_0 \leq -1 \\ & -x_1 - 2x_2 - x_0 \leq -2 \\ & x_2 - x_0 \leq 1 \\ & x_0, x_1, x_2 \geq 0. \end{cases} \tag{36}$$

1. The auxiliary problem (36) still has negatives on the right-hand side.
2. Using one row operation, we will be able to convert (36) into a system with nonnegative right-hand sides.

Example of an auxiliary problem (cont.)

1. We change to the usual notation and introduce the slack variables:

$$\begin{aligned} \xi &= & & & -x_0 \\ \hline w_1 &= -1 + x_1 & & - x_2 + x_0 \\ w_2 &= -2 + x_1 & + 2x_2 & + x_0 \\ w_3 &= \quad 1 & & - x_2 + x_0 \end{aligned} \tag{37}$$

2. We do **one pivot** with (leaving) variable x_0 and the *„most infeasible variable"* w_2:

$$\begin{aligned} \xi &= -2 + x_1 & + 2x_2 & - w_2 \\ \hline w_1 &= \quad 1 & & - 3x_2 + w_2 \\ x_0 &= \quad 2 - x_1 & & - 2x_2 + w_2 \\ w_3 &= \quad 3 - x_1 & & - 3x_2 + w_2 \end{aligned} \tag{38}$$

3. As (37) has nonnegative constant terms, we can proceed as before.

Row operations of an auxiliary problem

1. For the first step, we pick x_2 to enter and w_1 to leave the basis:

$$\begin{aligned} \xi &= -1.33 + x_1 - 0.67w_1 - 0.33w_2 \\ \hline x_2 &= 0.33 - 0.33w_1 + 0.33w_2 \\ x_0 &= 1.33 - x_1 + 0.67w_1 + 0.33w_2 \\ w_3 &= 2 - x_1 + w_1 \end{aligned} \tag{39}$$

2. For the second step, we pick x_1 to enter and x_0 to leave the basis:

$$\begin{aligned} \xi &= -x_0 \\ \hline x_2 &= 0.33 - 0.33w_1 + 0.33w_2 \\ x_1 &= 1.33 - x_0 + 0.67w_1 + 0.33w_2 \\ w_3 &= 0.67 + x_0 + 0.33w_1 - 0.33w_2 \end{aligned} \tag{40}$$

3. This system is optimal with objective function value $\xi = -x_0 = 0$.
4. This implies that the **original problem has feasible points!**

End of the initialization

1. We now drop x_0 from (40) and reintroduce the original objective function

$$\zeta = -2x_1 - x_2 = -3 - w_1 - w_2.$$

2. Hence, the starting feasible dictionary for the original problem is

$$\begin{aligned} \zeta &= -3 - w_1 - w_2 \\ \hline x_2 &= 0.33 - 0.33w_1 + 0.33w_2 \\ x_1 &= 1.33 + 0.67w_1 + 0.33w_2 \\ w_3 &= 0.67 + 0.33w_1 - 0.33w_2 \end{aligned} \tag{41}$$

3. *We are lucky,* this system is optimal for the original problem.
4. The *optimum* is $\zeta = -3$ with *solution*

$$(x_1, x_2, w_1, w_2, w_3) = (1.33, 0.33, 0, 0, 0.67).$$

5. *In general,* we continue with Phase II and apply the Simplex Method as explained.

Unboundedness

1. We have seen that the infeasibility of an LP problem can be noticed during the initialization phase, when the optimal value of x_0 is positive.
2. The unboundedness can be detected during the simplex algorithm as shown in the following example:

$$\begin{array}{rl} \zeta = 5 & + x_3 \quad - x_1 \\ \hline x_2 = 5 + 2x_3 & - 3x_1 \\ x_1 = 7 & - 4x_1 \\ w_3 = & x_1 \end{array} \tag{42}$$

3. Here, the entering variable is x_3 and the bounds on the increment (given by $x_1 = 0$) are

$$5 + 2x_3 \geq 0, \qquad 7 \geq 0, \qquad 0 \geq 0.$$

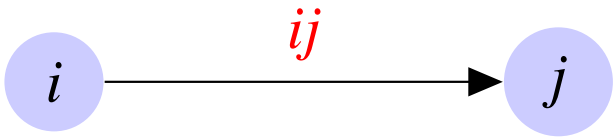
4. This implies that arbitrarily large x_3 yields a feasible point, hence the problem is unbounded.

