Appendix
Mathematical notions

This appendix is a review of notions the reader should already be familiar with: it is brief and to the point and mostly just provides formal definitions to be used in the book. We assume that the reader is acquainted with the concept of a *set*.[1]

A.1 Set cardinality

Sets can have finite or infinite cardinality; sets of the latter type can be countably or uncountably infinite. A set X is countably infinite if there is a bijection $X \leftrightarrow \mathbb{N}$. Among the sets of numbers, $\mathbb{N}, \mathbb{Z}, \mathbb{Q}$ are countably infinite. An infinite set is uncountably infinite if it is not countably infinite; $\mathbb{R}$ and $\mathbb{C}$ are examples of uncountably infinite sets. See [65, Chap. 1] for more information, and [67] for a formal treatment.

A.2 Some notation

We employ the following shorthand notations:

- $\forall$: for all, for each, for any;
- $\exists$: there is, there exists;
- $\exists!$: there is only one, there is a unique;
- $\wedge$: and;
- $\vee$: or (not the exclusive or—$A \vee B$ means either A or B or both);
- $\neg$: not;
- $\rightarrow, \Rightarrow$: implies;
- $\leftarrow, \Leftarrow$: is implied by;
- (): determine operation precedence.

If x, y are sets,

- $x \in y$: x is an element of y (sets can be elements of other sets);
- $x \cap y = \{z \mid z \in x \wedge z \in y\}$ is the intersection of x, y;
- $x \cup y = \{z \mid z \in x \vee z \in y\}$ is the union of x, y;
- $\bigcup x = \{z \mid \exists y \in x \ (z \in y)\}$ is the set of elements x;

[1] See [115] for a good technical introduction to set theory, and [67] for advanced notions.

L. Liberti and C. Lavor, *Euclidean Distance Geometry*, Springer Undergraduate Texts in Mathematics and Technology, DOI 10.1007/978-3-319-60792-4

- $x \setminus y = \{z \mid z \in x \wedge z \notin y\}$ is the set difference of x and y;
- $x \subseteq y$: x is a subset of y, i.e., $\forall z\ (z \in x \to z \in y)$;
- $x \supseteq y$: x is a superset of y, i.e.,$\forall z\ (z \in y \to z \in x)$;
- $x = y$: x, y are equal, i.e., $x \subseteq y \wedge y \subseteq x$;
- $x \neq y$: x, y are different, i.e., $\neg(x = y)$;
- $x \subsetneq y$: x is a strict subset of y, i.e., $x \subseteq y \wedge x \neq y$;
- $x \supsetneq y$: x is a strict superset of y, i.e., $x \supseteq y \wedge x \neq y$;
- whenever x is a set, $|x|$ is the cardinality of x
  (if x is a finite set, $|x|$ is the number of elements in x).

A.3 Fields

A *field* is a set $\mathbb{F}$ with two binary operations (sum and product) acting on its elements, according to the rules (or *axioms*) below. Fields usually contain numbers, and the two operations correspond to the ordinary sum and product.

1. Closure of sum
   $\forall \alpha, \beta \in \mathbb{F}\ (\alpha + \beta \in \mathbb{F})$;
2. closure of product
   $\forall \alpha, \beta \in \mathbb{F}\ (\alpha\beta \in \mathbb{F})$;
3. associativity of sum
   $\forall \alpha, \beta, \gamma \in \mathbb{F}\ (\alpha + (\beta + \gamma) = (\alpha + \beta) + \gamma)$;
4. associativity of product
   $\forall \alpha, \beta, \gamma \in \mathbb{F}\ (\alpha(\beta\gamma) = (\alpha\beta)\gamma)$;
5. commutativity of sum
   $\forall \alpha, \beta \in \mathbb{F}\ (\alpha + \beta = \beta + \alpha)$;
6. commutativity of product
   $\forall \alpha, \beta \in \mathbb{F}\ (\alpha\beta = \beta\alpha)$;
7. distributivity of product over sum
   $\forall \alpha, \beta, \gamma \in \mathbb{F}\ (\alpha(\beta + \gamma) = \alpha\beta + \alpha\gamma)$;
8. $\mathbb{F}$ contains two elements called 0 and 1 which satisfy Axioms 9–13 (and no other element of $\mathbb{F}$ satisfies them);
9. sum is invariant w.r.t. 0
   $\forall \alpha \in \mathbb{F}\ (0 + \alpha = \alpha)$;
10. product is zero w.r.t. 0
    $\forall \alpha \in \mathbb{F}\ (0\alpha = 0)$;
11. product is invariant w.r.t. 1
    $\forall \alpha \in \mathbb{F}\ (\alpha \neq 0 \to 1\alpha = 1)$;
12. every $\alpha \in \mathbb{F}$ has an inverse $-\alpha$ with respect to sum
    $\forall \alpha \in \mathbb{F}\ \exists!(-\alpha) \in \mathbb{F}\ (\alpha + (-\alpha) = 0)$;
13. (every nonzero $\alpha \in \mathbb{F}$ has an inverse α^{-1} with respect to product)
    $\forall \alpha \in \mathbb{F}\ (\alpha \neq 0 \to \exists!\alpha^{-1} \in \mathbb{F}\ (\alpha\alpha^{-1} = 1))$.

We write $\alpha - \beta$ to mean $\alpha + (-\beta)$, for any $\alpha, \beta \in \mathbb{F}$. From the field axioms, we can prove elementary statements such that $\alpha + 0 = \alpha$ for all $\alpha \in \mathbb{F}$: this follows from Axioms 9 and 5.

The rational numbers, denoted by $\mathbb{Q}$, are a field. The real numbers $\mathbb{R}$ are also a field, and so are the complex numbers $\mathbb{C}$. The integers $\mathbb{Z}$ are not a field and neither are the nonnegative integers $\mathbb{N}$.

A.4 Vector spaces

A *vector space* over a field $\mathbb{F}$ is a set $\mathcal{V}$ satisfying the following axioms:

1. associativity
   $\forall x, y, z \in \mathcal{V}\ (x + (y + z) = (x + y) + z)$;
2. commutativity
   $\forall x, y \in \mathcal{V}\ (x + y = y + x)$;
3. $\mathcal{V}$ contains a unique element called 0 such that $\forall x \in \mathcal{V}\ (0 + x = x)$;
4. every $x \in \mathcal{V}$ has an inverse $-x$ with respect to sum
   $\forall x \in \mathcal{V}\ \exists!(-x) \in \mathbb{F}\ (x + (-x) = 0)$;
5. closure of scalar[2] product
   $\forall \alpha \in \mathbb{F}, x \in \mathcal{V}\ (\alpha x \in V)$;
6. associativity of the product by a scalar
   $\forall \alpha, \beta \in \mathbb{F}, x \in \mathcal{V}\ (\alpha(\beta x) = (\alpha\beta)x)$;
7. distributivity of $\mathbb{F}$ w.r.t. V
   $\forall \alpha \in \mathbb{F},\ x, y \in \mathcal{V}\ (\alpha(x + y) = \alpha x + \alpha y)$;
8. distributivity of $\mathcal{V}$ w.r.t. $\mathbb{F}$
   $\forall \alpha, \beta \in \mathbb{F}, x \in \mathcal{V}\ ((\alpha + \beta)x = \alpha x + \beta x)$;
9. invariance of product w.r.t. 1
   $\forall x \in \mathcal{V}\ (1x = x)$.

We write $-x$ for $-1x$, and $x - y$ to mean $x + (-y)$, for any $x, y \in \mathcal{V}$.

From these axioms, we can formally prove elementary statements such as $0x = 0$ for all $x \in \mathcal{V}$:

$$
\begin{aligned}
&\forall 0 \neq \alpha \in \mathbb{F}\ 0x = (\alpha + (-\alpha))x \text{ by field Axiom (12)} \\
&= \alpha x + (-\alpha)x \text{ by (8)} \\
&= \alpha x + (-1\alpha)x) \text{ by field Axiom (11)} \\
&= \alpha x + (-1)(\alpha x) \text{ by (6)} \\
&= (\alpha x) + -(\alpha x) \text{ by (9)} \\
&= 0 \text{ by (4)}.
\end{aligned}
$$

A.4.1 Linear independence and bases

Let $\mathcal{V}$ be a vector space over $\mathbb{F}$, and consider vectors $x_1, \ldots, x_n \in \mathcal{V}$ such that, for any choice of n scalars $\lambda_1, \ldots, \lambda_n \in \mathbb{F}$, the following holds:

$$\lambda_1 x_1 + \cdots + \lambda_n x_n = 0 \quad \Leftrightarrow \quad \lambda_1 = \ldots = \lambda_n = 0. \tag{A.1}$$

Then, we say that the set $B = \{x_1, \ldots, x_n\}$ is *linearly independent*. Otherwise, B is *linearly dependent*, i.e., there are scalars $\lambda_1, \ldots, \lambda_n$ not all zeroes, such that

$$\sum_{i=1}^{n} \lambda_i x_i = 0. \tag{A.2}$$

[2] When a vector space is defined over a field, the field elements are often called *scalars*.

If B is linearly dependent, then there must exist an index $j \leq n$ such that $\lambda_j \neq 0$ in Eq. (A.2), so:

$$\lambda_j x_j + \sum_{\substack{i \leq n \\ i \neq j}} \lambda_i x_i = 0$$
$$\Rightarrow \quad \sum_{\substack{i \leq n \\ i \neq j}} -\frac{\lambda_i}{\lambda_j} x_i = x_j,$$

which means that x_j can be expressed as a *linear combination* of $x_1, \ldots, x_{j-1}, x_{j+1}, \ldots, x_n$. A linearly independent set in V having maximum cardinality is called a *basis*.

