

Linear Algebra, First Part

Lecture Notes HS25

Bernd Gärtner
Department of Computer Science
ETH Zürich

September 16, 2025

Contents

0	Introduction	4
0.1	About these notes	4
0.1.1	Major changes compared to HS24	5
0.2	About computer science (and mathematics)	6
0.3	About linear algebra	7
1	Vectors	11
1.1	Vectors and linear combinations	11
1.1.1	Vector addition	14
1.1.2	Scalar multiplication	15
1.1.3	Linear combinations	16
1.1.4	Affine, conic, and convex combinations	19
1.1.5	Defining the dots: sequences, sums, sets, and vectors	21
1.2	Scalar products, lengths and angles	23
1.2.1	Scalar product	24
1.2.2	Euclidean norm	25
1.2.3	Cauchy-Schwarz inequality	28
1.2.4	Angles	30
1.2.5	Triangle inequality	33
1.2.6	Covectors and $\mathbf{v}^\top \mathbf{w}$	34
1.3	Linear (in)dependence	36
1.3.1	Definition and examples	36
1.3.2	Alternative definitions	38
1.3.3	Span of vectors	40
2	Matrices	45
2.1	Matrices and linear combinations	46
2.1.1	Matrix-vector multiplication	49
2.1.2	Column space and rank	51
2.1.3	Row space and transpose	53
2.1.4	Rank-1 matrices	55
2.1.5	Nullspace	57
2.2	Matrices and linear transformations	58

2.2.1	Matrix transformations	58
2.2.2	Linear transformations and linear functionals	63
2.2.3	The matrix of a linear transformation	67
2.2.4	Kernel and Image	68
2.3	Matrix multiplication	69
2.3.1	Combining matrix transformations	70
2.3.2	Definition and basic properties	71
2.3.3	Distributivity and associativity	75
2.3.4	Everything “is” matrix multiplication	76
2.3.5	CR decomposition	82
2.4	Invertible and Inverse matrices	85
2.4.1	Undoing matrix transformations	85
2.4.2	Definitions and basic properties	89
3	Solving Linear Equations $Ax = b$	93
3.1	Systems of linear equations	94
3.1.1	The PageRank algorithm	95
3.1.2	Matrix inversion	98
3.2	Gauss elimination	98
3.2.1	Back substitution	99
3.2.2	Elimination	100
3.2.3	Row operations: the general picture	104
3.2.4	Success and failure	107
3.2.5	Runtime	109
3.2.6	Computing inverse matrices	110
3.2.7	Solving $Ax = b$ from A^{-1} , LU, and LUP	113
3.3	Gauss-Jordan elimination	114
3.3.1	Reduced row echelon form	114
3.3.2	Direct solution	117
3.3.3	Elimination	118
3.3.4	Standard form and CR decomposition	122
3.3.5	Computing inverse matrices, and solving $Ax = b$ from R and M . .	124
4	The Three Fundamental Subspaces	126
4.1	Vector spaces	127
4.1.1	Definition and examples	127
4.1.2	Subspaces	130
4.2	Bases and dimension	134
4.2.1	Linear combinations and related concepts in vector spaces	134
4.2.2	Bases	136
4.2.3	The Steinitz exchange lemma	140
4.2.4	Dimension	142
4.2.5	Linear transformations between vector spaces	143

4.2.6	Computing a vector space	147
4.3	Computing the three fundamental subspaces	148
4.3.1	Column space	148
4.3.2	Row space	149
4.3.3	Nullspace	150
4.4	All solutions of $Ax = \mathbf{b}$	154
4.4.1	Affine subspaces	156
4.4.2	The number of solutions	156
Bibliography		159
Index		160

Chapter 0

Introduction

0.1 About these notes

These are the lecture notes for the first part of the course

Lineare Algebra (401-0131-00L)

held at the Department of Computer Science at ETH Zürich in HS25.

These notes can be considered as a full version of what will be written on the tablet during the lectures. Lecture plans will be made available before each lecture. The tablet notes (in German) reflect the reality and will be made available after each lecture. Together with the explanations (and answers to questions) given in the lectures, they will contain the “essentials”, but in order to fully understand them, it can be helpful to look up more details and additional explanations in the lecture notes.

In content, the lecture notes are loosely based on the first 8 chapters of the book

Introduction to Linear Algebra (Sixth Edition) by Gilbert Strang, Wellesley - Cambridge Press, 2023 [Str23].

The ordering of material is somewhat different here. But the main difference is that the approach taken here is more rigorous than Strang’s. Strang introduces the material on an intuitive level, guided by many examples; this provides a great informal introduction to Linear Algebra. What we add here (hopefully without losing the intuition) are formal definitions of concepts, as well as mathematical statements with proofs for the key results.

Strang’s book is *not* part of the course’s official material, and there is no need for students to buy the book (it doesn’t have an official electronic version). With the lectures, lecture plans, tablet notes, lecture notes, exercises, and exercises classes, the course is self-contained. Strang’s book and others mentioned on the course web page serve as optional literature.

There are a number of major changes (on top of many minor changes to improve clarity), compared to the lecture notes of HS24. We outline them next, including brief explanations why these changes were made.

0.1.1 Major changes compared to HS24

Column vector notation. This is the most visible change. For example, instead of

$$\begin{bmatrix} 1 \\ 2 \end{bmatrix} \in \mathbb{R}^2, \text{ we now write } \begin{pmatrix} 1 \\ 2 \end{pmatrix} \in \mathbb{R}^m.$$

The reason is that the square bracket notation was (and still is) used for matrices as well, so it was previously not possible to visually distinguish between a vector in \mathbb{R}^2 and 2×1 matrix. In most cases, one could tell this from the context; however, always using square brackets creates the (wrong) impression that vectors and matrices with one column are the same objects. This has previously led to quite some confusion. Now, we clearly distinguish between vectors and matrices with one column, and this also shows in the notation. We are now also clear about the difference between row and column vectors; previously, we used both to write vectors in \mathbb{R}^m , but now we consistently use column vectors for \mathbb{R}^m , and row vectors for the dual space $(\mathbb{R}^m)^*$.

Matrix multiplication (AB), matrix inversion (A^{-1}). These two important matrix operations are now motivated from “combining” and “undoing” linear transformations. Previously, they came a little “out of the blue”. Both operations are now introduced in Chapter 2 where they logically belong. Previously, matrix inversion was introduced after Gauss elimination, because only at this point, we had the necessary theory for it. The crucial (previously missing) piece of theory now already occurs in Chapter 1, in the form of the rather innocent-looking but impactful Lemma 1.28.

LU and LUP decompositions. These have been removed from the notes. Previously, the LU decomposition was presented only for a special class of matrices (the ones for which Gauss elimination works without row exchanges), and the LUP decomposition was only sketched in class. For theoretical (non-numerical) purposes, both LU and LUP decompositions are of minor relevance, since Gauss-Jordan elimination (that we present in full detail now) can easily replace them, without any restrictions on the matrix. As a consequence, our longstanding first exam problem (compute the LU decomposition of a 3×3 matrix) will be replaced. The new first exam problem will ask you to solve a 3×3 system of linear equations.

Linear transformations between vector spaces. Previously, we have introduced linear transformations as functions from some \mathbb{R}^n to some \mathbb{R}^m . However, the right level of abstraction is to consider linear transformations between two vector spaces V and W . Since we cover vector spaces anyway, it is easy to also include this more abstract view of linear transformations. This now allows us to formally define when two vector spaces are “essentially the same”, a concept that we have previously only covered informally. As a concrete benefit, a basis of the nullspace of a matrix can now be derived in a much more transparent and systematic way than previously.

0.2 About computer science (and mathematics)

As a first semester student of computer science, you may wonder why one of the first things you see is mathematics, and in particular linear algebra. In order to answer this, we first need to answer a different question: *What is computer science?*

Computer scientists are frequently approached for help with installing computers, getting the internet to work, and similar technical tasks. To many people, a computer scientist is simply an information technology expert. But this point of view is fundamentally wrong. The computer scientist Mike Fellows has explained this very well already in 1991, using a simple analogy:

Computer science is not about machines in the same way that astronomy is not about telescopes. There is an essential unity of mathematics and computer science [Fel93].

The first sentence of this quote (often misattributed to Edward Dijkstra) is well-known and gets the main point across: computer science is *not* mainly about computers. Everyone agrees that computers, just like telescopes, are great tools, but they merely help us in achieving some goals, they are not the goals themselves. It is true that astronomers are involved in building telescopes, and computer scientists are involved in building computers. Generally, if you need a tool that you cannot buy off the shelf, you will team up with people that can build it for you, and this needs expertise from both sides. But in the end, you want to use the tool for *something*, so you are not mainly interested in the tool itself.

The second part of the quote is less known, but not less important. It indicates that computer science and mathematics are very strongly connected. To understand this, we need to say what computer science actually is. If you ask the internet for a definition of computer science, you get many wrong answers that start with “the study of computers...”, even from serious sources. The Wikipedia article about computer science starts differently and does not mention computers in the first sentence:¹

Computer science is the study of computation, information, and automation.

This is correct but a bit too short; the German version of the page,² has a more detailed and clearer definition, adapted from the “Duden Informatik A-Z” [CSB06, Eintrag Informatik, S. 305].

Informatik ist die Wissenschaft von der systematischen Darstellung, Speicherung, Verarbeitung und Übertragung von Informationen, wobei in der Regel die automatische Verarbeitung mit Computern betrachtet wird.

¹https://en.wikipedia.org/wiki/Computer_science, accessed on September 2, 2025

²<https://de.wikipedia.org/wiki/Informatik>, accessed on September 2, 2025

In English, this reads as follows:

Computer science is the science of the systematic representation, storage, processing and transfer of information, where one usually considers the automatic processing using computers.

This means, computer science is about dealing with *information*. The German term *Informatik* transports this message much better than the English term *computer science*.

As an example, let's consider addition as you (and children throughout many centuries) have learned it in primary school, for example

$$\begin{array}{r} 123 \\ + 486 \\ \hline = 609 \end{array}$$

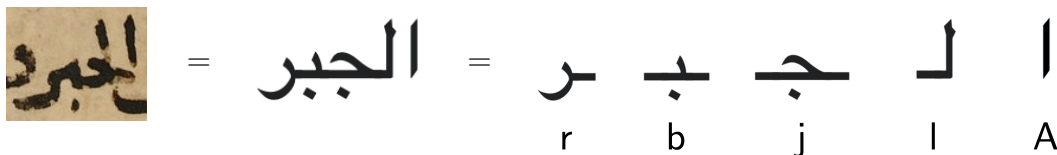
This is systematic processing of information (in this case numbers), represented in decimal place-value system. Hence, this is not only mathematics, but according to the above definition, it is also computer science! In modern terms, we would call *school book addition* an *algorithm*. Computers can do such additions automatically and very fast (this is what the “where one...” part of the definition is about), but the algorithm itself was invented much earlier than the computers.

While an algorithm is mostly associated with computer science, the theoretical foundations of many algorithms and other computer science inventions are inherently mathematical. School book addition is a prime example. Here, the most important theoretical foundation is the place-value system. This is one of the great historical developments in mathematics, driven by the need to efficiently compute with numbers.

Today, there are new needs, for example, efficiently training huge machine learning models, or securing computer systems against cyberattacks. This needs established as well as new mathematics. Mathematical research is often motivated by applications in computer science, and mathematicians work with computer science tools on a daily basis. For these reasons, every computer science student needs mathematical foundations, and every mathematics student needs computer science foundations. An essential unity, indeed.

0.3 About linear algebra

The origins of (linear) algebra can be traced back to the 9th century when the Persian polymath *Al-Khwarizmi* published *The Compendious Book on Calculation by Completion and Balancing*. The word *algebra* also goes back to this book, see the highlighted part on the title page shown in Figure 1. In modern Arabic letters and transcribed to Latin letters (right to left), this reads as follows:



ر ب ج ل ا
r b j l A

Al Khwarizmi's book teaches how to systematically solve linear and quadratic equations in one variable. While this is basic highschool material today, the theory underlying it had to be developed at some point, and it was Al-Khwarizmi who did this; for example, the word *balancing* in the title of his work refers to the technique of moving a term to the other side of an equation, something you need to do when you want to solve for a variable.