Subsection 5

Other applications and generalizations

Networks and graphs

1. A **network** consists of two types of objects: *nodes* and *arcs*.
2. The nodes are connected by arcs. In the present context, arcs are assumed to be *directed*.
3. The arc connecting node i to node j is an *ordered pair* (i, j) which we will denote simply as ij.
4. We say that ij is an *in-arc* for j and on *out-arc* for i.



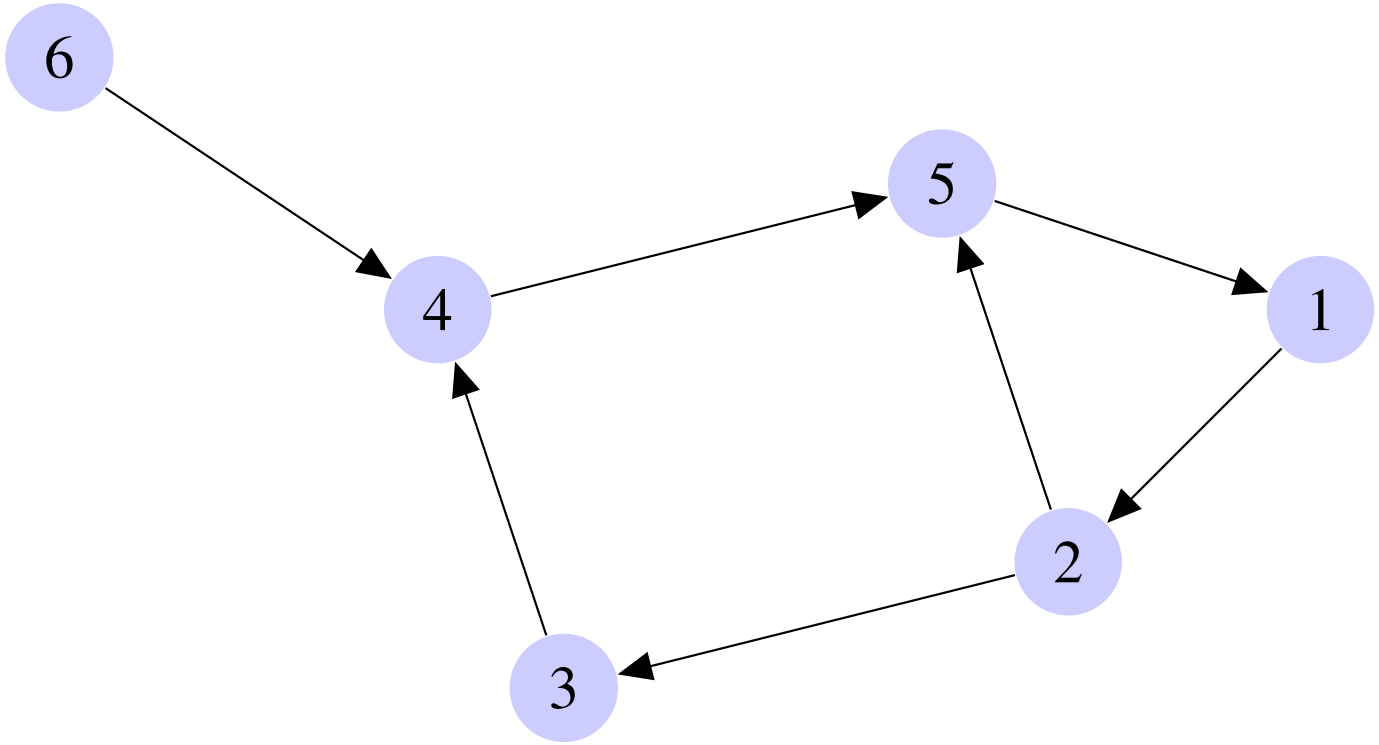
5. Let $\mathcal{N}$ denote the set of nodes.
6. Let $\mathcal{A}$ denote the set of arcs; this is a subset of all possible arcs:

$$\mathcal{A} \subseteq \{ij \mid i, j \in \mathcal{N}, i \neq j\}.$$

7. The pair $(\mathcal{N}, \mathcal{A})$ is called a *network*. In mathematics, it is also called a *graph*, a *directed graph*, or a *digraph*.