A.4.2 Dimension

It turns out that all bases of a given vector space have the same cardinality, which is called the *dimension* of $\mathcal{V}$ and denoted by $\mathsf{dim}\mathcal{V}$. With respect to a given basis $B = \{x_1, \ldots, x_n\}$, every vector $y \in \mathcal{V}$ is a linear combination of vectors in B: if it were not so, then $B \cup \{y\}$ would be linearly independent and have a larger cardinality than B, which contradicts the fact that B is a basis. Hence, we can express any vector $v \in \mathcal{V}$ as $v = \sum_{i \leq n} v_i x_i$ for some set of n scalars $v_1, \ldots, v_n$ in $\mathbb{F}$. This justifies the following representation:

$$v = (v_1, \ldots, v_n). \tag{A.3}$$

Now, for any $i \leq n$ (where $n = \mathsf{dim}\mathcal{V}$), define e_i to be the vector $(0, \ldots, 0, 1, 0, \ldots, 0)$ which has a 1 in the ith component, and 0 elsewhere. The set $B = \{e_1, \ldots, e_n\}$ is a basis, called the *standard basis* of $\mathcal{V}$.

If $\mathcal{V}$ has dimension 1, then the standard basis consists of a single vector $e_1 = (1)$, and any vector $v \in \mathcal{V}$ can be written as (v_1), where $v_1 \in \mathbb{F}$. This makes it obvious that there is a bijection between $\mathbb{F}$ and a one-dimensional vector space over $\mathbb{F}$. If we take $\mathbb{F} = \mathbb{R}$, then a 1D vector space over $\mathbb{R}$ is essentially the same as the real line $\mathbb{R}$. For two dimensions, the standard basis is $B = \{e_1, e_2\}$, and 2D vectors can be written as (v_1, v_2), where v_1, v_2 are scalars in $\mathbb{R}$. This is the same representation used for the Cartesian plane, where each vector is given by a horizontal and a vertical coordinate. For 3D or general K-dimensional vector spaces, we just append components: from $v = (v_1, v_2, v_3)$ to $v = (v_1, \ldots, v_K)$.

A.4.3 Subspaces

If $\mathcal{U}, \mathcal{V}$ are vector spaces and $\mathcal{U} \subseteq \mathcal{V}$, then $\mathcal{U}$ is a *subspace* of $\mathcal{V}$. The intersection of two subspaces is a subspace.

Any (infinite) line containing the origin $(0, 0)$ is a subspace of $\mathbb{R}^2$; any line or plane containing the origin $(0, 0, 0)$ is a subspace of $\mathbb{R}^3$. The dimension of a line is 1; the dimension of a plane is 2.

The *span* of a set of vectors S in a vector space $\mathcal{V}$ is the set of vectors T that consists of vectors of the form $\sum_{x \in S} \lambda_x x$, where λ_x are scalars. The span of S is a subspace of $\mathcal{V}$.

An *affine space* is a subset $S = \{x + b \mid x \in \mathcal{V}', b \in \mathcal{V}\}$ of a vector space $\mathcal{V}$, where $\mathcal{V}'$ is a subspace of $\mathcal{V}$. The *dimension* of an affine space is the dimension of the subspace $\mathcal{V}'$. A set of vectors in $\mathcal{V}$ is *affinely dependent* if their translated pairwise differences all belong to an affine subspace S of $\mathcal{V}$, and *affinely independent* if the pairwise differences span the whole of $\mathcal{V}$. If X is a finite set of vectors in

$\mathcal{V}$, $\mathsf{aff}\, X$ is the affine space in $\mathcal{V}$ of smallest dimension which contains all vectors in X, also called the *affine hull* of X.

Geometrically, a subspace can be visualized in 2D as a line passing through the origin, or in 3D as a line or plane passing through the origin. An affine space is a translation of a subspace.

A.5 Matrices

A *matrix* is a rectangular array of scalars of a field $\mathbb{F}$, e.g., $(1\ 2)$ is a 1×2 matrix in $\mathbb{Q}$, $\begin{pmatrix} 1.1 & 2 \\ -1.5 & \sqrt{2} \end{pmatrix}$ is a 2×2 *square* matrix over $\mathbb{C}$ (and also $\mathbb{R}$), and so on. An $m \times n$ matrix A having component a_{ij} in row $i \leq m$ and column $j \leq n$ is denoted as $A = (a_{ij})$.

A.5.1 Transpose

The *transpose* of an $m \times n$ matrix $A = (a_{ij})$ is the $n \times m$ matrix $A^\top = (a_{ji})$. For example, the transpose of $\begin{pmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \end{pmatrix}$ is $\begin{pmatrix} 1 & 4 \\ 2 & 5 \\ 3 & 6 \end{pmatrix}$. An $m \times 1$ matrix is a *column vector*, and a $1 \times n$ matrix is a *row vector*.

A.5.2 Sum of matrices

A sum is defined on $m \times n$ matrices, namely:

$$\begin{pmatrix} a_{11} & \dots & a_{1n} \\ \vdots & \ddots & \vdots \\ a_{m1} & \dots & a_{mn} \end{pmatrix} + \begin{pmatrix} b_{11} & \dots & b_{1n} \\ \vdots & \ddots & \vdots \\ b_{m1} & \dots & b_{mn} \end{pmatrix} = \begin{pmatrix} a_{11} + b_{11} & \dots & a_{1n} + b_{1n} \\ \vdots & \ddots & \vdots \\ a_{m1} + b_{m1} & \dots & a_{mn} + b_{mn} \end{pmatrix}.$$

The invariant element of the matrix sum is the *zero* matrix (a matrix consisting of all zeroes).

A.5.3 Product by a scalar

A product by scalars is defined on $m \times n$ matrices, namely:

$$\lambda \begin{pmatrix} a_{11} & \dots & a_{1n} \\ \vdots & \ddots & \vdots \\ a_{m1} & \dots & a_{mn} \end{pmatrix} = \begin{pmatrix} \lambda a_{11} & \dots & \lambda a_{1n} \\ \vdots & \ddots & \vdots \\ \lambda a_{m1} & \dots & \lambda a_{mn} \end{pmatrix}.$$

A.5.4 Inner product

A product is defined between $m \times n$ and $n \times p$ matrices, namely:

$$\begin{pmatrix} a_{11} & \dots & a_{1n} \\ \vdots & \ddots & \vdots \\ a_{m1} & \dots & a_{mn} \end{pmatrix} \cdot \begin{pmatrix} b_{11} & \dots & b_{1p} \\ \vdots & \ddots & \vdots \\ b_{n1} & \dots & b_{np} \end{pmatrix} = \begin{pmatrix} \sum_{i \le n} a_{1i} b_{i1} & \dots & \sum_{i \le n} a_{1i} b_{ip} \\ \vdots & \ddots & \vdots \\ \sum_{i \le n} a_{mi} b_{i1} & \dots & \sum_{i \le n} a_{mi} b_{ip} \end{pmatrix},$$

or, in other words, the $m \times n$ matrix $A = (a_{ij})$ can be multiplied by an $n \times p$ matrix $B = (b_{ij})$ to obtain the $m \times p$ matrix $C = (c_{ij})$ where $c_{ij} = a_{i\cdot} \cdot b_{\cdot j}$, where $a_{i\cdot} = (a_{i1}, \dots, a_{in})$ and $b_{\cdot j} = (b_{1j}, \dots, b_{nj})$.

The invariant element of the matrix product is the *identity* matrix I: every diagonal element of I is 1, and every off-diagonal element is 0.

If $x, y \in \mathbb{R}^n$ are both column vectors, two possible products are defined: the inner product $x^\top \cdot y$ and the outer product $x \cdot y^\top$. The first yields the 1×1 matrix $(\sum_{i \le n} x_i y_i)$, which can be simply interpreted as the scalar $\sum_{i \le n} x_i y_i$, and the second yields the square $n \times n$ matrix having $x_i y_j$ as its (i, j)th component. By notational convention, $x \cdot y$ means $x^\top \cdot y$, whereas we explicitly write $x \cdot y^\top$ for the second case. If there is no ambiguity, we sometimes omit the $\cdot$ product symbol altogether.

The inner product $x \cdot y$ is often called *scalar product* of two vectors.

A.5.5 Linear transformations

Since a (column) vector $x \in \mathbb{R}^n$ is the same as an $n \times 1$ matrix, for a given $m \times n$ matrix A, we restrict the matrix product to multiply matrices and vectors: Ax is an $m \times 1$ column vector. We can therefore interpret matrices as mappings from $\mathbb{R}^n$ to $\mathbb{R}^m$. Square $n \times n$ matrices are mappings from $\mathbb{R}^n$ to itself.

This product is linear: $(\lambda A)x = \lambda(Ax)$ and $(A + B)x = Ax + Bx$ for any appropriately sized matrices A, B, vector x, scalar λ. Matrices interpreted as mappings defined on vector spaces are also known as *linear transformations*.

Consider a linear transformation $\mathbb{R}^n \to \mathbb{R}^m$ represented by a $m \times n$ matrix A. The *kernel* of A is $\mathsf{ker}\, A = \{x \in \mathbb{R}^n \mid Ax = 0\}$, and the *image* of A is $\mathsf{Im}\, A = \{y \in \mathbb{R}^m \mid \exists x \in \mathbb{R}^n \ (Ax = y)\}$. The kernel is a subspace of $\mathbb{R}^n$ and the image is a subspace of $\mathbb{R}^m$. Moreover, $\mathsf{dim\,Ker}\, A + \mathsf{dim\,Im}\, A = n$ (this is known as the *kernel and image* or *rank and nullity* theorem). We also call $\mathsf{dim\,Ker}\, A$ the *nullity* and $\mathsf{dim\,Im}\, A$ the *rank* of A and denote it $\mathsf{rk}\, A$. A has *full rank*, if $\mathsf{rk}\, A = \min(m, n)$. A square matrix with full rank is *nonsingular*; otherwise, it is *singular*. If A is square and nonsingular, then there exists another square $n \times n$ matrix A^{-1}, called the *inverse* of A, such that $A^{-1}A = AA^{-1} = I$.