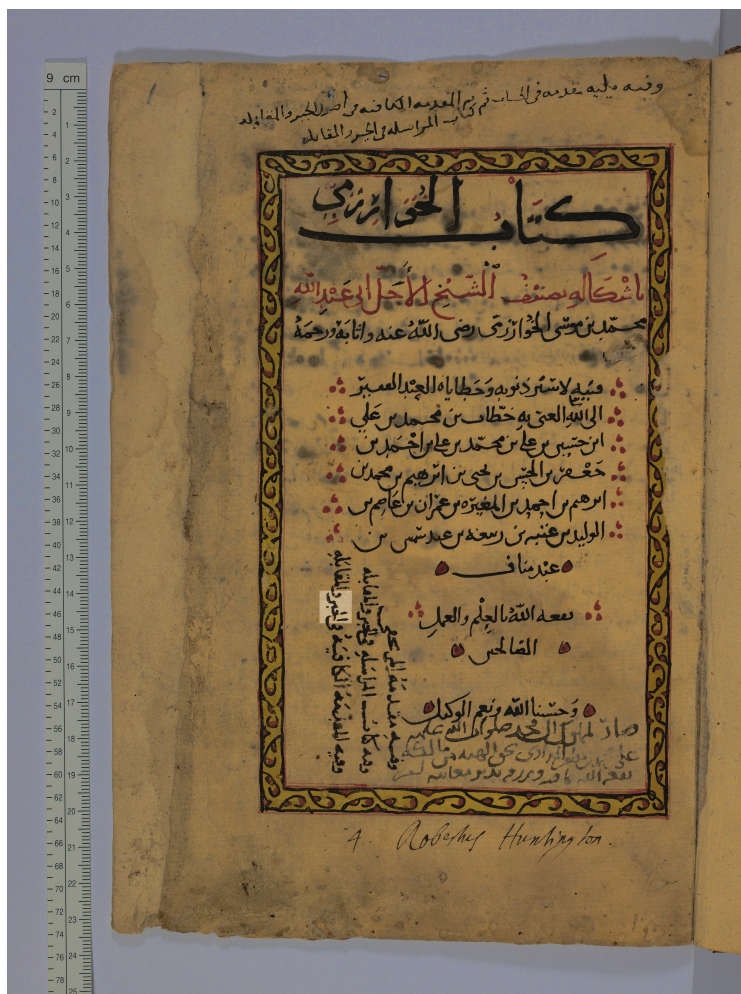


Figure 1: Title page of Al-Khwarizmi's book; the highlighted text in the lines written from bottom to top is the Arabic word *Al-jabr*. This is a higher-resolution image of the one found on Wikipedia (<https://en.wikipedia.org/wiki/Al-Jabr>), provided to the author by Digital Bodleian, <https://digital.bodleian.ox.ac.uk>, CC-BY-NC 4.0. A translation of the title page can be found here.

The field of linear algebra has developed from two historical roots: *analytic geometry* and *linear equations*. Analytic geometry deals with the description of and calculation with geometric objects through coordinates and formulas. We could also call it "rigorous"

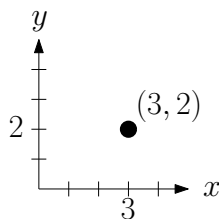


Figure 2: A point in the two-dimensional plane, expressed in Cartesian coordinates

geometry. Everybody knows what a circle is, but if you want to describe a particular circle, you need to say what the center and the radius are. The center is a point that you typically describe with *Cartesian coordinates*, as in Figure 2.

Having such rigorous descriptions of geometric objects, you can start to answer questions about them *analytically* by using mathematics; for example, do two given lines in three-dimensional space intersect or not?

In answering this and many other questions, systems of linear equations in more than one variable come up, and this is the second root of linear algebra. Today, such systems are considered as easy to solve in many cases, but this is the result of developments that happened over centuries. And they still happen now: solving systems of linear equations is an active research topic, in particular since the traditional algorithms cannot handle the very large systems that we have in many applications today. Even small systems are not necessarily easy for humans and form a basis of many puzzles. Consider this one:

Dominik is twice as old as Susanne and three years older than Claudia.
Together, the children are 17 years old. How old are the three children?

Even without having heard about linear equations, you will be able to figure this out, employing some guesswork and mental arithmetic. But this quickly reaches its limits when more children are involved in the puzzle. There is a reason why such puzzles that are made for entertainment rarely have more than three variables (in this case, the ages of the children). As a system of three linear equations, the puzzle reads as follows:

$$\begin{aligned} D &= 2S \\ D &= C + 3 \\ D + S + C &= 17 \end{aligned}$$

Here, D , S , and C are variables for the unknown ages of the children, and the three equations encode the three pieces of information that the puzzle provides. The equations are *linear* because every variable occurs only in the first power. In contrast, $x^2 - 2x + 3 = 0$ is a quadratic equation, because the variable x occurs in the second power as well. There are also cubic, quartic, quintic, etc. equations. Solving systems of those falls into the domain of (non-linear) algebra. Linear algebra deals with m linear equations in n variables, where both numbers m and n can be very large in practice.

In the above linear puzzle equations, you can convince yourself that there are unique numbers that you can plug in for D , S , and C such that all three equations are satisfied. Conveniently, these numbers are natural numbers; you don't want a puzzle where a child is 2.7 years or -3 years old. But in principle, the solutions to a system of linear equations can be arbitrary numbers, and it is up to the application to decide whether these make sense or not.

Interestingly, Al-Khwarizmi only provided formulas for positive solutions of equations in his book, as he thought that only those make sense. Indeed, in a world where numbers count physical quantities, the idea of a negative number seems downright crazy. However, in Al-Khwarizmi's computations (including "balancing"), negative numbers implicitly occur in order to arrive at the positive solutions.

Starting from the roots of analytic geometry and systems of linear equations, linear algebra has grown many important branches, some of which appear in Figure 3 and also later in these notes.

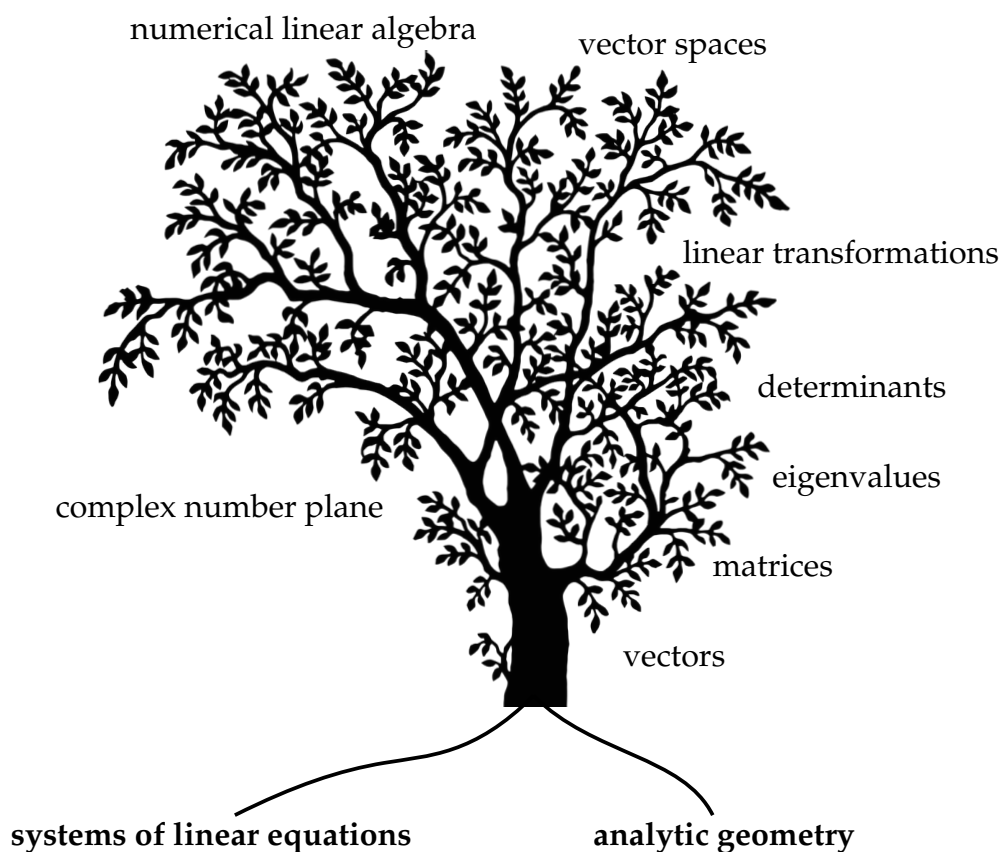


Figure 3: The tree of linear algebra: Roots and some important branches

Chapter 1

Vectors

Asking ChatGPT what a vector is will return something like this: “A vector is a mathematical object that has both magnitude (length) and direction.” This is in line with how we visualize vectors in analytic geometry, namely as *arrows* that naturally have a length and a direction:



These arrow pictures are good starting points, but in linear algebra, we will (have to) do more. We can draw arrows in 2- and even 3-dimensional space, but we cannot do this in 10-dimensional space. Higher-dimensional spaces where the geometric intuition fails us are important in applications, therefore we need an actual definition of the mathematical object behind a vector. That a vector has (according to ChatGPT) both length and direction is a *property* of the object, but not a definition. On top of this, it is not clear what “length” and “direction” mean in higher dimensions.

The language of pictures is extremely useful for the intuitive understanding of mathematical objects, and we will keep using this language whenever we can. But if we want to logically argue about the objects (in particular when we cannot “see” them anymore in high dimensions), we need the language of mathematics. Mathematical objects are imaginary and not physical, but once we can talk about them, they exist in some way (for example, you can think about the way in which the number 5 exists).

In this chapter, we get to know the mathematical language around the (coordinate) vectors whose 2- and 3-dimensional versions we know from analytic geometry. But in Section 4.1, we will discover that there are also other kinds of vectors.

1.1 Vectors and linear combinations

Vectors in \mathbb{R}^m and their linear combinations are fundamental in linear algebra and at the same time easy to understand. In fact, you may know much of this material from high school. This leaves room to also learn about some important elements of mathematical thinking and writing. For example, you will see a first proof.

Vectors as you know them from high school “live” in 2-dimensional or 3-dimensional space, see Figure 1.1.

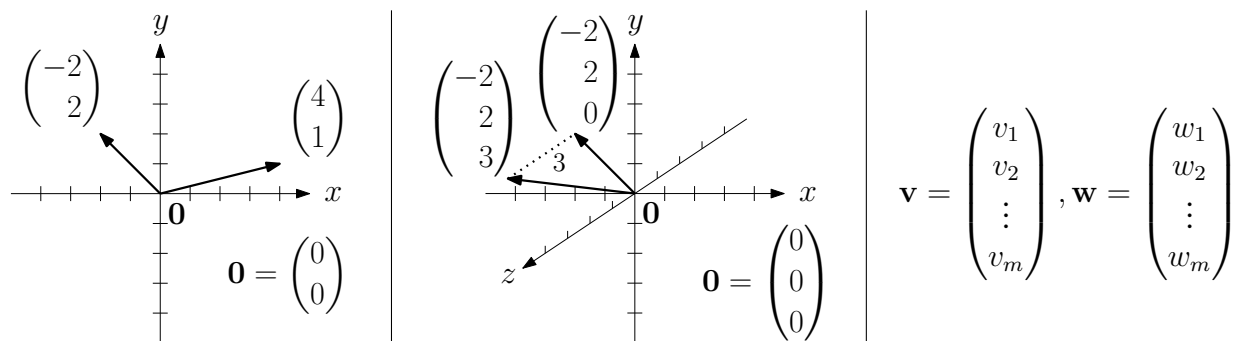


Figure 1.1: Vectors in the plane \mathbb{R}^2 (left), in space \mathbb{R}^3 (middle) and in \mathbb{R}^m (right)

More abstractly, they live in some m -dimensional space \mathbb{R}^m , where $m \in \mathbb{N}$, the set $\{0, 1, 2, \dots\}$ of *natural numbers*; see Figure 1.1 (right).¹ Starting from $m = 4$, these spaces are hard to visualize, but linear algebra can handle them as easily as \mathbb{R}^2 and \mathbb{R}^3 .

Mathematically, these spaces are sets that are called \mathbb{R}^2 , \mathbb{R}^3 , and \mathbb{R}^m , where \mathbb{R} is the set of *real numbers*. \mathbb{R}^2 , the xy -plane, contains (has as elements) all pairs (v_1, v_2) of real numbers, for example $(4, 1)$. \mathbb{R}^3 , the xyz -space, contains all triples (v_1, v_2, v_3) of real numbers, for example $(-2, 2, 3)$. \mathbb{R}^m contains all m -tuples or *sequences* (v_1, v_2, \dots, v_m) of m real numbers. Here, each number from 1 to m serves as an *index*, indicating the position in the sequence: for example, v_5 denotes the 5-th element of the sequence.

Upfront, a sequence is just a mathematical object that contains some numbers that are ordered; these numbers could mean anything. For example, the pair $(94, 189) \in \mathbb{R}^2$ could stand for the weight (in kg) and height (in cm) of a person. We call an element of \mathbb{R}^2 a *vector* when we think of the numbers in the pair as coordinates in 2-dimensional space; we typically draw such vectors as arrows in a *Cartesian coordinate system*, with the tail of the arrow at the *origin* (where the coordinate axes cross), and the head at the respective coordinates. The zero vector would have both head and tail at the origin, with undefined arrow direction, so we cannot nicely draw it as an arrow but think of it as a point at the origin. For vectors, it is customary to use *column vector* notation to indicate that we now think of a sequence as a vector; see Figure 1.1.

In referring to a vector in text or in a formula, we use bold lower case Latin letters such as \mathbf{v} and \mathbf{w} . You may have learned to write vectors as \vec{v} , \vec{w} , but \mathbf{v} is as good (or bad) as \vec{v} . The important thing is to be consistent.