Example of a network

- $\mathcal{N} = \{1, 2, \ldots, 6\}$
- $\mathcal{A} = \{12, 23, 25, 34, 45, 51, 64\}$



- The set of in-arcs of node 5 is $\{45, 25\}$.
- The set of out-arcs of node 2 is $\{23, 25\}$.

Source, target, capacity

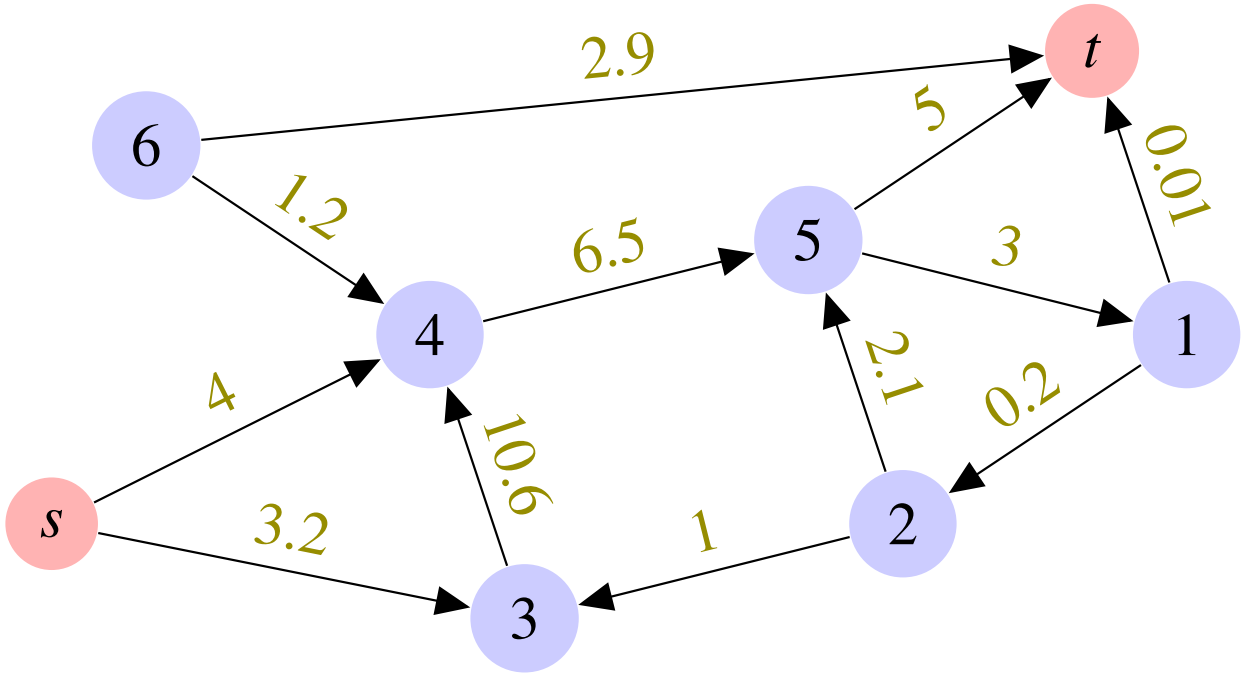
In our network $(\mathcal{N}, \mathcal{A})$ we highlight two specific nodes:

- One node is called the **source,** which is denoted by s.
- One node is called the **target,** which is denoted by t.

Moreover, we assume that a *capacity function*

$$c : \mathcal{A} \to \mathbb{R}^+$$

is assigned to our network.



Flows

1. Our aim is to *maximize the „flow"* from the source to the target through the networks such that the capacity constraints are kept.
2. A network flow is formalized as follows.
3. The function $f : \mathcal{A} \to \mathbb{R}$ is called a *flow,* if for all nodes $i \neq s, t$ the *conservation law*

$$\sum_{k,\, ki \in \mathcal{A}} f(ki) = \sum_{k,\, ik \in \mathcal{A}} f(ik) \qquad \text{(Cons[i])}$$

   holds.
4. The flow f is *admissible* if for any arc $ij \in \mathcal{A}$,

$$0 \leq f(ij) \leq c(ij). \qquad \text{(Adm[a])}$$

5. The *value* of the flow f is measured by the function

$$\mathrm{Val}(f) = \sum_{k,\, kt \in \mathcal{A}} f(kt) - \sum_{k,\, tk \in \mathcal{A}} f(tk) = \sum_{k,\, ks \in \mathcal{A}} f(ks) - \sum_{k,\, sk \in \mathcal{A}} f(sk).$$