A.5.6 Congruences

The transformation that adds a fixed vector to each vector of a set is called a *translation*. The transformation that rotates all the vectors in S by a given angle around a given center is called a *rotation* of the vector set. The transformation that reflects all the vectors in S with respect to a given hyperplane is called a *reflection* of the vector set.

Translations, rotations, and reflections of S are congruences of S, i.e., they preserve all distances (see Sect. A.6) between pairs of vectors in S. Less trivially, every congruence of S turns out to be a composition of translations, rotations, and reflections.

A.5.6.1 Translations

Algebraically, a translation τ acts on $S \subseteq \mathbb{R}^n$ as follows:

$$\forall x \in S \quad \tau(x) = x + b, \tag{A.4}$$

where $b \in \mathbb{R}^n$. This is also written in the more compact form $\tau(S) = S + b$.

A.5.6.2 Rotations

A rotation ρ_θ by an angle θ with respect to the origin acts on S as follows:

$$\forall x \in S \quad \rho_\theta(x) = A^\theta x, \tag{A.5}$$

where A^θ is an $n \times n$ matrix resulting from a product of scaled Givens matrices, defined below. With respect to a standard basis $e_1, \ldots, e_n$, where e_j is the zero vector with a 1 in the jth component, a *Givens matrix* $G^{\eta ij}$ rotates the unit projection of the unit vector $\hat{x} = \frac{x}{\sqrt{x \cdot x}}$ on the plane spanned by e_i, e_j by an angle η with respect to the origin. It embeds the 2D rotation matrix $r = \begin{pmatrix} \cos\eta & -\sin\eta \\ \sin\eta & \cos\eta \end{pmatrix}$ in an $n \times n$ identity matrix: $G^{\eta ij}_{ii} = r_{11}, G^{\eta ij}_{jj} = r_{22}, G^{\eta ij}_{ij} = r_{12}, G^{\eta ij}_{ji} = r_{21}$, and the rest of the components of $G^{\eta ij}$ equal to the components of the identity matrix:

$$G^{\eta ij} = \begin{pmatrix} & & \text{i} & & \text{j} & \\ 1 & \ldots & 0 & \ldots & 0 & \ldots 0 \\ \vdots & \ddots & \vdots & & \vdots & \vdots \\ 0 & \ldots & \cos\eta & \ldots & -\sin\eta & \ldots 0 \\ \vdots & & \vdots & \ddots & \vdots & \vdots \\ 0 & \ldots & \sin\eta & \ldots & \cos\eta & \ldots 0 \\ \vdots & & \vdots & & \vdots & \ddots \vdots \\ 0 & \ldots & 0 & \ldots & 0 & \ldots 1 \end{pmatrix} \begin{matrix} \\ \\ \text{i} \\ \\ \text{j} \\ \\ \end{matrix} \tag{A.6}$$

Thus, A^θ is a product of appropriately scaled Givens matrices $G^{\eta ij}$ for all $(i, j) \in H = \{(1, 1), \ldots, (1, n)\}$ (the reason why we fix i to 1 is that every swap (i, j) can be generated compositionwise as a product of the 2-cycle permutation swaps in H). In order to compute η from θ for each $(i, j) \in H$, one must have θ in terms of the angle between two vectors $x, y \in \mathbb{R}^n$, project x, y in the plane defined by each coordinate pair $(i, j) \in H$, then compute the angles η between the projections $\bar{x}, \bar{y}$, which must be scaled by $\|\bar{x}\|_2$ and $\|\bar{y}\|_2$, respectively (we are using the notation $\|\cdot\|_2$ before formally introducing it in Sect. A.6, but $\|z\|_2$ can be written as $z \cdot z$ for any vector z): this makes the projections have unit length, so that the Givens rotations apply. Accordingly, the product of the $G^{\eta ij}$ must be scaled back so that, when applied to x, it yields a vector with the same length as x.

Since this procedure only holds for rotations with respect to the origin, arbitrary rotations with respect to a vector $b \in \mathbb{R}^n$ must be prefixed by a translation to the origin and postfixed by its inverse, yielding $\rho_\theta(x) = A^\theta(x - b) + b$. Notationally, this is applied to each vector in S as $\rho_\theta(S) = A^\theta(S - b) + b$.

A.5.6.3 Reflections

A reflection $R_{a,0}$ acting on S, with respect to a hyperplane $a^\top x = 0$ (i.e., containing the origin and orthogonal to the unit vector $a \in \mathbb{R}^n$), is as follows:

$$\forall x \in S \quad R_{a,0}(x) = (I_n - 2aa^\top)x,$$

where $I_n - 2aa^\top$ is the matrix

$$\begin{pmatrix} 1-2a_1^2 & -2a_1a_2 & \dots & -2a_1a_n \\ -2a_2a_1 & 1-2a_2^2 & \dots & -2a_2a_n \\ \vdots & \vdots & \ddots & \vdots \\ -2a_na_1 & -2a_na_2 & \dots & 1-a_n^2 \end{pmatrix}.$$

For a reflection with respect to an affine subspace $a^\top x = b$, let $j \leq n$ be such that $a_j \neq 0$, and notice that the vector $\bar{b} = (0, \dots, \bar{b}_j, \dots, 0)$, having $\bar{b}_j = \frac{b}{a_j}$ as the only nonzero component, satisfies $a^\top \bar{b} = b$. So we simply translate x by $\bar{b}$ before reflecting it along $a^\top x = 0$, and by $-\bar{b}$ after. The resulting generalized reflection operator is:

$$\forall x \in S \quad R_{a,b}(x) = (I_n - 2aa^\top)(x - \bar{b}) + \bar{b}.$$

As before, we shorten the above as: $R_{a,b}(S) = (I_n - 2aa^\top)(S - \bar{b}) + \bar{b}$.

A.5.7 Determinants

Let A be an $n \times n$ square matrix $A = (a_{ij})$. For any given $i, j \leq n$, the $(n-1) \times (n-1)$ matrix A^{ij} given by deleting the ith row and jth column of A is the *submatrix* of A with respect to i, j.

For any $i, j \leq n$, consider the following recursive definitions of the functions ϕ_i, ψ_j mapping A to a scalar in $\mathbb{R}$:

$$\phi_i(A) = \sum_{j \leq n} (-1)^{i+j} a_{ij} \phi_i(A^{ij})$$
$$\psi_j(A) = \sum_{i \leq n} (-1)^{i+j} a_{ij} \psi_j(A^{ij}).$$

Notice that each ϕ_i applied to an $n \times n$ matrix is defined in terms of ϕ_i applied to an $(n-1) \times (n-1)$ matrix. The recursion starts by setting ϕ_i and ψ_j applied to 1×1 matrices (a_{11}) to be equal to their only component a_{11}.

It turns out that $\phi_i(A) = \psi_j(A)$ for all $i, j \leq n$, so we call this value the *determinant* of A and denote it by $|A|$; then, given any row index i or column index j, $|A| = \sum_{j \leq n} (-1)^{i+j} a_{ij} |A^{ij}| = \sum_{i \leq n} (-1)^{i+j} a_{ij} |A^{ij}|$.

If A is a square nonsingular $n \times n$ matrix, its inverse is given by:

$$A^{-1} = \frac{\mathsf{Adj}A}{|A|}$$

where $\mathsf{Adj}A$ is the *adjugate* matrix of A, having $(-1)^{i+j}|A^{ij}|$ as its (i, j)th component. We also have that $|A^{-1}| = \frac{1}{|A|}$ and that $|A \cdot B| = |A||B|$ for any square $n \times n$ matrix B.

A.5.8 Eigenvalues and eigenvectors

For a square $n \times n$ matrix B, consider the equation $By = \lambda y$, where λ is a scalar and $y \in \mathbb{C}^n$ is a nonzero vector: λ is an *eigenvalue* of B and y is an *eigenvector* of B associated to λ. By elementary linear algebra, B has at most $K = \mathsf{rk}\, B$ distinct eigenvalues. If B is a *symmetric matrix* (i.e., $B^\top = B$), all its eigenvalues and eigenvectors are real, and it is possible to choose K orthogonal eigenvectors of B which span the same K-dimensional subspace of $\mathbb{R}^n$ as the columns of B. Moreover, if one of the eigenvalues is zero, the corresponding eigenvectors span $\mathsf{Ker}\, B$, whereas the eigenvectors corresponding to nonzero eigenvalues span $\mathsf{Im}\, B$.

In general, if Λ is the $K \times K$ *diagonal matrix* having nonzero eigenvalues $\lambda_1, \ldots, \lambda_K$ (some of which may be equal) along the diagonal, and zeroes everywhere else, and if Y is the $n \times K$ matrix of corresponding eigenvectors of B, then $B = Y \cdot \Lambda \cdot Y^\top$.

A.6 Norms and metrics

Let $\mathcal{V}$ be a K-dimensional vector space over $\mathbb{R}$. We introduce a function of one argument, called *norm*, which maps each vector to a nonnegative real number. A function $\|\cdot\| : \mathcal{V} \to \mathbb{R}$ is a norm if it satisfies the following axioms:

1. $\forall x \in \mathcal{V}\ (\|x\| \geq 0)$;
2. $\forall x \in \mathcal{V}\ (\|x\| = 0 \leftrightarrow x = 0)$;
3. $\forall \lambda \in \mathbb{R}, x \in \mathcal{V}\ (\|\lambda x\| = \lambda \|x\|)$;
4. $\forall x, y \in \mathcal{V}\ (\|x + y\| \leq \|x\| + \|y\|)$.

A typical example of a norm is the *Euclidean norm*, defined as follows:

$$\forall x = (x_1, \ldots, x_n) \in \mathcal{V} \quad \|x\|_2 = \sqrt{x \cdot x} = \sqrt{\sum_{i \leq n} x_i^2}. \tag{A.7}$$

Now, we introduce a function of two arguments, called *metric*, which maps ordered pairs of vectors to a nonnegative real number. A function $d(\cdot, \cdot)$ is a metric if it satisfies the following axioms:

1. $\forall x, y \in \mathcal{V}\ (d(x, y) = 0 \leftrightarrow x = y)$;
2. $\forall x, y \in \mathcal{V}\ (d(x, y) = d(y, x))$;
3. $\forall x, y, z \in \mathcal{V}\ (d(x, z) \leq d(x, y) + d(y, z))$;
4. $\forall x, y \in \mathcal{V}\ (d(x, y) \geq 0)$.