Definition 1.1 (Vector). Let $m \geq 0$ be a natural number. An m -dimensional coordinate vector (simply called vector in the following) is an element of \mathbb{R}^m , written in column vector notation

¹For mathematicians, 1 is typically the first natural number; for computer scientists it is 0. None of the two choices is better or worse than the other one. In these “linear algebra for computer scientists” notes, we start with 0.

as

$$\mathbf{v} = \begin{pmatrix} v_1 \\ v_2 \\ \vdots \\ v_m \end{pmatrix}. \text{ The vector } \begin{pmatrix} 0 \\ 0 \\ \vdots \\ 0 \end{pmatrix} \text{ is the zero vector in } \mathbb{R}^m, \text{ denoted by } \mathbf{0}.$$

The numbers v_1, v_2, \dots, v_m are the coordinates of \mathbf{v} . In \mathbb{R}^0 , there is only one vector, namely the empty sequence of real numbers, and this is simply written as $()$.

While examples are very useful to understand a concept, a *definition* is the official reference. The goal of a definition is to introduce a concept in full generality, and to be able to refer back to it later.

The notation $\mathbf{0}$ is actually imprecise and represents what mathematicians call an *abuse of notation*. The abuse here is that the meaning of $\mathbf{0}$ depends on the context and may, as in Figure 1.1, refer to the zero vector in \mathbb{R}^2 or the zero vector in \mathbb{R}^3 . Such abuse of notation is common and also known from natural language. If you take “the bike”, it is clear that you mean your bike, while your friends, saying the same thing, refer to their bikes. A fully precise alternative would be to write $\mathbf{0}_2$ for the zero vector in \mathbb{R}^2 , $\mathbf{0}_3$ for the zero vector in \mathbb{R}^3 , and $\mathbf{0}_m$ for the zero vector in \mathbb{R}^m . This usually has not much of a benefit and instead makes the text less readable— $\mathbf{0}_2$ for example looks more like oxygen than a vector.

The arrow drawing suggests an interpretation of a vector as a movement, for example “go 4 steps right and 1 step up!” in case of the vector

$$\begin{pmatrix} 4 \\ 1 \end{pmatrix}.$$

Under this interpretation, the arrow can actually be placed anywhere and still visualizes the same vector. It can also be useful to visualize not only the zero vector but any vector as a point (at its coordinates); see Figure 1.2.

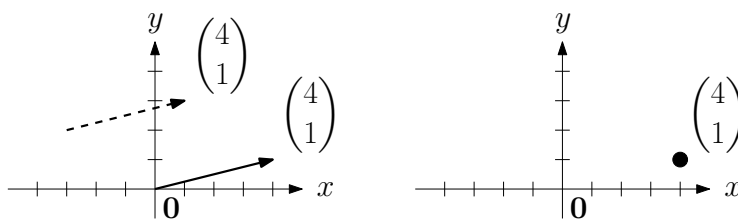


Figure 1.2: Vector: visualization as arrow (left), or point (right)

As an example why this is useful, let us try to visualize the set of vectors in \mathbb{R}^2 whose coordinates sum up to 5, highlighting a specific such vector. Then it should be clear that Figure 1.3 (right), with the gray line showing the vectors in question as points, is more suitable than Figure 1.3 (left) that is just cluttered with arrows.

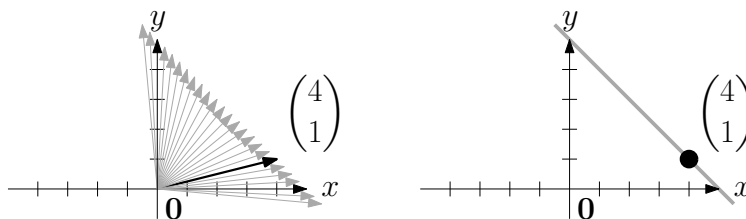


Figure 1.3: Vectors \mathbf{v} in \mathbb{R}^2 with $v_1 + v_2 = 5$: visualization as arrows (left), or points (right)

1.1.1 Vector addition

Adding two vectors combines their movements. Depending on which movement we do first, we can take two different routes to the same result; together, these routes form a *parallelogram*. Algebraically, *vector addition* works *coordinate-wise*, i.e. we add up corresponding coordinates of the two vectors; see Figure 1.4.

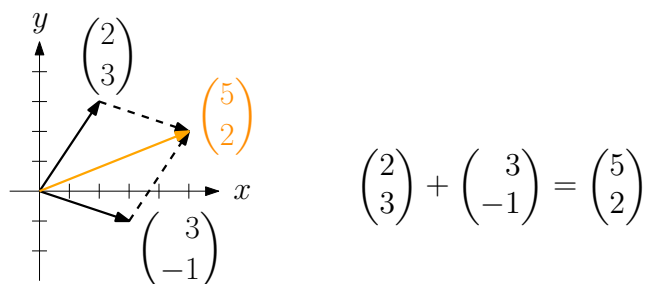


Figure 1.4: The parallelogram of vector addition

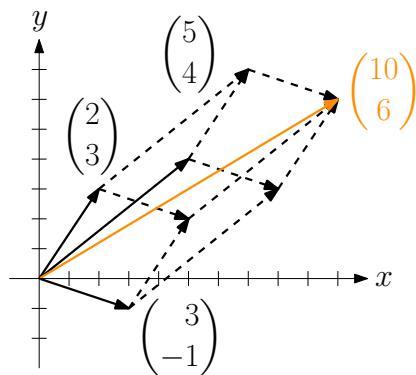
Definition 1.2 (Vector addition). *Let*

$$\mathbf{v} = \begin{pmatrix} v_1 \\ v_2 \\ \vdots \\ v_m \end{pmatrix}, \mathbf{w} = \begin{pmatrix} w_1 \\ w_2 \\ \vdots \\ w_m \end{pmatrix} \in \mathbb{R}^m. \text{ The vector } \mathbf{v} + \mathbf{w} := \begin{pmatrix} v_1 + w_1 \\ v_2 + w_2 \\ \vdots \\ v_m + w_m \end{pmatrix} \in \mathbb{R}^m \text{ is the sum of } \mathbf{v} \text{ and } \mathbf{w}.$$

Definition 1.2 tells you how to add *any* two vectors of *any* dimension. The symbol $:=$ is used to define what is left of it by what is right of it.

In comparison to Definition 1.1, we have omitted the first sentence: *Let $m \geq 0$ be a natural number*. This omission is not formally correct, but acceptable; the symbols m (and also n later) will *always* stand for natural numbers, so there is no need to explicitly say this each time.

In the same way, we can add more vectors. For example $\mathbf{u} + \mathbf{v} + \mathbf{w}$ is the sum of three vectors, and in this case, we get six possible routes to the result; see Figure 1.5.



$$\begin{pmatrix} 2 \\ 3 \end{pmatrix} + \begin{pmatrix} 3 \\ -1 \end{pmatrix} + \begin{pmatrix} 5 \\ 4 \end{pmatrix} = \begin{pmatrix} 10 \\ 6 \end{pmatrix}$$

Figure 1.5: Adding three vectors

Strictly speaking, Definition 1.2 does not define $\mathbf{u} + \mathbf{v} + \mathbf{w}$. It only talks about adding *two* vectors at a time. The natural official definition of $\mathbf{u} + \mathbf{v} + \mathbf{w}$ would be $(\mathbf{u} + \mathbf{v}) + \mathbf{w}$ (add vectors one by one, from left to right), but since addition in \mathbb{R} is *associative*, it does not matter where we put the brackets, so we can as well omit them and write $\mathbf{u} + \mathbf{v} + \mathbf{w}$. Similarly, in talking about the two possible routes to arrive at $\mathbf{v} + \mathbf{w}$, we implicitly used that $\mathbf{v} + \mathbf{w} = \mathbf{w} + \mathbf{v}$ which holds since addition in \mathbb{R} is *commutative*. The six possible routes for three vectors come from the fact that we can add them up in six different orders.

It is an element of mathematical thinking to clarify the meaning of $\mathbf{u} + \mathbf{v} + \mathbf{w}$ when this was not explicitly defined. Only after that, you *really* understand the meaning and can safely write $\mathbf{u} + \mathbf{v} + \mathbf{w}$. As a Swiss luxury watch commercial puts it: to break the rules, you must first master them.

1.1.2 Scalar multiplication

Scalar multiplication corresponds to moving λ times as far, for some real number λ (in this context also called a *scalar*). For scalars, we typically use lower case Greek letters.

We say that the resulting vector is a *scalar multiple* of the original one. The scalar can be negative, in which case we move into the opposite direction. Algebraically, *scalar multiplication* multiplies each coordinate of the vector with the scalar; see Figure 1.6.

Definition 1.3 (Scalar multiplication). *Let*

$$\mathbf{v} = \begin{pmatrix} v_1 \\ v_2 \\ \vdots \\ v_m \end{pmatrix} \in \mathbb{R}^m, \lambda \in \mathbb{R}. \text{ The vector } \lambda \mathbf{v} := \begin{pmatrix} \lambda v_1 \\ \lambda v_2 \\ \vdots \\ \lambda v_m \end{pmatrix} \in \mathbb{R}^m \text{ is a scalar multiple of } \mathbf{v}.$$

Note that the zero vector $\mathbf{0}$ is a scalar multiple of every vector, obtained by choosing scalar $\lambda = 0$.

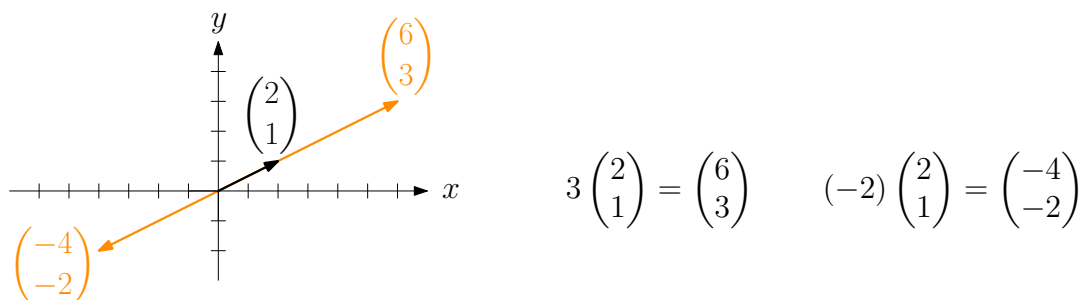


Figure 1.6: Scalar multiplication

1.1.3 Linear combinations

This operation combines vector addition and scalar multiplication.

Definition 1.4 (Linear combination). Let $\mathbf{v}, \mathbf{w} \in \mathbb{R}^m$, $\lambda, \mu \in \mathbb{R}$. The vector

$$\lambda \mathbf{v} + \mu \mathbf{w} \in \mathbb{R}^m$$

is a linear combination of \mathbf{v} and \mathbf{w} . In general, if $\mathbf{v}_1, \mathbf{v}_2, \dots, \mathbf{v}_n \in \mathbb{R}^m$ and $\lambda_1, \lambda_2, \dots, \lambda_n \in \mathbb{R}$, then

$$\lambda_1 \mathbf{v}_1 + \lambda_2 \mathbf{v}_2 + \dots + \lambda_n \mathbf{v}_n$$

is a linear combination of $\mathbf{v}_1, \mathbf{v}_2, \dots, \mathbf{v}_n$.

$\mathbf{v} = \begin{pmatrix} 2 \\ 3 \end{pmatrix}, \mathbf{w} = \begin{pmatrix} 3 \\ -1 \end{pmatrix} :$

λ	μ	$\lambda \mathbf{v}$	$\mu \mathbf{w}$	$\lambda \mathbf{v} + \mu \mathbf{w}$
-3	2	$\begin{pmatrix} -6 \\ -9 \end{pmatrix}$	$\begin{pmatrix} 6 \\ -2 \end{pmatrix}$	$\begin{pmatrix} 0 \\ -11 \end{pmatrix}$
1	-1	$\begin{pmatrix} 2 \\ 3 \end{pmatrix}$	$\begin{pmatrix} -3 \\ 1 \end{pmatrix}$	$\begin{pmatrix} -1 \\ 4 \end{pmatrix}$
3	0	$\begin{pmatrix} 6 \\ 9 \end{pmatrix}$	$\begin{pmatrix} 0 \\ 0 \end{pmatrix}$	$\begin{pmatrix} 6 \\ 9 \end{pmatrix}$

Table 1.1: Three linear combinations of two vectors \mathbf{v}, \mathbf{w}

Table 1.1 gives three linear combinations of two specific vectors \mathbf{v} and \mathbf{w} . What are all the linear combinations of \mathbf{v} and \mathbf{w} that can we get in this way? Here is the answer:

Fact 1.5. Every vector in \mathbb{R}^2 is a linear combination of the two vectors

$$\mathbf{v} = \begin{pmatrix} 2 \\ 3 \end{pmatrix}, \mathbf{w} = \begin{pmatrix} 3 \\ -1 \end{pmatrix}.$$

Proof. Let

$$\mathbf{u} = \begin{pmatrix} u_1 \\ u_2 \end{pmatrix}$$

be an arbitrary vector in \mathbb{R}^2 . We need to show that we can find scalars $\lambda, \mu \in \mathbb{R}$ such that

$$\lambda \begin{pmatrix} 2 \\ 3 \end{pmatrix} + \mu \begin{pmatrix} 3 \\ -1 \end{pmatrix} = \begin{pmatrix} u_1 \\ u_2 \end{pmatrix}.$$

Considering the rules of vector addition and scalar multiplication from Definitions 1.2 and 1.3, this vector equation is actually made up of two “normal” equations, one for each coordinate:

$$\begin{aligned} 2\lambda + 3\mu &= u_1, \\ 3\lambda - 1\mu &= u_2. \end{aligned}$$

This is a system of two linear equations in two variables λ and μ , and the rest is therefore a high school job.