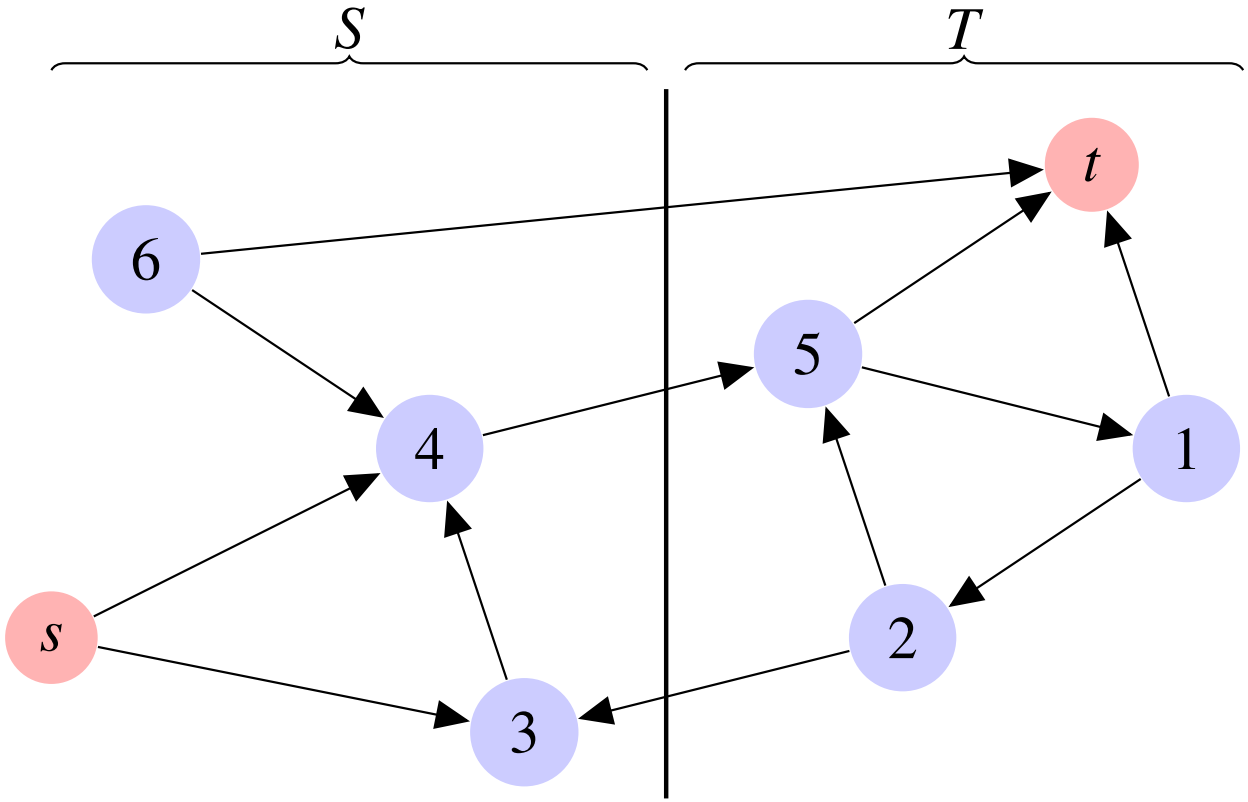
Maximal Flow Problem

The maximal flow of the network is the solution of the following LP problem:

$$\begin{array}{lll} \text{maximize} & \text{Val}(f) & \\ \text{subject to} & \text{(Cons[i])} \quad i \in \mathcal{N} & \text{(MaxFlow)} \\ & \text{(Adm[a])} \quad a \in \mathcal{A} & \end{array}$$

Source-target cuts

1. Let $\mathcal{N} = S \dot{\cup} T$ be a disjoint partition of $\mathcal{N}$ such that $s \in S$ and $t \in T$.
2. Then we call the pair $\Sigma = (S, T)$ a *source-target cut* of the network.



3. We denote by Σ^+ (or by Σ^-) the arcs from S to T (or from T to S).
4. In our example $\Sigma^+ = \{45, 6t\}$ and $\Sigma^- = \{32\}$.