A typical example of a metric is the *Euclidean metric*, defined in function of the Euclidean norm as:

$$\forall x, y \in \mathcal{V}\ (d(x, y) = \|x - y\|_2). \tag{A.8}$$

A vector space endowed with a metric is called a *metric space*.

Given a point x of a metric space $\mathcal{V}$, a neighborhood χ of x contains all elements y of $\mathcal{V}$ such that $d(x, y) \leq \epsilon$ for some (usually small) given constant $\epsilon > 0$. This concept extends to finite metric spaces (intended as finite sets, not necessarily vector spaces, endowed with a metric), see Sect. A.8.1.

A.6.1 The intuition behind vectorial geometry

A *Euclidean space* is a vector space $\mathcal{V}$ over $\mathbb{R}$ endowed with the Euclidean norm and distance. If $\mathsf{dim}\mathcal{V} = K$, we denote the Euclidean space by $\mathbb{R}^K$. $\mathbb{R}^1$ is a good mathematical model for a straight line, $\mathbb{R}^2$ for a plane, and $\mathbb{R}^3$ for the physical space that surrounds us locally. Vectors correspond to points in this space, with coordinates indicating width, depth and height. The norm of a vector x is its length (or the Euclidean distance from the point x to the origin), and the Euclidean metric between x and y is the Euclidean distance between the point x and the point y.

A.7 Groups

A *group* is a set G with a product operation that satisfies the following axioms:

1. closure
   $\forall g, h \in G\ (gh \in G)$;
2. associativity
   $\forall f, g, h \in G\ ((fg)h = f(gh))$;
3. G contains a unique identity element e
   $\exists! e \in G\ \forall g \in G\ (eg = ge = g)$;
4. every element has a unique inverse
   $\forall g \in G\ \exists! g^{-1} \in G\ (gg^{-1} = g^{-1}g = e)$.

From these axioms, we can easily prove elementary statements. For example, the uniqueness of the inverse actually follows from other axioms:

$$\begin{aligned} & \forall g \in G \exists h, h' \in G\ (gh = hg = gh' = h'g = e) \\ \Rightarrow\ & gh' = e \\ \Rightarrow\ & hgh' = he \\ \Rightarrow\ & eh' = he \\ \Rightarrow\ & h' = h. \end{aligned}$$

Note that the field axioms define a group over the sum and over the product—hence, the uniqueness of the inverses can also be reduced to other axioms in the same way.

A.7.1 A finite planar rotation group

For example, the set $R = \{0, \pi/2, \pi, 3\pi/2\}$ of planar rotations around the origin, under the operation given by $+$, is a group: all sums of elements of R are in R (because $2\pi = 0$); the sum is associative by definition, 0 is the unique identity, and given any rotation $g \in R$, there is always another rotation $g^{-1} \in R$ (denote it by $-g$) with $g + (-g) = 0$.

A.7.2 Abelian groups

A group for which commutativity holds, i.e., $\forall g, h \in G\ (gh = hg)$ is called *Abelian*. $(R, +)$ above is Abelian.

A.7.3 The group table

The *table* of a finite group G is a square $|G| \times |G|$ matrix whose (g, h)th entry is the product gh. For example, the table of $(R, +)$ above is:

	0	$\pi/2$	π	$3\pi/2$
0	0	$\pi/2$	π	$3\pi/2$
$\pi/2$	$\pi/2$	π	$3\pi/2$	0
π	π	$3\pi/2$	0	$\pi/2$
$3\pi/2$	$3\pi/2$	0	$\pi/2$	π

Let G be a group and H a subset of G. If H is itself a group under the same product as G, then H is a *subgroup* of G, denoted $H \leq G$. If $H \leq G$, then $|H|$ divides $|G|$. For example, if $S = \{0, \pi\}$, then $(S, +) \leq (R, +)$. Notice that 2 divides 4.

An element $g \in G$ such that $gg = g^2 = e$ is called *idempotent*. For example, since $(\pi)^2 = \pi + \pi = 0$, π is idempotent.

A.7.4 Group actions

Let V be any set, and G be a group of functions on V, where the product of two functions f, g is defined by their composition $f \circ g$. Then, G defines an *action* on V; we say that G *acts on* V.

For example, $(R, +)$ defines an action on $\mathbb{R}^2$: for any vector $x \in \mathbb{R}^2$ and any rotation $g \in G$, gx is the vector in $\mathbb{R}^2$ corresponding to x being rotated by g.

A.7.5 Generators

Let G be a group and $S \subseteq G$ a subset of its elements. If we can write any $g \in G$ as a product of some elements in S, then G is *generated* by S, which is a set of *generators* of G (written $G = \langle S \rangle$). We are often interested in finding *minimal* generator sets, i.e., sets of generators of minimum cardinality.

For example, $(R, +) = \langle \pi/2 \rangle$ and $(S, +) = \langle \pi \rangle$.

A.7.6 Orbits

Let G be a group acting on V. Define an equivalence relation $\sim$ on V given by $u \sim v$ if $\exists g \in G$ with $v = gu$. Then, $\sim$ partitions[3] V into equivalence classes called *orbits*.

[3] A *partition* of a set V is a set of subsets of V such that their union is V and their pairwise intersection is empty.

Another definition for orbits is as follows. For any $v \in V$, let $Gv = \{gv \mid g \in G\} \subseteq V$. So Gv is the orbit of v with respect to G. The two definitions are equivalent: if $u, w \in Gv$ for some $v \in V$, obviously $u \sim w$ by definition; and if u, w are part of an orbit, then there must exist $v \in V$ such that $u = gv$ and $w = hv$ for some $g, h \in G$: namely, take $w = v$ and $h = e$.

Let Ω be a partition of V into orbits $\omega_1, \ldots, \omega_\ell$. What this means, in practice, is that there is no $g \in G$ that can map an element in a given orbit to an element in a different orbit. If Ω contains only one orbit, i.e., $\ell = 1$ and $\Omega = \{\omega_1\}$, then obviously $\omega_1 = V$, and the action of G on V is said to be *transitive* (sometimes, we abuse terminology and simply say that G is transitive).

By the second definition of orbit, for each orbit $\omega \in \Omega$ we can single out an element $v \in \omega$ and call it an *orbit representative*.

Orbits come in useful when V is expensive to compute, but G is not: then, we might want to compute a small subset $S \subset V$ and then try to generate all the elements of V as products gv, where $g \in G$ and $v \in S$. In general, S must be a set of orbit representatives. Then, $\bigcup_{v \in S} Gv = V$. The computationally most convenient setting is when G is transitive, as S can be reduced to a single orbit representative.

The group $(R, +)$ defined above naturally acts on $\mathbb{R}^2$ by rotating vectors in the plane. The orbit of any vector y is a set ω_y, which contains the four vectors $\{y, A^{\pi/2}y, A^{\pi}y, A^{3\pi/2}y\}$, where A^θ is the rotation matrix of the angle θ in the plane. If $y = (\sqrt{2}/2, \sqrt{2}/2)$, ω_y consists of the four corners of the unit square centered at the origin. Since rotations do not change relative lengths, $(R, +)$ also naturally acts on the unit circle $S^1 = ((0, 0), 1)$ centered at the origin, and having unit radius. $(R, +)$ partitions the unit circles into a set Ω of uncountably many orbits: any $\omega \in \Omega$ consist of four points on the unit circle, and $\bigcup_{\omega \in \Omega} \omega$ *is* the unit circle.

A.7.7 Isomorphism

A *group homomorphism* from G to H is a mapping $\phi: G \to H$ such that, for any $f, g \in G$, $\phi(fg) = \phi(f)\phi(g)$. A *group isomorphism* between G and H is a bijection $\phi: G \to H$ such that ϕ, ϕ^{-1} are both homomorphisms. If ϕ is a group isomorphism between G and H, then we write $G \cong H$.

In practice, two groups are isomorphic if their group table looks the same, aside from the names of the elements. We tend to think of isomorphic groups as "essentially the same group."

For example, the *cyclic group* $C_2 = \{e, g\}$ is isomorphic to the group $(S, +)$ above (with $S = \{0, \pi\}$). Their group tables are:

	e	g
e	e	g
g	g	e

	0	π
0	0	π
π	π	0

It is obvious that it suffices to change the name g into π to recover the same group table. C_2 is the simplest nontrivial finite group; it has a single nonidentity element which is idempotent. Since $(S, +) \cong C_2$, $(S, +)$ and C_2 are essentially the same group.

A.8 Graphs

Graphs are a model of sets of pairs of elements in a set. Let V be any finite set, and let E a set of (unordered) pairs of elements in V. Thus, for example, we might take $V = \{A, B, C, S\}$ and $E = \{\{A, B\}, \{A, C\}, \{A, S\}, \{B, C\}\}$. Then, the couple $G = (V, E)$ is called a *graph*. If E consists of ordered pairs, then the graph is *directed*; otherwise, it is *undirected*. A directed graph is also sometimes called a *digraph*. Usually, if G is undirected, elements of V are called *vertices* and elements of E are

called *edges*. For *digraph*, the corresponding terminology is usually *nodes* and *arcs*. If G is a given graph, we also refer to its vertex set as $V(G)$ and to its edge set as $E(G)$. Two vertices u, v are *adjacent* if $\{u, v\} \in E$. An edge $e = \{u, v\}$ is *incident* to u and v, and vice versa u and v are *incident* to e. Two edges e, f are adjacent if $|e \cap f| = 1$, i.e., if they are both incident to the same vertex. Given a set of edges E, we let $V[E]$ be the set of vertices incident to (or induced by) all edges in E.