Well, almost: What you may find unusual here is that the system also contains the variables u_1, u_2 . But while the variables λ and μ stand for the unknown numbers that we want to compute, u_1 and u_2 stand for known numbers, the coordinates of our target vector \mathbf{u} . By solving this system of equations (the solution will depend on u_1 and u_2), we therefore solve it for all possible values of u_1 and u_2 at the same time. And this was exactly the point, since we want to make the argument for every possible vector \mathbf{u} . This is what differentiates a proof from a calculation: a proof makes one argument for many (possibly infinitely many) situations, while a calculation just handles one situation.

After this digression, let us solve the system: Multiplying the second equation by 3 and adding it to the first one cancels the variable μ :

$$\begin{array}{rcl} 2\lambda & + & 3\mu = u_1 \\ 9\lambda & - & 3\mu = 3u_2 \\ \hline 11\lambda & & = u_1 + 3u_2 \end{array}$$

This gives

$$\lambda = \frac{u_1 + 3u_2}{11}.$$

To get μ , we isolate μ in one of the equations (let us take the first one) and substitute the value of λ that we just got:

$$3\mu = u_1 - 2\lambda = u_1 - \frac{2u_1 + 6u_2}{11} = \frac{11u_1 - (2u_1 + 6u_2)}{11} = \frac{9u_1 - 6u_2}{11}.$$

Dividing by 3 yields

$$\mu = \frac{3u_1 - 2u_2}{11}.$$

So λ and μ have been found for all possible values of u_1 and u_2 which completes the proof. \square

It is always good to double check this on examples. Let us look at the three linear combinations that we have previously computed in Table 1.1 and see whether the formulas for λ and μ give us back the correct scalars. Table 1.2 shows that they do.

$$\mathbf{v} = \begin{pmatrix} 2 \\ 3 \end{pmatrix}, \mathbf{w} = \begin{pmatrix} 3 \\ -1 \end{pmatrix} :$$

λ	μ	$\lambda\mathbf{v} + \mu\mathbf{w} = \mathbf{u}$	u_1	u_2	$\frac{u_1+3u_2}{11}$	$\frac{3u_1-2u_2}{11}$
-3	2	$\begin{pmatrix} 0 \\ -11 \end{pmatrix}$	0	-11	-3	2
1	-1	$\begin{pmatrix} -1 \\ 4 \end{pmatrix}$	-1	4	1	-1
3	0	$\begin{pmatrix} 6 \\ 9 \end{pmatrix}$	6	9	3	0

Table 1.2: Checking the formulas from the proof of Fact 1.5 on three vectors

We can also understand this proof geometrically. The two equations $2\lambda + 3\mu = u_1$ and $3\lambda - 1\mu = u_2$ can be drawn as lines in the $\lambda\mu$ -plane, and their point of intersection is the desired solution to both equations. For $u_1 = -1, u_2 = 4$ (middle row of Table 1.2), the two lines are drawn in Figure 1.7 (left). Unsurprisingly, their intersection is at $(\lambda, \mu) = (1, -1)$.

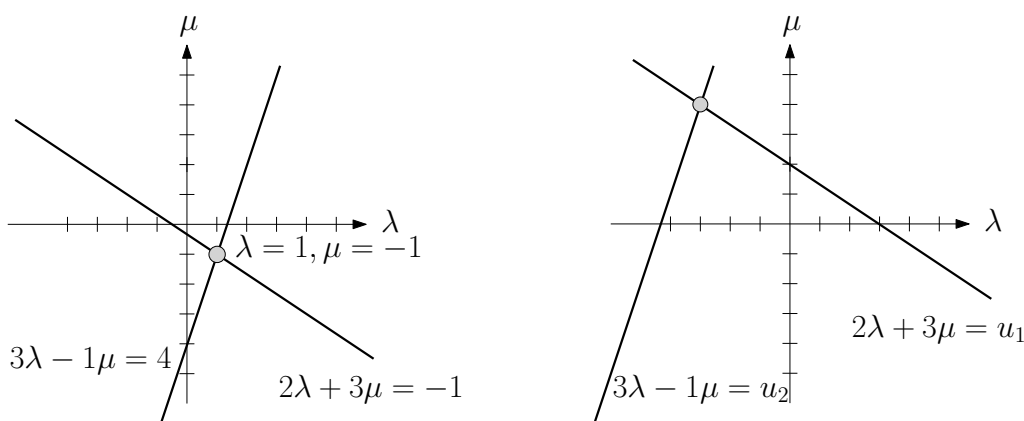


Figure 1.7: “Row picture” behind the proof of Fact 1.5, based on the rows of the equation system. We find the target scalars by intersecting two lines in the $\lambda\mu$ -plane.

The crucial observation is that for other values of u_1, u_2 , the lines are different but still parallel to the lines for $u_1 = -1, u_2 = 4$; see Figure 1.7 (right). So there is always an intersection, meaning that we find values λ, μ for every vector \mathbf{u} . This was the “row picture” behind the proof.

There is also a “column picture”, see Figure 1.8. The two vectors \mathbf{v} and \mathbf{w} define axes of a skewed coordinate system (solid lines). Given a target vector \mathbf{u} , we make shifted copies of both axes such that they cross in \mathbf{u} (dashed lines). Between the two pairs of

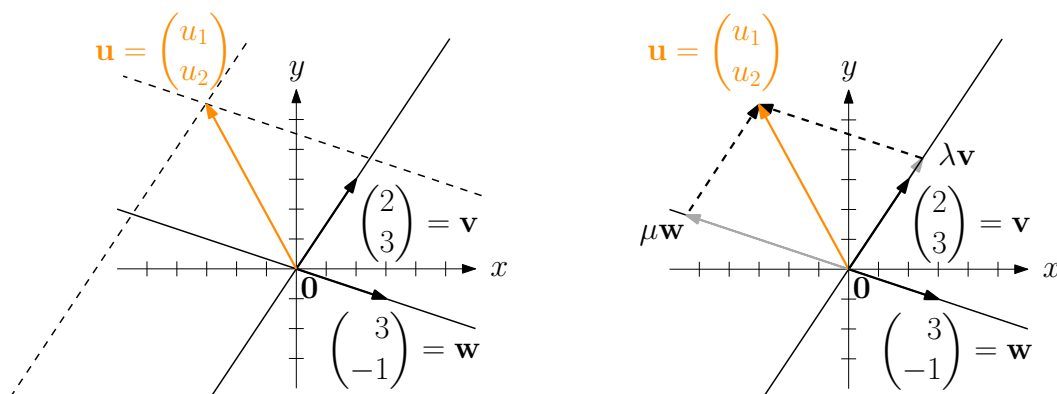


Figure 1.8: “Column picture” behind the proof of Fact 1.5, based on the column vectors of the equation system: We find the target scalars as coordinates in a skewed coordinate system.

axes, a parallelogram emerges, and this is exactly the one that expresses \mathbf{u} as a sum of scaled versions of \mathbf{v} and \mathbf{w} , the gray arrows in Figure 1.8 (right).

Fact 1.5 may be wrong for other vectors. As an example, consider

$$\mathbf{v} = \begin{pmatrix} 2 \\ 3 \end{pmatrix}, \mathbf{w} = \begin{pmatrix} 4 \\ 6 \end{pmatrix}.$$

Here, \mathbf{w} is a scalar multiple of \mathbf{v} , so every linear combination of the two is just another scalar multiple of \mathbf{v} . Therefore, the linear combinations form a line through \mathbf{v} , and a vector not on that line cannot be obtained as a linear combination.

Challenge 1.6. Try to prove a version of Fact 1.5 where

$$\mathbf{v} = \begin{pmatrix} v_1 \\ v_2 \end{pmatrix}, \mathbf{w} = \begin{pmatrix} w_1 \\ w_2 \end{pmatrix}$$

are arbitrary vectors. Where does the proof fail if the two vectors are scalar multiples of each other? What goes wrong in the row and column pictures in this case?

1.1.4 Affine, conic, and convex combinations

In many applications, we are not interested in all linear combinations of the given vectors. The following three types of special linear combinations are of particular importance.

Definition 1.7 (Affine, conic, convex combination). A linear combination $\lambda_1 \mathbf{v}_1 + \lambda_2 \mathbf{v}_2 + \cdots + \lambda_n \mathbf{v}_n$ of vectors $\mathbf{v}_1, \mathbf{v}_2, \dots, \mathbf{v}_n$ is called

- (i) an affine combination if $\lambda_1 + \lambda_2 + \cdots + \lambda_n = 1$,
- (ii) a conic combination if $\lambda_j \geq 0$ for $j = 1, 2, \dots, n$, and
- (iii) a convex combination if it is both an affine and a conic combination.

We will illustrate these notions with linear combinations $\lambda \mathbf{v} + \mu \mathbf{w}$ of two vectors $\mathbf{v}, \mathbf{w} \in \mathbb{R}^2$ that are not scalar multiples of each other. By Challenge 1.6, the set of linear combinations of such vectors is the whole plane \mathbb{R}^2 ; see Figure 1.9 (i). What is the set of affine combinations? If $\lambda + \mu = 1$, we can write

$$\lambda \mathbf{v} + \mu \mathbf{w} = \lambda \mathbf{v} + \underbrace{(1 - \lambda)}_{\mu} \mathbf{w} = \mathbf{w} + \lambda(\mathbf{v} - \mathbf{w}).$$

Hence, in *set builder notation* (see also Section 1.1.5 below) the set of all affine combinations is the set

$$\{\mathbf{w} + \lambda(\mathbf{v} - \mathbf{w}) : \lambda \in \mathbb{R}\},$$

the set of all vectors of the form $\mathbf{w} + \lambda(\mathbf{v} - \mathbf{w})$ where λ is a real number.

This is the line through \mathbf{v} and \mathbf{w} (interpreted as points; see Figure 1.3). Plugging in $\lambda = 0$ gives \mathbf{w} , and for $\lambda = 1$, we obtain \mathbf{v} . Values of λ between 0 and 1 lead to points in between. For $\lambda < 0$ we end up left of \mathbf{w} , and for $\lambda > 1$, we will be right of \mathbf{v} . Generally, any point on the line is obtained by the rule “go to \mathbf{w} , then add a scalar multiple of $\mathbf{v} - \mathbf{w}$!”. See Figure 1.9 (ii).

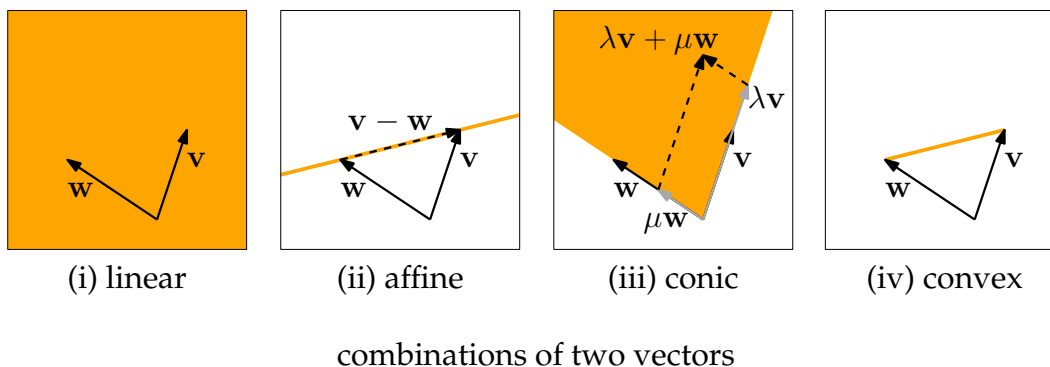
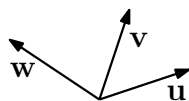


Figure 1.9: Two vectors in \mathbb{R}^2 and their linear, affine, conic, and convex combinations

In a conic combination, the parallelogram for adding $\lambda \mathbf{v}$ and $\mu \mathbf{w}$ is always in the angle between \mathbf{v} and \mathbf{w} , and by varying λ and μ , we can get every vector in this angle, officially called the *cone* spanned by \mathbf{v} and \mathbf{w} ; see Figure 1.9 (iii). Finally, the convex combinations are by definition all points on the line through \mathbf{v} and \mathbf{w} that are also in the cone spanned by \mathbf{v} and \mathbf{w} . This set is the *line segment* spanned by \mathbf{v} and \mathbf{w} , see Figure 1.9 (iv).

Challenge 1.8. Understand the affine, conic and convex combinations of three vectors in \mathbb{R}^2 !