Minimal cuts and the easy inequality

1. We define the *capacity of a source-target cut* Σ by

$$\mathrm{Cap}(\Sigma) = \sum_{ij \in \Sigma^+} c(ij).$$

2. The **Minimal Cut Problem** is to determine the cut of the network with minimum capacity.
3. It is easy to show that for any flow f and any source-target cut Σ the following inequality holds:

$$\mathrm{Val}(f) \leq \mathrm{Cap}(\Sigma). \tag{43}$$

4. In fact, one has **equality** in (43).
5. This is rather surprising, since source-target cuts form a *discrete* set, while flows are *continuous*.

The „Min Cut Max Flow" duality

„Min Cut Max Flow" Theorem

Let f^* be a maximal flow and Σ^* a minimal cut in the network $(\mathcal{N}, \mathcal{A})$. Then

$$\mathrm{Val}(f^*) = \mathrm{Cap}(\Sigma^*). \tag{44}$$

Proof. We use the Duality Theory of Linear Programming. The dual LP problem of (MaxFlow) is

$$\begin{aligned} \text{minimize} \quad & \sum_{ij \in \mathcal{A}} C(ij) w_{ij} \\ \text{subject to} \quad & u_j - u_i + w_{ij} \geq 0 \qquad ij \in \mathcal{A} \\ & u_s - u_t \geq 1 \\ & w_{ij} \geq 0 \qquad ij \in \mathcal{A} \end{aligned} \tag{MinCut}$$

The hard part is to show that in the solution $u_i^*, w_{ij}^* \in \{0, 1\}$. Then, $S = \{i \mid u_i^* = 1\}$ and $T = \{j \mid u_j^* = 0\}$ is a minimal cut. □

The Assignment Problem

- Suppose that a taxi firm has m cars available, ans n customers wishing to be picked up. The cost for car i to pick up customer j is given in the $m \times n$ matrix $A = (a_{ij})$.
- The problem is to find an assigment

$$u : \{1, \ldots, m\} \to \{1, \ldots, n\}$$

between taxis and customers such that the total cost

$$C_u = \sum_{i=1}^{m} a_{i,u(i)} \qquad \text{is minimal.}$$

- We relax this problem by looking for a solution matrix $X = (x_{ij})$ such that

$$0 \leq x_{ij} \leq 1, \quad \sum_i x_{ij} = 1, \quad \sum_k x_{ij} = 1, \tag{45}$$

and $\sum a_{ij}x_{ij}$ is minimal.
- The Integrality Lemma ensures that in the optimal solution the entries x_{ij} are 0 or 1.

The Stable Marriage Problem

- Given n men and n women, where each person has ranked all members of the opposite sex with an unique number between 1 and n in order of preference.
- Marry the men and women together such that there are no two people of opposite sex who would both rather have each other than their current partners.
- If there are no such people, all the marriages are **stable.**
- Beside the constrains (45), we have the stability constrains

$$\sum_{j >_m w} x_{m,j} + \sum_{i >_w m} x_{i,w} + x_{m,w} \geq 1$$

  for all man-woman pairs (m, w).
- By theorems of Vate (1989)and Rothblum (1992), the optimal LP solution is integer.

The Hungarian Method

- The Hungarian Method is a combinatorial optimization algorithm that solves the Assignment Problem in polynomial time and which anticipated later general primal-dual methods for LP problems.
- It was developed and published by Harold Kuhn in 1955, who gave the name „Hungarian Method" because the algorithm was largely based on the earlier works of two Hungarian mathematicians: Dénes Kőnig and Jenő Egerváry.
- The Hungarian Method can be generalized for the Min-Cut-Max-Flow problem resulting the Ford-Fulkerson algorithm.

Generalizations of Linear Programming

- Fractional Linear Programming
- Convex Programming
- Quadratic Programming
- Semidefinite Programming (SDP)
- Integer Programming (IP)
- Mixed Integer Programming (MIP)

Section 7

The Discrete Fourier Transform

The Discrete Fourier Transform

Euler's formula

For any real number φ

$$\exp(i\varphi) = e^{i\varphi} = \cos\varphi + i\sin\varphi.$$

The Discrete Fourier Transform (DFT)

Let $\mathbf{f}$ be a (complex) $N \times 1$ vector. The Discrete Fourier Transform of $\mathbf{f}$ is $\mathbf{y} = A\mathbf{f}$, where $A = A_N$ is an $N \times N$ matrix with $a_{kj} = \exp(2i\pi(k-1)(j-1)/N)$. In other words,

$$y_k = \sum_{j=1}^{N} f_j \exp\left(\frac{2i\pi(k-1)(j-1)}{N}\right).$$

Inverse DFT

Lemma

Define A as above. Then $A^{-1} = \bar{A}/N$, where $\bar{A}$ is the complex (elementwise) conjugate of A.