A.8.1 Neighborhoods, degrees and cutsets

For $G = (V, E)$ an undirected graph and $v \in V$, let $N_G(v) = \{u \in V \mid \{u, v\} \in E\}$ be the *neighborhood* of v, and $\delta_G(v) = \{\{u, v\} \in E\}$ be the *star* of v. Let $|N_G(v)|$ be the *degree* of v. Given $U \subseteq V$, we let $N_G(U) = \{v \in V \mid \exists u \in U \ \{u, v\} \in E\}$ be the *cut* or *neighborhood* of U, i.e., the set of vertices of $V \setminus U$ adjacent to a vertex in U (also see Sect. A.6). We let $\delta_G(U) = \{e \in E \mid |e \cap U| = 1\}$, the *cutset* of U, be the set of edges of E incident to exactly one vertex in U. We write $N(\cdot)$, $\delta(\cdot)$ whenever there is no ambiguity.

A.8.2 Simplicity and connectedness

A graph is *nonsimple* if some vertex is adjacent to itself (i.e., E contains some singleton sets $\{v\}$), or if E is a multiset, i.e., a list, or sequence, of some possibly repeated values. Singleton sets in E are called *loops*, and repeated occurrences of the same edge in E are called *parallel edges*. A graph is *simple* if it has no loops nor parallel edges. A graph $G = (V, E)$ is *connected* if no nontrivial cutset is empty (see Fig. A.1):

$$\forall U \subseteq V \quad (U \notin \{\varnothing, V\} \rightarrow \delta(U) \neq \varnothing). \tag{A.9}$$

A.8.3 Subgraphs

For $U \subseteq V$, we let $E[U] = \{\{u, v\} \in E \mid u, v, \in U\}$ be the set of edges *induced* by U, and let $G[U] = (U, E[U])$ be the *induced subgraph* of G w.r.t. U. Any graph (U, F) where $F \subseteq E[U]$ is a *subgraph* of G (Fig. A.2).

A subgraph of G is *spanning* in G if its vertex set is $V(G)$.

Fig. A.1 *Left* the graph is not connected, as the cutset defined by $\{1, 2, 3\}$ is nontrivial and empty. *Right* a connected graph.

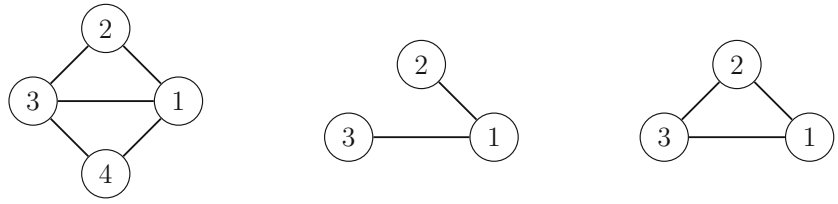

Fig. A.2 A graph, a (noninduced) subgraph and an induced subgraph.

A.8.4 Simple cycles and paths

Given a set of edges F, it induces the set $\bigcup F$ of vertices, and a graph $(U, \bigcup F)$. A *simple cycle* is a connected graph where each vertex has degree 2. If C is a simple cycle and the edge set F is such that $E(C) \setminus F$ induces a connected graph P with $E(P) \subseteq E(C)$, then P is called a *simple path* (see Fig. A.3):

$$\forall U \subseteq V \quad (U \notin \{\varnothing, V\} \to \delta(U) \neq \varnothing). \tag{A.10}$$

A simple cycle or path in a graph G is *Hamiltonian* if it is a spanning subgraph of G.

A.8.5 Edge weights

An *edge weight function* is a mapping $d : E \to \mathcal{S}$ for some set $\mathcal{S}$ of numbers. In the following, we often take $\mathcal{S} = \mathbb{R}_+$. If an edge weight function is defined on a graph, then the graph is *weighted*; otherwise, it is *unweighted*.

A.8.6 Some graph families

A graph is *complete* if it has all possible edges/arcs. A simple undirected complete graph on K vertices is called a K*-clique* (see Fig. A.4, left).

A graph $G = (V, E)$ is *bipartite* if its vertex set V can be partitioned into two subsets U, W such that for each edge $\{u, w\} \in E$, we have $u \in U$ and $w \in W$. A complete bipartite graph is also called a *biclique*.

The *complement* of a graph $G = (V, E)$ is a graph $\bar{G}$ on V where $\{u, v\}$ is an edge if and only if $\{u, v\} \notin E$. The *empty* graph is the complement of a clique. An induced subgraph S of G is *stable* (or an *independent set*) if $E[S]$ is empty (see Fig. A.4, right).

Fig. A.3 *Left* a simple cycle. *Right* a simple path yielded by $F = \{\{1, 6\}\}$.

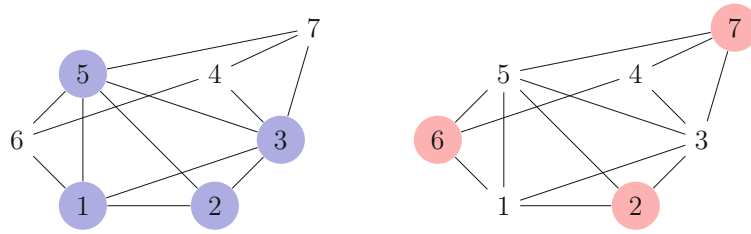

Fig. A.4 A 4-clique (*left*) and a stable set (*right*) in G.

A.9 Computational complexity

We discuss the very basic notions of computational complexity [104].

A.9.1 Worst-case complexity of an algorithm

The efficiency of algorithms is evaluated according to several criteria. The most common ones are the number of *elementary operations* (i.e., taking unit time to complete) executed before termination, or the amount of memory used during execution.

We focus on the former, which is an approximation for the execution time in practice. The *worst-case time complexity* $O(f(n))$ of an algorithm is the maximum number $f(n)$ of elementary operations executed by the algorithm over all the possible inputs encoded into n bits. This "big-oh" notation $O(f(n))$ follows asymptotic rules: if k is a constant, then $O(f(n)+k) = O(kf(n)) = O(f(n))$. If $\exists h, k, N \in \mathbb{N}\ \forall n > N\ g(n) \leq h + kf(n)$ (i.e., g is asymptotically smaller than f), then $O(g(n) + f(n)) = O(f(n))$.

There is a further convention on the big-oh notation: since these are worst-case bounds, rather than exact complexity counts, an algorithm which takes $O(n)$ at worst might also be said to take $O(n^2)$ at worst, or $O(f(n))$ for any f that is asymptotically larger than n. Formally, if g is asymptotically smaller than f, then $O(g(n)) = O(f(n))$. Of course, the tighter the function f providing the bound, the better the estimate will be.

A.9.1 Example

Let us find the worst-case time complexity of Alg. 13. We assume that $a_1, \ldots, a_n$ are integers having size bounded by 32 bits, so the size of the input is $32n$, which, asymptotically, behaves like n. The two statements inside the j loop are composed of 4 elementary operations: two sums and two assignments. These 4 operations are executed n times, for a total of $4n$ operations, and each j loop is executed n times, for a total of $4n^2$ operations. Each iteration of the k loop also requires 4 elementary operations: computing k mod 2, verifying whether it is zero, performing a difference $(c - k)$ and an assignment. These are executed $\lceil \frac{n}{2} \rceil$ times, for a total of $4\lceil \frac{n}{2} \rceil$ operations. Thus, the number of elementary operations over the whole algorithm is

$$f(n) = 4(n^2 + \left\lceil \frac{n}{2} \right\rceil).$$

Now, the asymptotical reduction is as follows: $O(\lceil \frac{n}{2} \rceil) = O(n)$ (because $\lceil \frac{n}{2} \rceil \leq n$ for each $n \in \mathbb{N}$), and $O(n^2 + n) = O(n^2)$ (because $n \leq n^2$ for each $n \in \mathbb{N}$). Also, $O(4n^2) = O(n^2)$ because 4 is a constant which does not depend on n. So the worst-case complexity of this algorithm is $O(n^2)$.

Algorithm 13 An $O(n^2)$ algorithm.

```
input (a_1, ..., a_n)
for i ≤ n do
  for j ≤ n do
    b = a_i + a_j
    c = c + b
  end for
end for
for k ≤ n such that k is odd do
  c = c − k
end for
return c
```

A.9.2 Decision and optimization problems

In computational complexity theory, by "problem" we usually mean a *decision problem*, where the output is just a YES or a NO. For example, "given a graph G and a positive integer K, does G contain a K-clique?" is a decision problem.

An *optimization problem* also provides a function from the set of possible inputs to some number field and asks for the input that maximizes or minimizes the function (which is called *objective function*). For example, "given a graph G, find the maximum K such that G contains a K-clique" is an *optimization problem*.

Every optimization problem has an associated decision problem. If $\phi(\cdot)$ is the objective function to be minimized, then the decision problem provides a lower bound K and asks whether there exists an input ι such that $\phi(\iota) \leq K$; if $\phi(\cdot)$ is to be maximized, the question is whether $\phi(\iota) \geq K$. The decision and optimization problems given above are associated in this sense.

The input of any problem is also known as an *instance*. In this sense, a problem P is just the set of all its instances.

A.9.3 Complexity of a problem

Let P be a problem, $A(P)$ be the set of algorithms that correctly solve P, and $O(a(n))$ be the worst-case complexity of the algorithm $a \in A(P)$. Then, the worst-case complexity of P is:

$$\min_{a \in A(P)} O(a(n)).$$

Because we write $O(g(n)) = O(f(n))$ whenever g is asymptotically smaller than f, we need not find the absolutely best algorithm a over the infinite set $A(P)$: the "best one found so far" will suffice and provide a worst-case complexity estimate for the problem P.

A.9.4 Easy problems

The Cobham–Edmonds thesis [26, 44] states that a problem is tractable whenever its worst-case complexity is $O(p(n))$, where p is a polynomial in n. If no polynomial-time (or *polytime*) algorithm

is known for P, then P is intractable—at least until a polytime algorithm is found, or is proven not to exist.