1.1.5 Defining the dots: sequences, sums, sets, and vectors

The dot notations as used in

$$\mathbf{v}_1, \mathbf{v}_2, \dots, \mathbf{v}_n \quad \text{and} \quad \lambda_1 \mathbf{v}_1 + \lambda_2 \mathbf{v}_2 + \dots + \lambda_n \mathbf{v}_n \quad \text{and} \quad \begin{pmatrix} v_1 \\ v_2 \\ \vdots \\ v_m \end{pmatrix}$$

are fine, but it is never too early to get to know the precise mathematical notations. This will also help in really understanding the dot notations. Formally, when we write $\mathbf{v}_1, \mathbf{v}_2, \dots, \mathbf{v}_n$, we mean the sequence $(\mathbf{v}_1, \mathbf{v}_2, \dots, \mathbf{v}_n)$ of n vectors; we typically omit the surrounding brackets if we do not need the sequence itself as a mathematical object. There is a more concise way of writing the sequence:

$$(\mathbf{v}_1, \mathbf{v}_2, \dots, \mathbf{v}_n) = (\mathbf{v}_j)_{j=1}^n.$$

Similarly, sums have a mathematical notation:

$$\lambda_1 \mathbf{v}_1 + \lambda_2 \mathbf{v}_2 + \dots + \lambda_n \mathbf{v}_n = \sum_{j=1}^n \lambda_j \mathbf{v}_j.$$

Here, j is called the *summation index*.

The benefit of this notation becomes apparent when we discuss some special cases. What if $n = 2$? It is not entirely clear how to interpret $\mathbf{v}_1, \mathbf{v}_2, \dots, \mathbf{v}_n$ in this case. Does \mathbf{v}_2 now appear twice? And if $n = 1$? Then there is not even a vector \mathbf{v}_2 , so why is it mentioned?

In starting with $\mathbf{v}_1, \mathbf{v}_2, \dots$, we want to indicate a pattern that continues until \mathbf{v}_n , and we therefore mean $(\mathbf{v}_1, \mathbf{v}_2)$ if $n = 2$ and (\mathbf{v}_1) if $n = 1$. Still, it is potentially confusing to always mention three vectors $\mathbf{v}_1, \mathbf{v}_2$ and \mathbf{v}_n , even if there are less. In contrast, the notation

$$(\mathbf{v}_j)_{j=1}^n$$

has a clear definition: it is the sequence of all vectors \mathbf{v}_j where j goes through the range from 1 to n in increasing order. This range contains the integers j that satisfy $1 \leq j \leq n$. This naturally covers the cases $n = 1$ and $n = 2$. For the sum, it is the same:

$$\sum_{j=1}^n \lambda_j \mathbf{v}_j$$

is obtained by summing up all $\lambda_j \mathbf{v}_j$ for which j goes through the range from 1 to n .

This brings us to the last special case: what if $n = 0$, so there are no vectors? The mathematical notation handles this elegantly: as the range of integers from 1 to 0 is empty, we have

$$(\mathbf{v}_j)_{j=1}^0 = (),$$

the empty sequence of vectors. And

$$\sum_{j=1}^0 \lambda_j \mathbf{v}_j = \mathbf{0},$$

the zero vector. Indeed, if you add up vectors, you start from $\mathbf{0}$ and then add one vector at a time. But if there are no vectors to add, you remain stuck at $\mathbf{0}$. This is like a scale that shows a weight of 0 before you put anything on it. As a consequence, $\mathbf{0}$ is a linear combination of *every* sequence of vectors, even the empty one.

Having made it clear that for $n = 0$, $\mathbf{v}_1, \mathbf{v}_2, \dots, \mathbf{v}_n$ is the empty sequence and $\lambda_1 \mathbf{v}_1 + \lambda_2 \mathbf{v}_2 + \dots + \lambda_n \mathbf{v}_n = \mathbf{0}$, there is also no harm if we keep using the dot notations. For novices, they may be easier to read than $(\mathbf{v}_j)_{j=1}^n$ and $\sum_{j=1}^n \lambda_j \mathbf{v}_j$. But as in particular the sum notation is standard throughout many sources, it is also good to get used to it early.

When we reorder a sequence of vectors, we get the same linear combinations, because vector addition is commutative. Also, if a vector appears more than once in the sequence, we can omit its duplicates and still get the same linear combinations. This means, all that matters is the *set* of vectors involved in the sequence $(\mathbf{v}_j)_{j=1}^n$. We will write this set as

$$\{\mathbf{v}_j : j \in [n]\}.$$

Here, $[n]$ is an unambiguous notation for the the range from 1 to n , considered as a set.

What we see in action here is the *set builder notation*. The notation $\{\mathbf{v}_j : j \in [n]\}$ means: the set of all \mathbf{v}_j for which the condition $j \in [n]$ behind the colon is true. So we can read the colon as *for which*. In some sources, this is written as $\{\mathbf{v}_j \mid j \in [n]\}$, but the meaning is the same. Here is another example. In set builder notation, we could define $[n]$ as

$$[n] := \{i : i \in \mathbb{N}, 1 \leq i \leq n\},$$

where a comma is to be read as *and*.

Within a set, the order does not matter, so we prefer the notation $[n]$ over $\{1, 2, \dots, n\}$. But the latter notation and also $\{\mathbf{v}_1, \mathbf{v}_2, \dots, \mathbf{v}_n\}$ for the set of vectors are perfectly fine as well. The important thing is that there is no order of vectors in the set, and every vector only appears once. We can write $\{\mathbf{v}_1, \mathbf{v}_1, \mathbf{v}_2\}$, but this is the same as $\{\mathbf{v}_1, \mathbf{v}_2\}$ or $\{\mathbf{v}_2, \mathbf{v}_1\}$.

Finally, the vertical dots: a vector

$$\begin{pmatrix} v_1 \\ v_2 \\ \vdots \\ v_m \end{pmatrix}$$

can more concisely be written as $(v_i)_{i=1}^m$. Indeed, a vector as an element of \mathbb{R}^m is formally just a sequence of m real numbers. This notation for example allows us to write down the vector $(i^2)_{i=1}^5$. What does this mean? The expression in brackets tells us what the i -th

coordinate of the vector should be, and the range after the bracket tells us which range of i 's we are considering. Hence,

$$(i^2)_{i=1}^5 = \begin{pmatrix} 1 \\ 4 \\ 9 \\ 16 \\ 25 \end{pmatrix}.$$

Similarly, $(0)_{i=1}^6$ is the 6-dimensional zero vector. Using this notation, we could also define the sum of two vectors more “efficiently” than Definition 1.2 does it, namely as follows:

$$(v_i)_{i=1}^m + (w_i)_{i=1}^m := (v_i + w_i)_{i=1}^m.$$

We can still use the vertical dots notation as it is more readable, despite needing more space. But now that we know what it precisely means, we for example understand that for $m = 1$,

$$\begin{pmatrix} v_1 \\ v_2 \\ \vdots \\ v_m \end{pmatrix} = (v_i)_{i=1}^m = (v_1) \in \mathbb{R}^1.$$

What about the case $m = 0$? As pointed out in Definition 1.1, \mathbb{R}^0 contains exactly one element, namely the empty sequence $()$ of real numbers. So $(v_i)_{i=1}^0 = () \in \mathbb{R}^0$.

1.2 Scalar products, lengths and angles

The scalar product of two vectors is a number that helps us in defining the length of a vector and the angle between two vectors. Scalar products naturally appear all over linear algebra and have many applications. Scalar products are also at the heart of two important inequalities, the Cauchy-Schwarz inequality, and the triangle inequality. We also introduce and explain the standard notation $\mathbf{v}^\top \mathbf{w}$ for the scalar product.

Given how vector addition works (Definition 1.2), you might expect vector multiplication to work like this:

$$\begin{pmatrix} 1 \\ 2 \end{pmatrix} \cdot \begin{pmatrix} 3 \\ 4 \end{pmatrix} = \begin{pmatrix} 1 \cdot 3 \\ 2 \cdot 4 \end{pmatrix} = \begin{pmatrix} 3 \\ 8 \end{pmatrix}.$$

This is known as the *Hadamard product*, but in the range of linear algebra products, it only occupies a niche. Among the topsellers, we find the scalar product whose result is not a vector, but a number (or a scalar, in linear algebra jargon; this is where the name comes from).

1.2.1 Scalar product

The *scalar product* of two vectors is obtained by multiplying corresponding coordinates, and adding up the products:

$$\begin{pmatrix} 1 \\ 2 \end{pmatrix} \cdot \begin{pmatrix} 3 \\ 4 \end{pmatrix} = 1 \cdot 3 + 2 \cdot 4 = 11.$$

Definition 1.9 (Scalar product). *Let*

$$\mathbf{v} = \begin{pmatrix} v_1 \\ v_2 \\ \vdots \\ v_m \end{pmatrix}, \mathbf{w} = \begin{pmatrix} w_1 \\ w_2 \\ \vdots \\ w_m \end{pmatrix} \in \mathbb{R}^m.$$

The scalar product of \mathbf{v} and \mathbf{w} is the number

$$\mathbf{v} \cdot \mathbf{w} := v_1 w_1 + v_2 w_2 + \cdots + v_m w_m = \sum_{i=1}^m v_i w_i.$$

Note that for $\mathbf{v}, \mathbf{w} \in \mathbb{R}^0$, we get $\mathbf{v} \cdot \mathbf{w} = 0$. For $m = 1$, scalar multiplication behaves like the normal multiplication: $(v_1) \cdot (w_1) = v_1 w_1$.

From this definition, we can directly derive some rules that are frequently needed in computing with scalar products. We summarize these in an *observation*. This is a statement whose proof is simple and straightforward enough to be omitted.

Observation 1.10. *Let $\mathbf{u}, \mathbf{v}, \mathbf{w} \in \mathbb{R}^m$ be vectors and $\lambda \in \mathbb{R}$ a scalar. Then*

- (i) $\mathbf{v} \cdot \mathbf{w} = \mathbf{w} \cdot \mathbf{v}$; (symmetry)
- (ii) $(\lambda \mathbf{v}) \cdot \mathbf{w} = \lambda(\mathbf{v} \cdot \mathbf{w}) = \mathbf{v} \cdot (\lambda \mathbf{w})$; (taking out scalars)
- (iii) $\mathbf{u} \cdot (\mathbf{v} + \mathbf{w}) = \mathbf{u} \cdot \mathbf{v} + \mathbf{u} \cdot \mathbf{w}$ and $(\mathbf{u} + \mathbf{v}) \cdot \mathbf{w} = \mathbf{u} \cdot \mathbf{w} + \mathbf{v} \cdot \mathbf{w}$; (distributivity)
- (iv) $\mathbf{v} \cdot \mathbf{v} \geq 0$, with equality exactly if $\mathbf{v} = \mathbf{0}$. (positive-definiteness)

While (i) and (iv) are quite obvious from the definition of the scalar product, (ii) and (iii) need some straightforward calculations as a proof. To show what we mean by “straightforward”, we provide the calculations for the first part of (iii). These use vector addition (Definition 1.2) and distributivity in \mathbb{R} :

$$\mathbf{u} \cdot (\mathbf{v} + \mathbf{w}) = \sum_{i=1}^m u_i(v_i + w_i) = \sum_{i=1}^m (u_i v_i + u_i w_i) = \sum_{i=1}^m u_i v_i + \sum_{i=1}^m u_i w_i = \mathbf{u} \cdot \mathbf{v} + \mathbf{u} \cdot \mathbf{w}.$$

Properties (ii) and (iii) together are known as *linearity in both arguments*.

1.2.2 Euclidean norm

The *Euclidean norm* of a vector \mathbf{v} is obtained by taking the square root of the scalar product with itself. We can do this, since this scalar product is nonnegative by Observation 1.10 (iv).

Definition 1.11 (Euclidean norm). Let $\mathbf{v} \in \mathbb{R}^m$. The Euclidean norm of \mathbf{v} is the number

$$\|\mathbf{v}\| := \sqrt{\mathbf{v} \cdot \mathbf{v}}.$$

Some sources use $|\mathbf{v}|$ for the norm, but we reserve this notation for the *absolute value* of a number and the *size of a set*; for example, $|3| = |-3| = 3$ and $|\{2, 3, 5, 7\}| = 4$.

You can think of the Euclidean norm as defining the length of a vector. You may argue that we do not need to define this, since a vector already has a length (just measure how long the arrow is). But this is true only in \mathbb{R}^2 and \mathbb{R}^3 where we can draw vectors as arrows. In higher dimensions, it is not a priori clear how to measure the length of a vector \mathbf{v} . But now we know: compute its Euclidean norm $\|\mathbf{v}\|$!

In \mathbb{R}^2 and \mathbb{R}^3 , the Euclidean norm indeed measures the length of the arrow; so we are not really inventing a new concept of length here, we simply extend a familiar concept to higher dimensions. To see this, we first expand the scalar product to obtain the following formula for the Euclidean norm:

$$\left\| \begin{pmatrix} v_1 \\ v_2 \\ \vdots \\ v_m \end{pmatrix} \right\| = \sqrt{v_1^2 + v_2^2 + \cdots + v_m^2} = \sqrt{\sum_{i=1}^m v_i^2}.$$

For example,

$$\left\| \begin{pmatrix} -4 \\ 2 \end{pmatrix} \right\| = \sqrt{(-4)^2 + 2^2} = \sqrt{20}, \quad \left\| \begin{pmatrix} -4 \\ 2 \\ 3 \end{pmatrix} \right\| = \sqrt{(-4)^2 + 2^2 + 3^2} = \sqrt{29}.$$

Let us first look at the situation in \mathbb{R}^2 where Figure 1.10 is the key.