Observe, that

$$\mathbf{f} = A^{-1}\mathbf{v} = (\bar{A}\mathbf{y})/N = \overline{\left(A\frac{\bar{\mathbf{y}}}{N}\right)}.$$

That is, we may calculate the inverse of the DFT, if we take DFT of $\bar{\mathbf{y}}/N$, and then conjugate the result.

Example

Let $\mathbf{f} = [2, 16, 32, 128]^T$. Then the DFT of $\mathbf{f}$ is

$$\begin{bmatrix} 178 \\ -30-112i \\ -100 \\ -30+112i \end{bmatrix} = \begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & i & -1 & -i \\ 1 & -1 & 1 & -1 \\ 1 & -i & -1 & i \end{bmatrix} \cdot \begin{bmatrix} 2 \\ 16 \\ 32 \\ 128 \end{bmatrix}$$

The intuitive meaning of the DFT

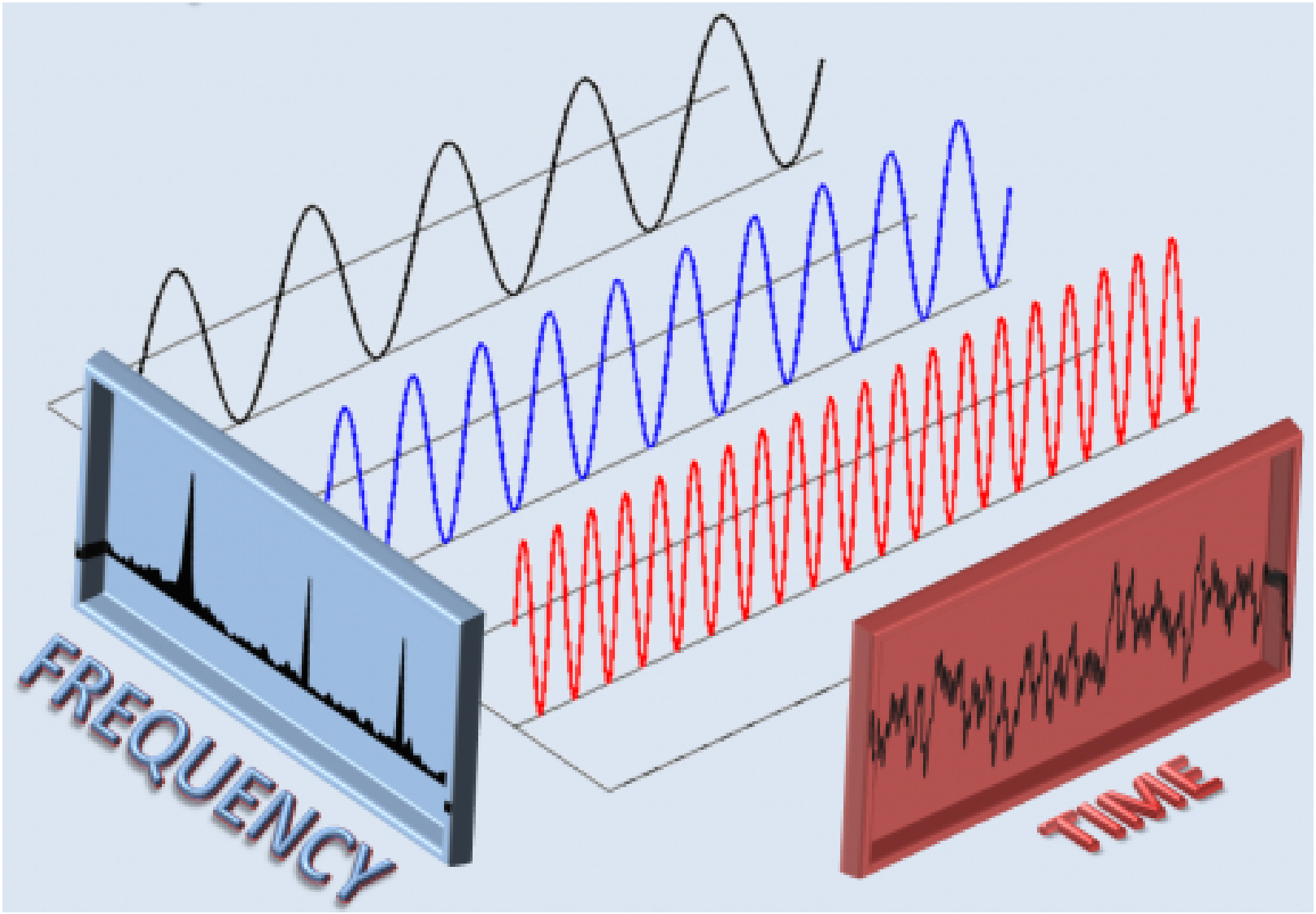
Assume that $\mathbf{y}$ is a data sample, for example the temprature at a certain weather station is recorded every hour for 100 years. ($\mathbf{y}_k$ is the temprature after k hours.) Here $N \approx 24 \cdot 365 \cdot 100 = 876000$.