The class of all tractable problems is denoted by **P**. A problem in **P** is also informally called *easy*. Also, let $\bar{A}(P)$ the set of all polynomial-time algorithms for solving P.

A.9.5 Nondeterministic polynomially solvable problems

Hardness in worst-case complexity is not the converse of tractability or easiness: the notion of what it means for a decision problem to be hard is more complicated. We start with the class **NP** of decision problems P such that, for each YES instance ι (i.e., one which yields a YES answer from any algorithm in $A(P)$), there is a *certificate* $c(\iota)$ having polynomially bounded length that proves that the answer is correct.

A.9.2 Example
Let P be the HAMILTONIAN CYCLE (HC) problem: given a simple, undirected, and connected graph $G = (V, E)$, does it contain a Hamiltonian cycle? Setting aside the issue of actually finding such a cycle, it is easy to show that if G is a YES instance and $\gamma(G)$ is a Hamiltonian cycle in G, it only takes $O(n)$ to check that γ really is Hamiltonian, where $n = |V|$: it suffices to thread around the cycle from any vertex v back to v again and check that every vertex has been reached in the process. So $\gamma(G)$ is a polynomial certificate for G. Since this algorithm is valid for any instance $G \in P$, and P is a decision problem, then $P \in \mathbf{NP}$.

If $P \in \mathbf{NP}$, $G \in P$ and $a \in A(P)$, then running the algorithm a on the instance G returns a pair (r, γ) where $r \in \{\text{YES}, \text{NO}\}$, and γ is the certificate $\gamma(G)$ if $r = \text{YES}$ or $\varnothing$ if $r = \text{NO}$.

Any problem P in **P** must also be in **NP**: since $P \in \mathbf{P}$ implies the existence of a polynomial-time algorithm $a \in A(P)$, the whole execution trace of $a(G)$ provides a polynomial certificate for the instance $G \in P$.

The reason for the name "nondeterministic polynomially solvable" is technical and shall not be given here.

A.9.6 Polynomial reductions

The K-CLIQUE problem is: given a graph G and $K \in \mathbb{N}$, does G have a K-clique subgraph? The K-STABLE problem asks whether G has a stable subgraph on K vertices. Recall that cliques and stables are related by taking the complement operation $\bar{S}$ on a subgraph S of G (see Sect. A.8.6). Both problems are easily shown to be in **NP**.

Let $a \in A(K\text{-CLIQUE})$ be a solution algorithm for K-CLIQUE. A solution algorithm b for K-STABLE can be constructed using a as shown in Alg. 14. In Step 1, we *reduce* an instance G of K-STABLE to an instance $\bar{G}$ of K-CLIQUE. In Step 2, we run the K-CLIQUE algorithm a on the reduced input $\bar{G}$, and in Steps 3 and following, we apply the inverse reduction to the answer given by a so that it applies to G. The algorithm b (Alg. 14) is correct because a K-stable in G becomes a K-clique in the complement graph $\bar{G}$, and vice versa.

We generalize this to arbitrary problems P, Q (with P playing the role of K-STABLE and Q of K-CLIQUE): if we can solve Q and we can reduce an instance $\alpha \in P$ to an instance $\beta \in Q$ in such a way that an answer to β can be transformed back to an answer to α, then, if we can solve Q, then we can also solve P. If the inverse transformations between instances and between answers can be performed in polytime, then $P \to Q$ is a *polynomial reduction*.

Algorithm 14 An algorithm for solving K- STABLE.

1: compute the complement graph $\bar{G}$;
2: let $(r, \gamma) = a(\bar{G})$;
3: **if** $r =$ NO **then**
4: return $(r, \varnothing)$;
5: **else**
6: return $(r, \bar{\gamma})$.
7: **end if**

The reason why polynomial reductions are important is:

$$\forall P, Q \quad (Q \in \mathbf{NP} \wedge (P \to Q) \Rightarrow P \in \mathbf{NP}) \tag{A.11}$$

$$\forall P, Q \quad (Q \in \mathbf{P} \wedge (P \to Q) \Rightarrow P \in \mathbf{P}). \tag{A.12}$$

Notice that we do not need to actually know an algorithm for Q in order to postulate the existence of a polynomial reduction $P \to Q$: in Alg. 14, we run a as an *oracle*, i.e., without needing to know what actually happens to the computer when a is run. Polynomial reductions are more about proving that two problems are in the same complexity class than actually finding solution algorithms.

A.9.7 Completeness for a class

The discussion in Sect. A.9.6 can be generalized to an arbitrary problem class **C** as follows. Let **C** be a class of problems with worst-case complexities $\{O(f(n)) \mid f \in \mathcal{F}\}$ for some given family of functions $\mathcal{F}$. Let P be any problem, and ρ be a reduction from P taking time $O(g(n))$ with $g \in \mathcal{F}$. If $\rho(P) \in \mathbf{C}$, then $P \in \mathbf{C}$: for any $\iota \in P$, it suffices to compute $\rho(\iota)$, feed it as input to an algorithm $a \in A(\rho(P))$ having worst-case complexity in $\mathcal{F}$ (which must exist since $\rho(P) \in \mathbf{C}$), then inverse-reduce the answer from a to get an answer from ι.

The problem $Q \in \mathbf{C}$ is **C**-*complete* if, for any $P \in \mathbf{C}$, there is a reduction $\rho : P \to Q$.

For example, LINEAR PROGRAMMING (LP) is a **P**-complete problem: any problem in **P** can be reduced to a linear program (under reductions which require at worst a logarithmic amount of memory). HC and K- CLIQUE are **NP**-complete.

A.9.8 Hardness

A problem P is *hard* for a class **C** if every problem in **C** has a reduction to P. This is a weaker definition than **C**-completeness, since it does not require that $P \in \mathbf{C}$. For example, the DGP is **NP**-hard, but no one knows whether it is in **NP** (there is some evidence but not proof against).

Saying that every problem in **C** has a reduction to P means that P is as hard as any problem in **C**, and in particular that P is as hard as the hardest problem in C (since this observation only rests on reduction rather than membership in the class, it also applies to **C**-complete problems). This defines **C**-hard problems as a category of "equally hardest problems in **C**." So, if there is a reduction from a **C**-hard problem P to Q, it means that Q is also **C**-hard.

A.9.9 Hard problems

Informally, a decision problem is *hard* when it is **NP**-hard. Notice that this is not the converse of being easy. Notice also that, although $\mathbf{P} \subseteq \mathbf{NP}$, no one knows whether $\mathbf{P} = \mathbf{NP}$ or not (again, there is evidence but not proof against). So, in the unlikely event that $\mathbf{P} = \mathbf{NP}$, hard problems would also be easy!

Notions of hardness for optimization problems are either based on their associated decision problems, or else on the difficulty of finding provable good approximate solutions.

A.10 Exercises

A.10.1 Exercise
Prove that $\mathbb{Q}$, $\mathbb{R}$, $\mathbb{C}$ are fields and that $\mathbb{Z}$, $\mathbb{N}$, and $\mathbb{R}_+$ are not.

A.10.2 Exercise
Prove that, in a field, $0 + 1 = 1$, $01 = 0$, and for each $\alpha \in \mathbb{F}$, $\alpha \neq 0$ implies that $\alpha^{-1}\alpha = 1$.

A.10.3 Exercise
Prove that the algebraic numbers, i.e., those numbers which are roots of polynomials having rational coefficients, are a field.

A.10.4 Exercise
Prove that the standard basis is indeed a basis.

A.10.5 Exercise
Prove that the intersection of two or more subspaces of a vector space is a subspace. Prove that the union of two or more subspaces of a vector space may not be a subspace. Give an example of the union of two subspaces that *is* a subspace.

A.10.6 Exercise
Prove that the Euclidean norm is indeed a norm.

A.10.7 Exercise
A *path* in a graph is a sequence of vertices $(v_1, \ldots, v_n)$ such that, for all i with $1 < i \leq n$, $\{v_{i-1}, v_i\}$ is an edge of the graph. A *cycle* in a graph is a path such that $v_1 = v_n$. Prove that a simple cycle is a cycle and a simple path is a path, but the converse does not hold.

A.10.8 Exercise
Show that if P is a problem, then its set of solution algorithms $A(P)$ is (countably) infinite.

A.10.9 Exercise
Prove that K- CLIQUE and K- STABLE are in **NP**.

A.10.10 Exercise
Prove Eqs. (A.11)–(A.12).

A.10.11 Exercise
Prove that K- STABLE is **NP**-complete.

A.10.12 Exercise
Let HAMILTONIAN PATH (HP) be the problem of determining whether a given graph has a simple spanning path. Prove that HP is **NP**-complete.