The vector

$$\mathbf{v} = \begin{pmatrix} v_1 \\ v_2 \end{pmatrix}$$

is the hypotenuse of a right-angled triangle whose legs have lengths $|v_1|$ and $|v_2|$, see Figure 1.10. Hence, using the *Pythagorean theorem*, the squared length of \mathbf{v} is

$$|v_1|^2 + |v_2|^2 = v_1^2 + v_2^2 = \|\mathbf{v}\|^2,$$

and “length of \mathbf{v} equals $\|\mathbf{v}\|$ ” is obtained by taking square roots. Building on this, Figure 1.11 deals with the situation in \mathbb{R}^3 .

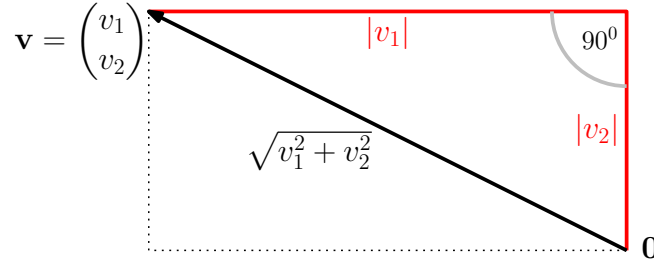


Figure 1.10: The Euclidean norm measures the length of a vector in \mathbb{R}^2 .

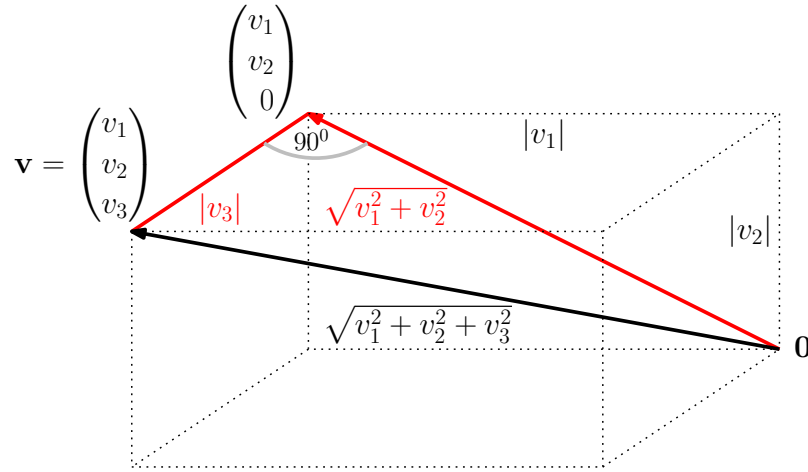


Figure 1.11: The Euclidean norm measures the length of a vector in \mathbb{R}^3 .

In \mathbb{R}^3 , the vector

$$\mathbf{v} = \begin{pmatrix} v_1 \\ v_2 \\ v_3 \end{pmatrix}$$

is the hypotenuse of a right-angled triangle whose legs have lengths $\sqrt{v_1^2 + v_2^2}$ (as just computed) and $|v_3|$; see Figure 1.11. Thus, Pythagoras tells us that the squared length of \mathbf{v} is

$$\sqrt{v_1^2 + v_2^2}^2 + |v_3|^2 = v_1^2 + v_2^2 + v_3^2 = \|\mathbf{v}\|^2.$$

Unit vectors. A *unit vector* is a vector \mathbf{u} such that $\|\mathbf{u}\| = 1$. In \mathbb{R}^2 , the unit vectors lie on the *unit circle* with center $\mathbf{0}$ and radius 1; see Figure 1.12.

Definition 1.11 of the Euclidean norm and taking out scalars (Observation 1.10 (ii)) easily give that for every vector $\mathbf{v} \neq \mathbf{0}$, the vector

$$\frac{\mathbf{v}}{\|\mathbf{v}\|}$$

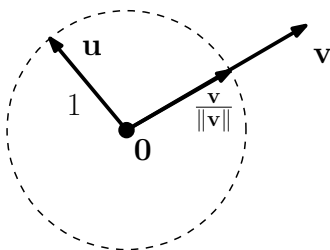


Figure 1.12: A unit vector \mathbf{u} on the unit circle. For every nonzero vector \mathbf{v} , the scaled vector $\mathbf{v}/\|\mathbf{v}\|$ is a unit vector.

is a unit vector, where *scalar division* is just scalar multiplication with the reciprocal:

$$\frac{\mathbf{v}}{\|\mathbf{v}\|} := \frac{1}{\|\mathbf{v}\|} \mathbf{v}.$$

In \mathbb{R}^m , there are m *standard unit vectors*. These are the ones that have one coordinate equal to 1 and all others equal to 0. We use the notation \mathbf{e}_i for the standard unit vector that has the 1 at the i -th coordinate (in abuse of notation, we call this \mathbf{e}_i in every dimension):

$$\mathbb{R}^3 : \mathbf{e}_1 = \begin{pmatrix} 1 \\ 0 \\ 0 \end{pmatrix}, \mathbf{e}_2 = \begin{pmatrix} 0 \\ 1 \\ 0 \end{pmatrix}, \mathbf{e}_3 = \begin{pmatrix} 0 \\ 0 \\ 1 \end{pmatrix} \quad \left| \quad \mathbb{R}^m : \mathbf{e}_i = \begin{pmatrix} 0 \\ \vdots \\ 1 \\ \vdots \\ 0 \end{pmatrix} \leftarrow \text{coordinate } i$$

In \mathbb{R}^2 , the two standard unit vectors are the ones in the directions of x - and y -axis. In \mathbb{R}^3 , \mathbf{e}_3 goes along the z -axis; see Figure 1.13.

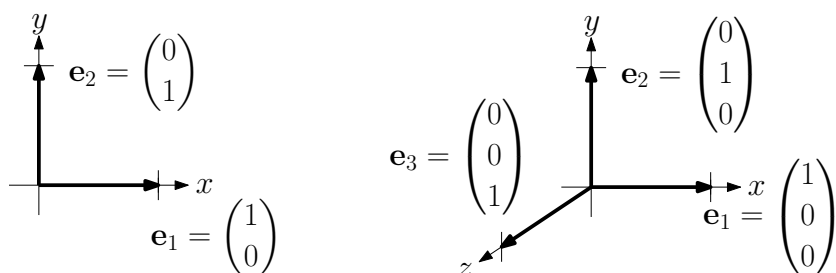


Figure 1.13: The standard unit vectors in \mathbb{R}^2 and \mathbb{R}^3

Other norms. To stress the point that the length of a vector is nothing God-given, we remark that there are many other norms for vectors that make sense and are being used.

Here are two examples, the 1 -norm and the ∞ -norm:

$$\|\mathbf{v}\|_1 := \sum_{i=1}^m |v_i| \quad (1\text{-norm})$$

and

$$\|\mathbf{v}\|_\infty := \max_{i=1}^m |v_i| \quad (\infty\text{-norm}).$$

In these norms, the “unit circles” look different, see Figure 1.14. The vectors in \mathbb{R}^2 that have length 1 according to the 1-norm form a “diamond”; under the ∞ -norm, we get a square. The standard unit vectors are also unit vectors under the 1-norm and the ∞ -norm.

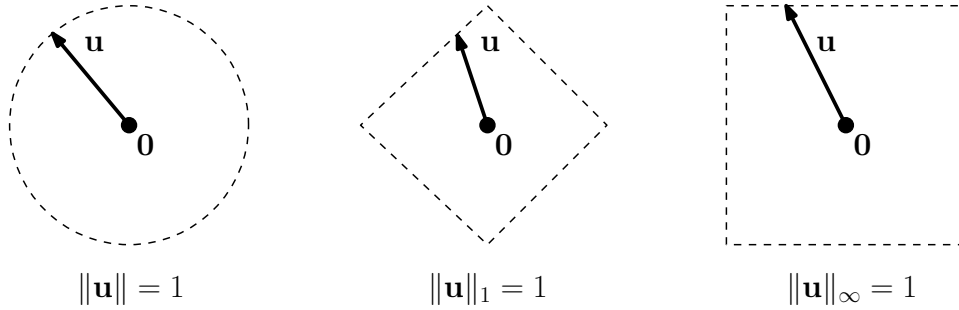


Figure 1.14: Unit vectors in the Euclidean norm (left), 1-norm (middle), ∞ -norm (right)

All three norms are special cases of p -norms, where $p \geq 1$ can be any real number:

$$\|\mathbf{v}\|_p = \sqrt[p]{\sum_{i=1}^m |v_i|^p}.$$

We see that the Euclidean norm is actually the 2-norm, but we still write it as $\|\mathbf{v}\|$, not $\|\mathbf{v}\|_2$, since for us, it is the “standard” norm. But some sources use the notation $\|\mathbf{v}\|_2$. The ∞ -norm is an abuse of notation, since ∞ is not a real number. But as “ p goes to infinity”, the largest coordinate of \mathbf{v} in absolute value is all that matters. In formulas,

$$\lim_{p \rightarrow \infty} \|\mathbf{v}\|_p = \|\mathbf{v}\|_\infty.$$

1.2.3 Cauchy-Schwarz inequality

As innocent as it looks, the importance of this inequality cannot be overestimated.

Lemma 1.12 (Cauchy-Schwarz inequality). *For any two vectors $\mathbf{v}, \mathbf{w} \in \mathbb{R}^m$,*

$$|\mathbf{v} \cdot \mathbf{w}| \leq \|\mathbf{v}\| \|\mathbf{w}\|.$$

Moreover, equality holds exactly if one vector is a scalar multiple of the other.

Generally, a *lemma* is a helper statement that may not be very interesting on its own; but the more it actually helps, the more important it becomes. In this sense, a lemma is like a Swiss army knife, where the Cauchy-Schwarz inequality is a highly multifunctional one.

Proof. We first consider the case where \mathbf{v} and \mathbf{w} are unit vectors, so $\|\mathbf{v}\| = \|\mathbf{w}\| = 1$. Using Observation 1.10 and Definition 1.11 of the Euclidean norm, we compute

$$\begin{aligned} 0 \leq (\mathbf{v} - \mathbf{w}) \cdot (\mathbf{v} - \mathbf{w}) &= \underbrace{\mathbf{v} \cdot \mathbf{v}}_{\|\mathbf{v}\|^2} + \underbrace{\mathbf{w} \cdot \mathbf{w}}_{\|\mathbf{w}\|^2} - 2\mathbf{v} \cdot \mathbf{w} = 2 - 2\mathbf{v} \cdot \mathbf{w} \Rightarrow \mathbf{v} \cdot \mathbf{w} \leq 1, \\ 0 \leq (\mathbf{v} + \mathbf{w}) \cdot (\mathbf{v} + \mathbf{w}) &= \underbrace{\mathbf{v} \cdot \mathbf{v}}_{\|\mathbf{v}\|^2} + \underbrace{\mathbf{w} \cdot \mathbf{w}}_{\|\mathbf{w}\|^2} + 2\mathbf{v} \cdot \mathbf{w} = 2 + 2\mathbf{v} \cdot \mathbf{w} \Rightarrow \mathbf{v} \cdot \mathbf{w} \geq -1. \end{aligned}$$

Summarized as $|\mathbf{v} \cdot \mathbf{w}| \leq 1$, this proves the Cauchy-Schwarz inequality for unit vectors.

Now suppose \mathbf{v} and \mathbf{w} are arbitrary vectors. If one of them is $\mathbf{0}$, the inequality holds (both sides are 0). If $\mathbf{v} \neq \mathbf{0}$ and $\mathbf{w} \neq \mathbf{0}$, we can apply the previous calculations after scaling \mathbf{v} and \mathbf{w} to unit length. This gives

$$-1 \leq \frac{\mathbf{v}}{\|\mathbf{v}\|} \cdot \frac{\mathbf{w}}{\|\mathbf{w}\|} \leq 1,$$

Taking out scalars according to Observation 1.10 (ii) and multiplying all three terms with $\|\mathbf{v}\|\|\mathbf{w}\|$ results in

$$-1 \leq \frac{\mathbf{v} \cdot \mathbf{w}}{\|\mathbf{v}\|\|\mathbf{w}\|} \leq 1 \quad \text{and} \quad -\|\mathbf{v}\|\|\mathbf{w}\| \leq \mathbf{v} \cdot \mathbf{w} \leq \|\mathbf{v}\|\|\mathbf{w}\|.$$

This can be summarized as $|\mathbf{v} \cdot \mathbf{w}| \leq \|\mathbf{v}\|\|\mathbf{w}\|$ which proves the Cauchy-Schwarz inequality in general.

For the “Moreover” part, we need to understand under which conditions equality holds. Going back to the calculations with the unit vectors, we see that the conditions are precisely $0 = (\mathbf{v} - \mathbf{w})(\mathbf{v} - \mathbf{w})$ or $0 = (\mathbf{v} + \mathbf{w})(\mathbf{v} + \mathbf{w})$. By positive-definiteness of the scalar product (Observation 1.10(iv)), this translates to $\mathbf{v} - \mathbf{w} = \mathbf{0}$ or $\mathbf{v} + \mathbf{w} = \mathbf{0}$.