A daily cycle (24 hours long period), and an annual cycle (8760 hours long period) will be easily observed.

If we calculate $\mathbf{f} = A^{-1}\mathbf{y}$, it will have one sharp peak at f_{100}, corresponding to the annual cycle (frequency 100), and a sharp peak at f_{36500} corresponding to the daily cycle (frequaency 36500).

Remark. In the literature DFT often means what we call the inverse DFT, and vice versa.

The intuitive meaning of the DFT

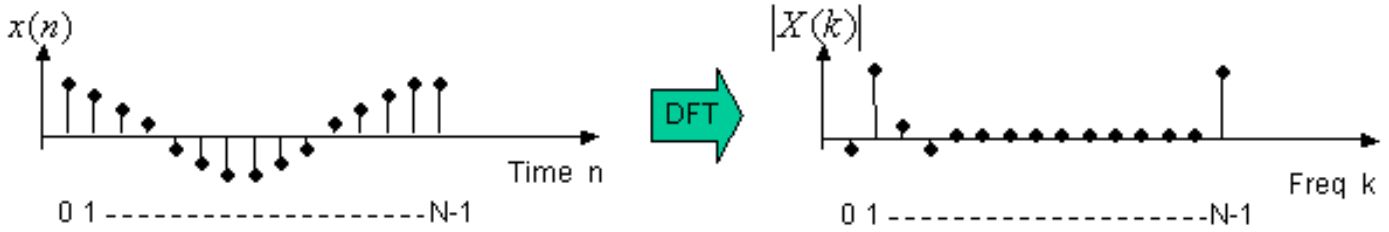


(Image source: http://felixonline.co.uk/science/2092/faster-fast-fourier-transform-found/)

Applications

The DFT and its inverse have many applications:

1. digital signal processing
2. image procressing
3. spectral analysis
4. data compression
5. partial differential equations
6. sound filtering
7. multiplication of large numbers



(Image source: `https://encrypted-tbn1.gstatic.com/images?q=tbn:ANd9GcSrm03Rwy5UG9Ypz5GKtdTLOG-p_N23YC3CQdCmtDuyjxRvpm4HmQ`)

Subsection 1

The Fast Fourier Transform

Computing DFT

To compute the DFT of a vector $\mathbf{f}$ is a matrix multiplicition, therefore it takes $O(N^2)$ time.

We can greatly reduce the running time, if N is a power of 2: $N = 2^m$. In this case the matrix A has a very special structure.

The Fast Fourier Transform (FFT) was invented by Cooley and Tukey in 1965. It has running time $O(mN) = O(N \log N)$.

FFT

Assume that $\mathbf{f}$ is an $N \times 1$ vector, $N = 2^m$, and we would like to calculate $\mathbf{y} = A\mathbf{f}$ the DFT of $\mathbf{f}$.

We introduce the following notation: the $k \times k$ matrix A_k denotes the matrix of the DFT as before, while D_k is a diagonal matrix of size $k \times k$ with diagonal elements $d_{jj} = \exp(2i\pi(j-1)/k)$. Furthermore, let $\mathbf{f}_{odd} = [f_1, f_3, \ldots, f_{N-1}]^T$ and $\mathbf{f}_{even} = [f_2, f_4, \ldots, f_N]^T$.