References

1. Alfakih, A.: Universal rigidity of bar frameworks in general position: a Euclidean distance matrix approach. In: Mucherino et al. [102], pp. 3–22
2. Alfakih, A., Khandani, A., Wolkowicz, H.: Solving Euclidean distance matrix completion problems via semidefinite programming. Comput. Optim. Appl. **12**, 13–30 (1999)
3. Asimow, L., Roth, B.: The rigidity of graphs. Trans. Am. Math. Soc. **245**, 279–289 (1978)
4. Asimow, L., Roth, B.: The rigidity of graphs II. J. Math. Anal. Appl. **68**, 171–190 (1979)
5. Bahr, A., Leonard, J., Fallon, M.: Cooperative localization for autonomous underwater vehicles. Int. J. Robot. Res. **28**(6), 714–728 (2009)
6. Bajaj, C.: The algebraic degree of geometric optimization problems. Discret. Comput. Geom. **3**, 177–191 (1988)
7. Bandeira, A., Chen, Y., Singer, A.: Non-unique games over compact groups and orientation estimation in cryo-em. Technical report (2015). arXiv:1505.03840v1
8. Barvinok, A.: Problems of distance geometry and convex properties of quadratic maps. Discret. Comput. Geom. **13**, 189–202 (1995)
9. Beeker, N., Gaubert, S., Glusa, C., Liberti, L.: Is the distance geometry problem in NP? In: Mucherino et al. [102], pp. 85–94
10. Benedetti, R., Risler, J.-J.: Real Algebraic And Semi-algebraic Sets. Hermann, Paris (1990)
11. Berman, H., Westbrook, J., Feng, Z., Gilliland, G., Bhat, T., Weissig, H., Shindyalov, I.N., Bourne, P.: The protein data bank. Nucl. Acid Res. **28**, 235–242 (2000)
12. Berthold, T., Gamrath, G., Gleixner, A., Heinz, S., Koch, T., Shinano, Y.: Solving mixed integer linear and nonlinear problems using the SCIP Optimization Suite. ZIB (2012). http://scip.zib.de/
13. Billinge, S., Duxbury, P., Gonçalves, D., Lavor, C., Mucherino, A.: Assigned and unassigned distance geometry: applications to biological molecules and nanostructures. *4OR* **14**, 337–376 (2016)
14. Biswas, P., Toh, K.-C., Ye, Y.: A distributed SDP approach for large-scale noisy anchor-free graph realization with applications to molecular conformation. SIAM J. Scient. Comput. **30**(3), 1251–1277 (2008)
15. Biswas, P., Ye, Y.: Semidefinite programming for ad hoc wireless sensor network localization. In: Proceedings of the 3rd International Symposium on Information Processing in Sensor Networks (IPSN04), pp. 46–54. ACM, New York, NY, USA (2004)
16. Björner, A., Las Vergnas, M., Sturmfels, B., White, N., Ziegler, G.: Oriented Matroids. Cambridge University Press, Cambridge (1993)
17. Blumenthal, L.: Theory And Applications Of Distance Geometry. Oxford University Press, Oxford (1953)
18. Blumenthal, L.: A Modern View Of Geometry. Freeman & C, San Francisco (1961)
19. Borcea, C., Streinu, I.: On the number of embeddings of minimally rigid graphs. Discret. Comput. Geom. **31**(2), 287–303 (2004)
20. Bowers, J., Bowers, P.: A Menger redux: embedding metric spaces isometrically. Am. Math. Mon. (accepted)
21. Carvalho, R., Lavor, C., Protti, F.: Extending the geometric build-up algorithm for the molecular distance geometry problem. Inf. Process. Lett. **108**, 234–237 (2008)
22. Cassioli, A., Günlük, O., Lavor, C., Liberti, L.: Complexity of discretization vertex orders for distance geometry. Technical report, LIX, Ecole Polytechnique (2013)
23. Cassioli, A., Günlük, O., Lavor, C., Liberti, L.: Discretization vertex orders for distance geometry. Discret. Appl. Math. **197**, 27–41 (2015)
24. Cauchy, A.-L.: Sur les polygones et les polyèdres. J. de l'École Polytech. **16**(9), 87–99 (1813)
25. Cayley, A.: A theorem in the geometry of position. Camb. Math. J. **II**, 267–271 (1841)

L. Liberti and C. Lavor, *Euclidean Distance Geometry*, Springer Undergraduate Texts in Mathematics and Technology, DOI 10.1007/978-3-319-60792-4

26. Cobham, A.: The intrinsic computational difficulty of functions. In: Bar-Hillel, Y. (ed.) Logic, Methodology and Philosophy of Science, pp. 24–30. North-Holland, Amsterdam (1965)
27. Connelly, R.: A counterexample to the rigidity conjecture for polyhedra. Pub. Mathématiques de l'IHES **47**, 333–338 (1978)
28. Connelly, R.: On generic global rigidity, applied geometry and discrete mathematics. In: DIMACS Series in Discrete Mathematics and Theoretical Computer Science, vol. 4. American Mathematical Society, Providence (1991)
29. Costa, V., Mucherino, A., Lavor, C., Cassioli, A., Carvalho, L., Maculan, N.: Discretization orders for protein side chains. J. Glob. Optim. **60**, 333–349 (2014)
30. Cox, T., Cox, M.: Multidimensional Scaling. Chapman & Hall, Boca Raton (2001)
31. Crippen, G., Havel, T.: Distance Geometry and Molecular Conformation. Wiley, New York (1988)
32. Dasgupta, S., Gupta, A.: An elementary proof of a theorem by Johnson and Lindenstrauss. Random Structures and Algorithms **22**, 60–65 (2002)
33. Dattorro, J.: Convex Optimization and Euclidean Distance Geometry. $\mathcal{M}\epsilon\beta oo$, Palo Alto (2015)
34. Davis, R., Ernst, C., Wu, D.: Protein structure determination via an efficient geometric build-up algorithm. BMC Struct. Biol. **10**(Suppl 1), S7 (2010)
35. Demmel, J.: Applied Numerical Linear Algebra. SIAM, Philadelphia (1997)
36. Descartes, R.: Discours de la Méthode. Ian Maire, Leiden (1637)
37. Dias, G., Liberti, L.: Diagonally dominant programming in distance geometry. In: Cerulli, R., Fujishige, S., Mahjoub, R. (eds.) International Symposium In Combinatorial Optimization. LNCS, vol. 9849, pp. 225–236. Springer, New York (2016)
38. Ding, Y., Krislock, N., Qian, J., Wolkowicz, H.: Sensor network localization, Euclidean distance matrix completions, and graph realization. Optim. Eng. **11**, 45–66 (2010)
39. Dokmanić, I., Parhizkar, R., Ranieri, J., Vetterli, M.: Euclidean distance matrices: Essential theory, algorithms and applications. IEEE Signal Process. Mag. **1053–5888**, 12–30 (2015)
40. Donald, B.: Algorithms in Structural Molecular Biology. MIT Press, Boston (2011)
41. Dong, Q., Wu, Z.: A linear-time algorithm for solving the molecular distance geometry problem with exact inter-atomic distances. J. Glob. Optim. **22**, 365–375 (2002)
42. Dong, Q., Wu, Z.: A geometric build-up algorithm for solving the molecular distance geometry problem with sparse distance data. J. Glob. Optim. **26**, 321–333 (2003)
43. Duxbury, P., Granlund, L., Juhas, P., Billinge, S.: The unassigned distance geometry problem. Discret. Appl. Math. **204**, 117–132 (2016)
44. Edmonds, J.: Paths, trees and flowers. Can. J. Math. **17**, 449–467 (1965)
45. Emiris, I., Tsigaridas, E., Varvitsiotis, A.: Mixed volume and distance geometry techniques for counting Euclidean embeddings of rigid graphs. In: Mucherino et al. [102], pp. 23–46
46. Eren, T., Goldenberg, D., Whiteley, W., Yang, Y., Morse, A., Anderson, B., Belhumeur, P.: Rigidity, computation, and randomization in network localization. IEEE, 2673–2684 (2004)
47. Euclid. Elements. Alexandria, ∼300BC
48. Euler, L.: Continuatio fragmentorum ex adversariis mathematicis depromptorum: II Geometria, 97. In: Fuss, P., Fuss, N. (eds.) Opera Postuma Mathematica Et Physica Anno 1844 Detecta, vol. I, pp. 494–496. Eggers & C, Petropolis (1862)
49. Floyd, R.W.: Algorithm 97: shortest path. Commun. ACM **5**(6), 345 (1962)
50. Franzen, T.: Gödel's Theorem: An Incomplete Guide to its Use and Abuse. Peters, Wellesley (2005)
51. Garey, M., Johnson, D.: Computers and Intractability: A Guide to the Theory of Np-completeness. Freeman and Company, New York (1979)
52. Gluck, H.: Almost all simply connected closed surfaces are rigid. In: Dold, A., Eckmann, B. (eds.) Geometric Topology. Lecture Notes in Mathematics, vol. 438, pp. 225–239. Springer, Berlin (1975)
53. Gödel, K.: On the isometric embeddability of quadruples of points of r_3 in the surface of a sphere. In: Feferman, S., Dawson, J., Kleene, S., Moore, G., Solovay, R., van Heijenoort, J. (eds.) Kurt Gödel: Collected Works, vol. I, pp. 276–279. Oxford University Press, Oxford (1986). 1933b
54. Gonçalves, D., Mucherino, A.: Discretization orders and efficient computation of cartesian coordinates for distance geometry. Optim. Lett. **8**, 2111–2125 (2014)
55. Gortler, S., Healy, A., Thurston, D.: Characterizing generic global rigidity. Am. J. Math. **132**(4), 897–939 (2010)
56. Graver, J.: Rigidity matroids. SIAM J. Discret. Math. **4**, 355–368 (1991)
57. Graver, J., Servatius, B., Servatius, H.: Combinatorial Rigidity. American Mathematical Society (1993)
58. Henneberg, L.: Die Graphische Statik Der Starren Systeme. Teubner, Leipzig (1911)
59. Heron. Metrica, vol. I. Alexandria, ∼100AD
60. IBM. ILOG CPLEX 12.2 User's Manual. IBM (2010)
61. Jackson, B., Jordán, T.: Connected rigidity matroids and unique realization of graphs. J. Comb. Theory Ser. B **94**, 1–29 (2005)