In other words, the two unit vectors are either the same or opposite vectors. For the unit vectors $\mathbf{v}/\|\mathbf{v}\|$ and $\mathbf{w}/\|\mathbf{w}\|$, it is easy to see that this is the case exactly if \mathbf{v} and \mathbf{w} are scalar multiples of each other. If equality is due to one of \mathbf{v} and \mathbf{w} being $\mathbf{0}$, it is still true that one vector (namely $\mathbf{0}$) is a scalar multiple of the other one. \square

Here is an application of the Cauchy-Schwarz inequality. Imagine that you have m squares with total area A , and you put them next to each other as in Figure 1.15. How much horizontal space do you need at most?

To solve this, let v_1, v_2, \dots, v_m be the side lengths of the squares. Then the horizontal space needed is

$$\sum_{i=1}^m v_i.$$

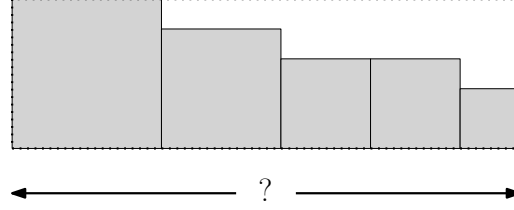


Figure 1.15: Horizontal alignment of m squares with total area A

Let $\mathbf{v} \in \mathbb{R}^m$ be the vector with coordinates v_1, v_2, \dots, v_m . We know that

$$\|\mathbf{v}\|^2 = \sum_{i=1}^m v_i^2 = A.$$

Furthermore, let $\mathbf{1} \in \mathbb{R}^m$ be the vector with all coordinates equal to 1. We have $\|\mathbf{1}\| = \sqrt{m}$. By the Cauchy-Schwarz inequality,

$$\sum_{i=1}^m v_i = \mathbf{1} \cdot \mathbf{v} \leq \|\mathbf{1}\| \|\mathbf{v}\| = \sqrt{m} \sqrt{A},$$

so we need at most \sqrt{mA} horizontal space. If \mathbf{v} is a scalar multiple of $\mathbf{1}$ (meaning that all squares have the same size), we need exactly \sqrt{mA} . Otherwise, we need less.

Exercise 1.13. For the 1-norm and ∞ -norm as defined on page 28, prove that the following inequalities hold for every vector $\mathbf{v} \in \mathbb{R}^m$.

$$\|\mathbf{v}\| \leq \|\mathbf{v}\|_1 \leq \sqrt{m} \|\mathbf{v}\|$$

and

$$\|\mathbf{v}\|_\infty \leq \|\mathbf{v}\| \leq \sqrt{m} \|\mathbf{v}\|_\infty.$$

1.2.4 Angles

In \mathbb{R}^2 and \mathbb{R}^3 , the angle between two vectors is simply the angle between their arrows; see Figure 1.16.

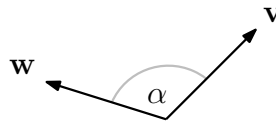


Figure 1.16: The angle α between two vectors \mathbf{v} and \mathbf{w}

As with the length, we are looking for a way to define the angle between two vectors also in higher dimensions, in such a way that nothing changes in \mathbb{R}^2 and \mathbb{R}^3 . The Cauchy-Schwarz inequality is the key here.

Definition 1.14 (Angle). Let $\mathbf{v}, \mathbf{w} \in \mathbb{R}^m$ be two nonzero vectors. The angle between them is the unique α between 0 and π (180 degrees) such that

$$\cos(\alpha) = \frac{\mathbf{v} \cdot \mathbf{w}}{\|\mathbf{v}\| \|\mathbf{w}\|} \in [-1, 1].$$

(Here, the interval $[-1, 1] = \{x \in \mathbb{R} : -1 \leq x \leq 1\}$ comes from the Cauchy Schwarz inequality, Lemma 1.12). In other words,

$$\alpha = \arccos \left(\frac{\mathbf{v} \cdot \mathbf{w}}{\|\mathbf{v}\| \|\mathbf{w}\|} \right).$$

Let us check that this coincides with the usual concept of angles in \mathbb{R}^2 . As the angle does not depend on how long the arrows are, and whether we simultaneously rotate them, we can look at the case where $\mathbf{v} = \mathbf{e}_1$ and \mathbf{w} is another unit vector, with an angle of α between the arrows; see Figure 1.17.

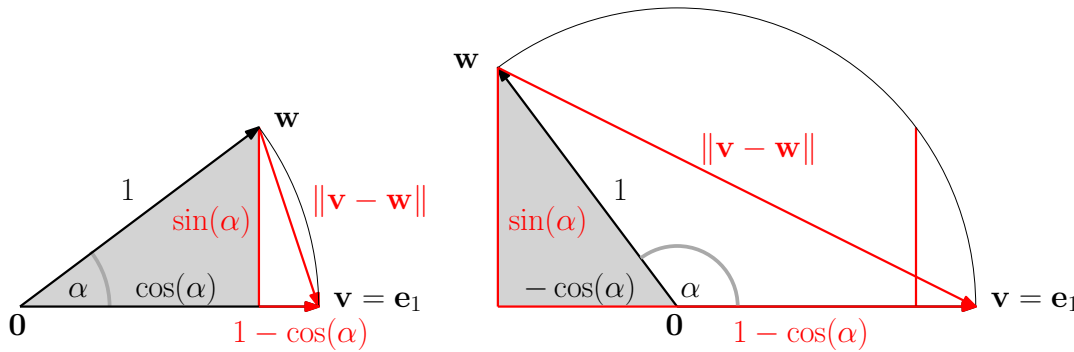


Figure 1.17: The angle α between two unit vectors in \mathbb{R}^2 . Left: α acute; right: α obtuse

From high school, we know that the legs of the gray triangle (whose hypotenuse is $\|\mathbf{w}\| = 1$) are $\sin(\alpha)$ and $\cos(\alpha)$ (if α is an acute angle) or $-\cos(\alpha)$ (if α is an obtuse angle). In both cases, the red triangle with hypotenuse $\|\mathbf{v} - \mathbf{w}\|$ therefore has legs $\sin(\alpha)$ and $1 - \cos(\alpha)$. By the Pythagorean theorem,

$$\|\mathbf{v} - \mathbf{w}\|^2 = \sin^2(\alpha) + (1 - \cos(\alpha))^2.$$

Using $\sin^2(\alpha) + \cos^2(\alpha) = 1$, the right-hand side is $2 - 2\cos(\alpha)$. By definition of the Euclidean norm, the left-hand side is $(\mathbf{v} - \mathbf{w}) \cdot (\mathbf{v} - \mathbf{w})$, and we have already argued in the proof of Lemma 1.12 (Cauchy-Schwarz inequality) that this equals $2 - 2\mathbf{v} \cdot \mathbf{w}$. Hence, we have shown

$$2 - 2\mathbf{v} \cdot \mathbf{w} = 2 - 2\cos(\alpha)$$

which simplifies to

$$\cos(\alpha) = \mathbf{v} \cdot \mathbf{w}.$$

This indeed agrees with what Definition 1.14 says for unit vectors.

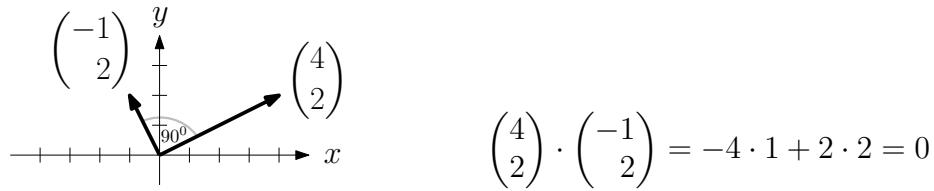


Figure 1.18: Orthogonal vectors: the scalar product equals 0.

Definition 1.15 (Orthogonal vectors). *Two vectors $\mathbf{v}, \mathbf{w} \in \mathbb{R}^m$ are orthogonal if $\mathbf{v} \cdot \mathbf{w} = 0$; in other words, if the cosine of the angle between them is 0 and the angle itself is 90 degrees.*

Figure 1.18 gives an example.

Given a nonzero vector $\mathbf{d} \in \mathbb{R}^m$ (for “direction”), we can collect all vectors $\mathbf{v} \in \mathbb{R}^m$ that are orthogonal to \mathbf{d} . The result is a *hyperplane* through the origin.

Definition 1.16 (Hyperplane through the origin). *Let $\mathbf{d} \in \mathbb{R}^m, \mathbf{d} \neq \mathbf{0}$. The set*

$$H_{\mathbf{d}} = \{\mathbf{v} \in \mathbb{R}^m : \mathbf{v} \cdot \mathbf{d} = 0\}$$

is called a hyperplane through the origin.

Figure 1.19 illustrates this. Indeed, $H_{\mathbf{d}}$ is going through the origin by definition: because $\mathbf{0}$ is orthogonal to every vector, we have $\mathbf{0} \in H_{\mathbf{d}}$.

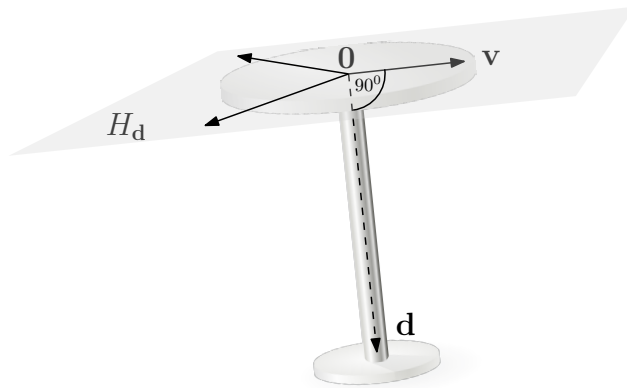


Figure 1.19: A hyperplane through the origin in \mathbb{R}^3 : if we imagine \mathbf{d} as the leg of a table, then $H_{\mathbf{d}}$ is the plane containing the table top.

A general hyperplane can be obtained from a hyperplane through the origin by a parallel translation. We will get to general hyperplanes at a later point.

1.2.5 Triangle inequality

Lemma 1.17. Let $\mathbf{v}, \mathbf{w} \in \mathbb{R}^m$. Then

$$\|\mathbf{v} + \mathbf{w}\| \leq \|\mathbf{v}\| + \|\mathbf{w}\|.$$

In \mathbb{R}^2 , Figure 1.20 shows that this is quite obvious from the parallelogram of vector addition (Figure 1.4).

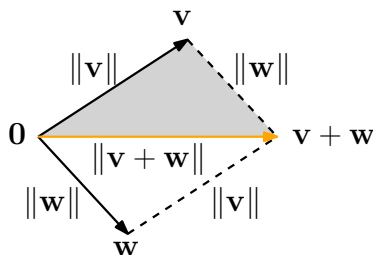


Figure 1.20: The triangle inequality: going from $\mathbf{0}$ directly to $\mathbf{v} + \mathbf{w}$ is shorter than making a detour via \mathbf{v} or \mathbf{w} .

In higher dimensions, the following proof shows that the triangle inequality is nothing but Cauchy-Schwarz in a different disguise.

Proof. Since both sides of the inequality are nonnegative, we can instead prove the squared triangle inequality

$$\|\mathbf{v} + \mathbf{w}\|^2 \leq (\|\mathbf{v}\| + \|\mathbf{w}\|)^2$$

and then take square roots on both sides to obtain the triangle inequality. To prove the squared version, we compute

$$\begin{aligned} \|\mathbf{v} + \mathbf{w}\|^2 &= (\mathbf{v} + \mathbf{w}) \cdot (\mathbf{v} + \mathbf{w}) \quad (\text{Definition 1.11 of the Euclidean norm}) \\ &= \mathbf{v} \cdot \mathbf{v} + \mathbf{w} \cdot \mathbf{w} + 2\mathbf{v} \cdot \mathbf{w} \quad (\text{Observation 1.10 (iii) on scalar products}) \\ &= \|\mathbf{v}\|^2 + \|\mathbf{w}\|^2 + 2\mathbf{v} \cdot \mathbf{w} \quad (\text{Definition 1.11 of the Euclidean norm}) \\ &\leq \|\mathbf{v}\|^2 + \|\mathbf{w}\|^2 + 2|\mathbf{v} \cdot \mathbf{w}| \\ &\leq \|\mathbf{v}\|^2 + \|\mathbf{w}\|^2 + 2\|\mathbf{v}\|\|\mathbf{w}\| \quad (\text{Cauchy Schwarz inequality, Lemma 1.12}) \\ &= (\|\mathbf{v}\| + \|\mathbf{w}\|)^2. \quad (\text{Binomial theorem}) \end{aligned}$$

□

Exercise 1.18. Turn this proof around and derive the Cauchy-Schwarz inequality from the (squared) triangle inequality!

As a consequence, some sources say that the Cauchy-Schwarz inequality is *equivalent* to the triangle inequality; this is another abuse of notation, since any two statements that are both true are logically equivalent, even if they have otherwise nothing to do with each other. What is meant here is that the triangle inequality can easily be proved using the Cauchy-Schwarz inequality (as we did in the proof above), and vice versa, as we ask you to do in Exercise 1.18.

1.2.6 Covectors and $\mathbf{v}^\top \mathbf{w}$

So far, we have used the notation $\mathbf{v} \cdot \mathbf{w}$ for the scalar product of two vectors, but “in the wild”, you also find other notations. A notable one is $\langle \mathbf{v}, \mathbf{w} \rangle$, but the most popular one is $\mathbf{v}^\top \mathbf{w}$. This notation is very useful in connection with matrices (see the next Chapter 2). Here, we want to introduce and explain it.