Theorem

With the notation above

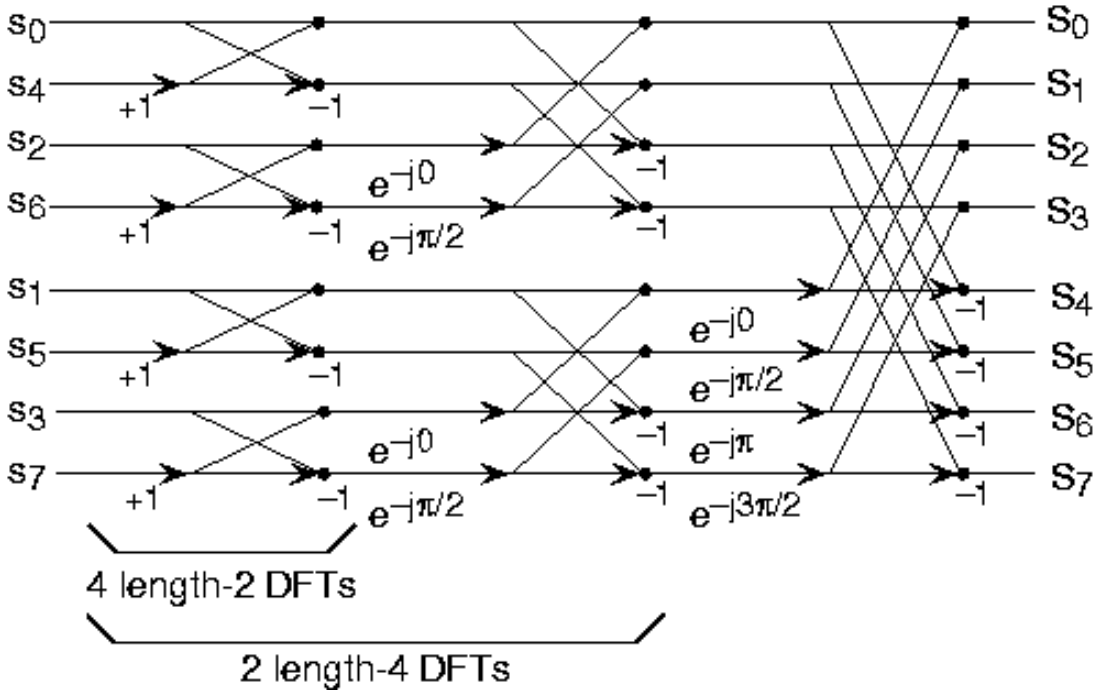
$$\mathbf{A}_N\mathbf{f} = \left[\begin{array}{c|c} A_{N/2} & D_{N/2}A_{N/2} \\ \hline A_{N/2} & -D_{N/2}A_{N/2} \end{array}\right] \cdot \left[\begin{array}{c} \mathbf{f}_{odd} \\ \hline \mathbf{f}_{even} \end{array}\right] = \left[\begin{array}{c} A_{N/2}\mathbf{f}_{odd} + D_{N/2}A_{N/2}\mathbf{f}_{even} \\ \hline A_{N/2}\mathbf{f}_{odd} - D_{N/2}A_{N/2}\mathbf{f}_{even} \end{array}\right].$$

The proof of the Theorem (once the result is known) is a straightforward calculation.

Divide and conquer

The Theorem shows us, that to calculate $A_N\mathbf{F}$, we basically need $A_{N/2}\mathbf{f}_{odd}$ and $A_{N/2}\mathbf{f}_{even}$. Once we know $A_{N/2}\mathbf{f}_{odd}$ and $A_{N/2}\mathbf{f}_{even}$, it is easy to see, that we can finish the calculation in just $O(N)$ steps.

Thus we diveded the DFT of size N into two DFTs of size $N/2$. In $O(\log N)$ steps we break down the problem into N obvious problem of size 1. The recursive algorithm we sketched has running time $O(N \log N)$.



Section 8

References

References

- Sewell G. *Computational Methods of Linear Algebra,* 2nd edition, Wiley, 2005.
- Demmel J. W. *Applied Numerical Linear Algebra*, SIAM, 1997.
- Galántai A. *Alkalmazott lineáris algebra (jegyzet)*, Miskolci Egyetemi Kiadó, 1999.
- Vanderbei R. J. *Linear Programming Foundations and Extensions,* Springer, 2007.
- Wikipedia, 2014., `www.wikipedia.com`
- `http://www8.cs.umu.se/kurser/5DV005/HT10/gauss.pdf`
- `http://people.inf.ethz.ch/arbenz/ewp/Lnotes/chapter3.pdf`