62. Johnson, W., Lindenstrauss, J.: Extensions of Lipschitz mappings into a Hilbert space. In: Hedlund, G. (ed.) Conference in Modern Analysis and Probability. Contemporary Mathematics, vol. 26, pp. 189–206. American Mathematical Society, Providence (1984)
63. Jolliffe, I.: Principal Component Analysis, 2nd edn. Springer, Berlin (2010)
64. Kolmogorov, A., Fomin, S.: Measure. Lebesgue Integrals and Hilbert Space. Academic Press, New York (1960)
65. Kolmogorov, A., Fomin, S.: Introductory Real Analysis. Dover, New York (1975)
66. Krislock, N., Wolkowicz, H.: Explicit sensor network localization using semidefinite representations and facial reductions. SIAM J. Optim. **20**, 2679–2708 (2010)
67. Kunen, K.: Set Theory. An Introduction to Independence Proofs. North Holland, Amsterdam (1980)
68. Laman, G.: On graphs and rigidity of plane skeletal structures. J. Eng. Math. **4**(4), 331–340 (1970)
69. Langville, A., Meyer, C.: Google's Pagerank and Beyond: The Science of Search Engine Rankings. Princeton University Press, Princeton, NJ (2006)
70. Laurent, M.: Cuts, matrix completions and graph rigidity. Math. Program. **79**, 255–283 (1997)
71. Laurent, M.: A connection between positive semidefinite and Euclidean distance matrix completion problems. Linear Algebr. Appl. **273**, 9–22 (1998)
72. Laurent, M.: Polynomial instances of the positive semidefinite and Euclidean distance matrix completion problems. SIAM J. Matrix Anal. Appl. **22**(3), 874–894 (2000)
73. Laurent, M.: Matrix completion problems. In: Floudas, C., Pardalos, P. (eds.) Encyclopedia of Optimization, 2nd edn, pp. 1967–1975. Springer, New York (2009)
74. Lavor, C.: On generating instances for the molecular distance geometry problem. In: Liberti, L., Maculan, N. (eds.) Global Optimization: From Theory To Implementation, pp. 405–414. Springer, Berlin (2006)
75. Lavor, C., Firer, M., Martinez, J.-M., Liberti, L.: Preface. Int. Trans. Oper. Res. **23**(5), 841 (2016)
76. Lavor, C., Lee, J., John, L.-S.A., Liberti, L., Mucherino, A., Sviridenko, M.: Discretization orders for distance geometry problems. Optim. Lett. **6**, 783–796 (2012)
77. Lavor, C., Liberti, L., Lodwick, W., Mendonça da Costa, T.: An Introduction to Distance Geometry Applied to Molecular Geometry. SpringerBriefs. Springer, New York (2017)
78. Lavor, C., Liberti, L., Maculan, N.: Computational experience with the molecular distance geometry problem. In: Pintér, J. (ed.) Global Optimization: Scientific and Engineering Case Studies, pp. 213–225. Springer, Berlin (2006)
79. Lavor, C., Liberti, L., Maculan, N., Mucherino, A.: The discretizable molecular distance geometry problem. Comput. Optim. Appl. **52**, 115–146 (2012)
80. Lavor, C., Liberti, L., Maculan, N., Mucherino, A.: Recent advances on the discretizable molecular distance geometry problem. Eur. J. Oper. Res. **219**, 698–706 (2012)
81. Liberti, L., Lavor, C.: Six mathematical gems in the history of distance geometry. Int. Trans. Oper. Res. **23**, 897–920 (2016)
82. Liberti, L., Lavor, C., Alencar, J., Abud, G.: Counting the number of solutions of ^{k}DMDGP instances. In: Nielsen and Barbaresco [103], pp. 224–230
83. Liberti, L., Lavor, C., Maculan, N.: A branch-and-prune algorithm for the molecular distance geometry problem. Int. Trans. Oper. Res. **15**, 1–17 (2008)
84. Liberti, L., Lavor, C., Maculan, N., Mucherino, A.: Euclidean distance geometry and applications. SIAM Rev. **56**(1), 3–69 (2014)
85. Liberti, L., Lavor, C., Mucherino, A.: The discretizable molecular distance geometry problem seems easier on proteins. In: Mucherino et al. [102], pp. 47–60
86. Liberti, L., Lavor, C., Mucherino, A., Maculan, N.: Molecular distance geometry methods: from continuous to discrete. Int. Trans. Oper. Res. **18**, 33–51 (2010)
87. Liberti, L., Masson, B., Lavor, C., Lee, J., Mucherino, A.: On the number of realizations of certain Henneberg graphs arising in protein conformation. Discret. Appl. Math. **165**, 213–232 (2014)
88. Lovász, L., Yemini, Y.: On generic rigidity in the plane. SIAM J. Algebr. Discret. Methods **3**(1), 91–98 (1982)
89. Makhorin, A.: GNU Linear Programming Kit. Free Software Foundation (2003). http://www.gnu.org/software/glpk/
90. Man-Cho So, A., Ye, Y.: Theory of semidefinite programming for sensor network localization. Math. Program. B **109**, 367–384 (2007)
91. Maxwell, J.: On reciprocal figures and diagrams of forces. Philos. Mag. **27**(182), 250–261 (1864)
92. Maxwell, J.: On the calculation of the equilibrium and stiffness of frames. Philos. Mag. **27**(182), 294–299 (1864)
93. Menger, K.: Untersuchungen über allgemeine Metrik. Mathematische Annalen **100**, 75–163 (1928)
94. Menger, K.: New foundation of Euclidean geometry. Am. J. Math. **53**(4), 721–745 (1931)
95. Menger, K. (ed.): Ergebnisse Eines Mathematischen Kolloquiums. Springer, Wien (1998)
96. Moré, J., Wu, Z.: Global continuation for distance geometry problems. SIAM J. Optim. **7**(3), 814–846 (1997)
97. Mucherino, A.: On the identification of discretization orders for distance geometry with intervals. In: Nielsen and Barbaresco [103], pp. 231–238
98. Mucherino, A., de Freitas, R., Lavor, C.: Preface. Discret. Appl. Math. **197**, 1–2 (2015)

99. Mucherino, A., Lavor, C., Liberti, L.: A symmetry-driven BP algorithm for the discretizable molecular distance geometry problem. In: Proceedings of Computational Structural Bioinformatics Workshop, pp. 390–395. IEEE, Piscataway (2011)
100. Mucherino, A., Lavor, C., Liberti, L.: The discretizable distance geometry problem. Optim. Lett. **6**, 1671–1686 (2012)
101. Mucherino, A., Lavor, C., Liberti, L.: Exploiting symmetry properties of the discretizable molecular distance geometry problem. J. Bioinform. Comput. Biol. **10**(1–15), 1242009 (2012)
102. Mucherino, A., Lavor, C., Liberti, L., Maculan, N. (eds.): Distance Geometry: Theory, Methods, and Applications. Springer, New York (2013)
103. Nielsen, F., Barbaresco, F. (eds.): Geometric Science of Information, vol. 8085. LNCS. Springer, New York (2013)
104. Papadimitriou, C.: Computational Complexity. Addison-Wesley, Reading, MA (1994)
105. Recski, A.: A network theory approach to the rigidity of skeletal structures. Part 2. Laman's theorem and topological formulae. Discret. Appl. Math. **8**, 63–68 (1984)
106. Rojas, N., Thomas, F.: Application of distance geometry to tracing coupler curves of pin-jointed linkages. J. Mech. Robotics **5**(2), 021001 (2013)
107. Saxe. J.: Embeddability of weighted graphs in k-space is strongly **NP**-hard. In: Proceedings of 17th Allerton Conference in Communications, Control and Computing, pp. 480–489 (1979)
108. Schlick, T.: Molecular Modelling and Simulation: An Interdisciplinary Guide. Springer, New York (2002)
109. Schoenberg, I.: Remarks to Maurice Fréchet's article "Sur la définition axiomatique d'une classe d'espaces distanciés vectoriellement applicable sur l'espace de Hilbert". Ann. Math. **36**(3), 724–732 (1935)
110. Singer, A.: Angular synchronization by eigenvectors and semidefinite programming. Appl. Comput. Harmon. Anal. **30**, 20–36 (2011)
111. Singer, A., Zhao, Z., Shkolnisky, Y., Hadani, R.: Viewing angle classification of cryo-electron microscopy images using eigenvectors. SIAM J. Imaging Sci. **4**(2), 543–572 (2011)
112. Sit, A., Wu, Z., Yuan, Y.: A geometric build-up algorithm for the solution of the distance geometry problem using least-squares approximation. Bull. Math. Biol. **71**, 1914–1933 (2009)
113. Sitharam, M., Zhou,Y.: A tractable, approximate, combinatorial 3D rigidity characterization. In: Fifth Workshop on Automated Deduction in Geometry (2004)
114. Spivak, M.: Calculus On Manifolds. Addison-Wesley, Reading, MA (1965)
115. Stoll, R.: Set Theory and Logic. Dover, New York (1979)
116. Tay, T.-S., Whiteley, W.: Generating isostatic frameworks. Struct. Topol. **11**, 21–69 (1985)
117. Tenenbaum, J., de Silva, V., Langford, J.: A global geometric framework for nonlinear dimensionality reduction. Science **290**, 2319–2322 (2000)
118. Thorpe, M., Duxbury, P. (eds.): Rigidity Theory and Applications. Fundamental Materials Research. Springer, New York (2002)
119. Venkatasubramanian, S., Wang, Q.: The Johnson-Lindenstrauss transform: an empirical study. In: Algorithm Engineering and Experiments, vol. 13, pp. 164–173. ALENEX. SIAM, Providence (2011)
120. Wu, D., Wu, Z.: An updated geometric build-up algorithm for solving the molecular distance geometry problem with sparse distance data. J. Glob. Optim. **37**, 661–673 (2007)
121. Wüthrich, K.: Protein structure determination in solution by nuclear magnetic resonance spectroscopy. Science **243**, 45–50 (1989)
122. Wüthrich, K., Billeter, M., Braun, W.: Pseudo-structures for the 20 common amino acids for use in studies of protein conformations by measurements of intramolecular proton-proton distance constraints with nuclear magnetic resonance. J. Mol. Biol. **169**, 949–961 (1983)
123. Xu, H., Izrailev, S., Agrafiotis, D.: Conformational sampling by self-organization. J. Chem. Inf. Comput. Sci. **43**, 1186–1191 (2003)

Index

L. Liberti and C. Lavor, *Euclidean Distance Geometry*, Springer Undergraduate Texts in Mathematics and Technology, DOI 10.1007/978-3-319-60792-4

G

S