Long story short. A vector $\mathbf{v} \in \mathbb{R}^m$ is a mathematical object (sequence of m real numbers) that we write down as a column vector. Its *transpose* \mathbf{v}^\top is another mathematical object called a *covector* that we write down as a *row vector*. For example, if

$$\mathbf{v} = \begin{pmatrix} 1 \\ 2 \end{pmatrix}, \text{ then } \mathbf{v}^\top = (1 \ 2).$$

Generally, if

$$\mathbf{v} = \begin{pmatrix} v_1 \\ v_2 \\ \vdots \\ v_m \end{pmatrix}, \text{ then } \mathbf{v}^\top = (v_1 \ v_2 \ \cdots \ v_m).$$

This notation explains why we use the term *transpose* and the symbol $^\top$. It exchanges (“transposes”) a column and a row, and if we do this twice, we expect to arrive back at the original, so $(\mathbf{v}^\top)^\top = \mathbf{v}$.

A covector \mathbf{v}^\top and a vector \mathbf{w} of the same size can be multiplied, with the result being exactly the scalar product.

Definition 1.19 (Scalar product as covector-vector multiplication). *Let $\mathbf{v}, \mathbf{w} \in \mathbb{R}^m$. Then*

$$\mathbf{v}^\top \mathbf{w} = (v_1 \ v_2 \ \cdots \ v_m) \begin{pmatrix} w_1 \\ w_2 \\ \vdots \\ w_m \end{pmatrix} := \sum_{i=1}^m v_i w_i = \mathbf{v} \cdot \mathbf{w}.$$

This kind of multiplication may seem unnecessary, as it does not do anything more than Definition 1.9, after turning \mathbf{v} into a covector \mathbf{v}^\top . For the *long story short*, it is sufficient to know that the above “transpose” and “covector times vector” operations also appear in the context of matrices (Chapter 2), with essentially the same definitions for matrices that look like row or column vectors. As a consequence, \mathbf{v}^\top and $\mathbf{v}^\top \mathbf{w}$ are the natural language in (the many) situations involving both vectors *and* matrices. We will discuss this in detail in Section 2.3.4.

Short story long. Now, what exactly is covector? It is formally a *function*, and we provide a brief introduction to functions as mathematical objects immediately after the following formal definition of a covector.

Definition 1.20 (Covector). Let $\mathbf{v} \in \mathbb{R}^m$ be a vector. The covector \mathbf{v}^\top (also called the transpose of \mathbf{v}) is the function $\mathbf{v}^\top : \mathbb{R}^m \rightarrow \mathbb{R}$,

$$\mathbf{v}^\top : \mathbf{x} \mapsto \sum_{i=1}^m v_i x_i.$$

We also define $(\mathbf{v}^\top)^\top := \mathbf{v}$ and call the vector \mathbf{v} the transpose of the covector \mathbf{v}^\top .

In row vector notation, \mathbf{v}^\top is written as

$$\mathbf{v}^\top = (v_1 \quad v_2 \quad \cdots \quad v_m).$$

The set of m -dimensional covectors is denoted by $(\mathbb{R}^m)^*$ and called the dual space of \mathbb{R}^m . A covector is in some sources also called linear form or linear functional.

As functions and their notation appear here for the first time, here is a very small “crash course” on functions.

Generally, a function $f : X \rightarrow Y$ does the following: whenever it gets an “input” x from its domain X (a set), it produces an “output” $f(x) \in Y$ from its codomain Y (another set). We also say that f maps x to $f(x)$, or that we apply f to the input x to get the output $f(x)$. The same input always leads to the same output. The element $f(x) \in Y$ is also called the function value.

To describe how the mapping from input to output works, we use a definition of the form $f : x \mapsto \text{“expression in } x\text{”}$. For example, $f : \mathbb{R} \rightarrow \mathbb{R}$, $f : x \mapsto x^2$ is the function that maps a given real number $x \in \mathbb{R}$ to its square $x^2 \in \mathbb{R}$. If f is clear from the context, we do not have to repeat it in the definition. For example, the previous square function could also be defined as $f : \mathbb{R} \rightarrow \mathbb{R}$, $x \mapsto x^2$.

In case of a covector $\mathbf{v}^\top : \mathbb{R}^m \rightarrow \mathbb{R}$, the definition of the mapping is $\mathbf{v}^\top : \mathbf{x} \mapsto \sum_{i=1}^m v_i x_i$, i.e. given input $\mathbf{x} \in \mathbb{R}^m$, the output is the real number $\mathbf{v}^\top(\mathbf{x}) = \sum_{i=1}^m v_i x_i$.

For example, the covector $\mathbf{v}^\top = (3 \quad 4) \in (\mathbb{R}^2)^*$ is the function $\mathbf{v}^\top : \mathbf{x} \mapsto 3x_1 + 4x_2$. Just as vectors, covectors can be added and multiplied with scalars, by simply performing the corresponding operations on their underlying vectors. For example, $2(3 \quad 4) = (6 \quad 8)$, the function $\mathbf{x} \mapsto 6x_1 + 8x_2$. The covector $\mathbf{0}^\top$ is the zero covector, the function $\mathbf{x} \mapsto 0$.

With covectors, the scalar product of two vectors $\mathbf{v}, \mathbf{w} \in \mathbb{R}^n$ can be written as $\mathbf{v}^\top(\mathbf{w})$ (meaning that we apply the function \mathbf{v}^\top to the vector \mathbf{w}). Indeed, by Definition 1.20, the result of this is

$$\mathbf{v}^\top(\mathbf{w}) = \sum_{i=1}^m v_i w_i = \mathbf{v} \cdot \mathbf{w},$$

the scalar product! If we omit the brackets around the function argument \mathbf{w} (a very common abuse of notation), we arrive at the notation $\mathbf{v}^\top \mathbf{w}$ for the scalar product. So although we think of $\mathbf{v}^\top \mathbf{w}$ as multiplication (of a covector and a vector), it formally is the application of the function \mathbf{v}^\top to the vector \mathbf{w} .

1.3 Linear (in)dependence

Linear independence of vectors is probably the single most important concept of linear algebra. A sequence of vectors is called linearly independent if none of the vectors is a linear combination of the others, and linearly dependent otherwise. We provide a number of alternative definitions of linear (in)dependence that illuminate the concept from different angles. We also introduce the span of a sequence of vectors, the set of all their linear combinations, and prove some important properties of the span.

1.3.1 Definition and examples

Definition 1.21 (Linear (in)dependence). *Vectors $\mathbf{v}_1, \mathbf{v}_2, \dots, \mathbf{v}_n \in \mathbb{R}^m$ are linearly dependent if at least one of them is a linear combination of the others, i.e. there is an index $k \in [n]$ and scalars λ_j such that*

$$\mathbf{v}_k = \sum_{\substack{j=1 \\ j \neq k}}^n \lambda_j \mathbf{v}_j.$$

Otherwise, $\mathbf{v}_1, \mathbf{v}_2, \dots, \mathbf{v}_n$ are linearly independent.

Here, the additional “ $j \neq k$ ” below the sum adds a *condition* to the j ’s considered in the sum: take only the ones in the given range that satisfy the condition. Hence, the sum is over all j except k , so the equation indeed says that \mathbf{v}_k is a linear combination of the other vectors.

Let us do some examples. The two vectors

$$\begin{pmatrix} 2 \\ 3 \end{pmatrix}, \begin{pmatrix} 4 \\ 6 \end{pmatrix}$$

are linearly dependent, because

$$\begin{pmatrix} 2 \\ 3 \end{pmatrix} = \frac{1}{2} \begin{pmatrix} 4 \\ 6 \end{pmatrix} \text{ or } \begin{pmatrix} 4 \\ 6 \end{pmatrix} = 2 \begin{pmatrix} 2 \\ 3 \end{pmatrix}.$$

We also call these two vectors *collinear* because they are on the same line; see Figure 1.21 (left). In contrast, the two vectors

$$\begin{pmatrix} 2 \\ 3 \end{pmatrix}, \begin{pmatrix} 3 \\ -1 \end{pmatrix}$$

that we have considered in Fact 1.5 are linearly independent, as none of them is a linear combination (scalar multiple) of the other one; see Figure 1.21 (right).

Let us now consider three vectors in \mathbb{R}^2 . These are always linearly dependent: If two of them are collinear, one of them is a linear combination of the other one and therefore

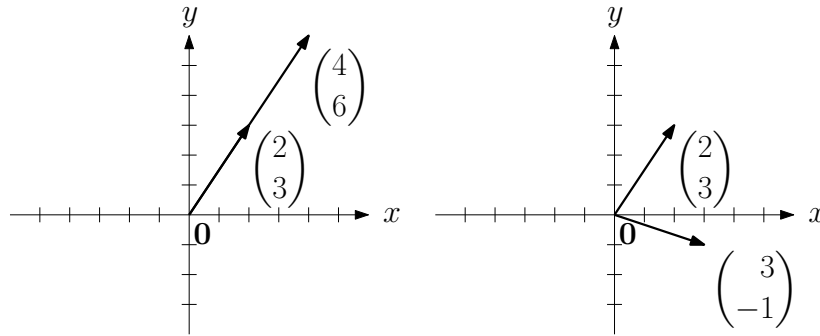


Figure 1.21: Two linearly dependent vectors (left); two linearly independent vectors (right)

of *both* other ones (pick scalar 0 for the second other one). Otherwise, each of the vectors is a linear combination of the other two by Challenge 1.6.

What if we have just one vector $\mathbf{v} \in \mathbb{R}^m$? Can \mathbf{v} even be a linear combination of the other vectors when there are no other vectors? Yes, if $\mathbf{v} = \mathbf{0}$, because $\mathbf{0}$ is a linear combination of the empty sequence of vectors (Section 1.1.5). But if $\mathbf{v} \neq \mathbf{0}$, the sequence consisting only of the vector \mathbf{v} is linearly independent.

Generally, when $\mathbf{0}$ is one of the vectors, they are automatically linearly dependent, because $\mathbf{0}$ is always a linear combination of the other ones, even if there are no other ones. Similarly, when some vector appears twice, the vectors are linearly dependent, because one of the copies is already a linear combination of the other copy.

Finally, what about the empty sequence of vectors? This in turn is linearly independent by Definition 1.21: because $[n] = \emptyset$ in this case (\emptyset is the symbol for the *empty set*), there is no index $k \in [n]$, whatever we may require of it. Table 1.3 summarizes these examples.

linearly independent	linearly dependent
$\begin{pmatrix} 2 \\ 3 \end{pmatrix}, \begin{pmatrix} 3 \\ -1 \end{pmatrix}$	
	$\begin{pmatrix} 2 \\ 3 \end{pmatrix}, \begin{pmatrix} 4 \\ 6 \end{pmatrix}$
	$\mathbf{v}_1, \mathbf{v}_2, \mathbf{v}_3 \in \mathbb{R}^2$
$\mathbf{v} \neq \mathbf{0}$	
	$\mathbf{v} = \mathbf{0}$
	$\dots, \mathbf{0}, \dots$
	$\dots, \mathbf{v}, \dots, \mathbf{v}, \dots$
empty sequence $()$	

Table 1.3: Linear (in)dependence of some sequences of vectors

1.3.2 Alternative definitions

There are two more important alternative definitions of linear dependence. The following lemma provides them. In some sources, (ii) is the “standard” definition of linear dependence.

Lemma 1.22 (Alternative definitions of linear dependence). *Let $\mathbf{v}_1, \mathbf{v}_2, \dots, \mathbf{v}_n \in \mathbb{R}^m$. The following statements are equivalent (meaning that they are either all true, or all false).*

- (i) *At least one of the vectors is a linear combination of the other ones. (This means, the vectors are linearly dependent according to Definition 1.21.)*
- (ii) *There are scalars $\lambda_1, \lambda_2, \dots, \lambda_n$ besides $0, 0, \dots, 0$ such that $\sum_{j=1}^n \lambda_j \mathbf{v}_j = \mathbf{0}$. We also say that $\mathbf{0}$ is a nontrivial linear combination of the vectors.*
- (iii) *At least one of the vectors is a linear combination of the previous ones.*

For the proof, we apply the basic principles of logic. We first argue that (i) *implies* (ii), meaning that *if* (i) is true, *then* also (ii) is true. Logically, this is written as $(i) \Rightarrow (ii)$. Next we prove $(ii) \Rightarrow (iii)$ and $(iii) \Rightarrow (i)$.

Having done this, we know that (i), (ii) and (iii) are equivalent: either all true or all false, written as $(i) \Leftrightarrow (ii) \Leftrightarrow (iii)$. Indeed, because of the three (circular) implications, it cannot be that one of the statements is true and another one is false.

In math prose, an equivalence such as $(i) \Leftrightarrow (ii)$ is also written as “(i) *if and only if* (ii)”. This summarizes the two implications : “(i) *if* (ii)” writes out $(ii) \Rightarrow (i)$, while “(i) *only if* (ii)” excludes the possibility that (i) is true and (ii) is false. In other words, it writes out the implication $(i) \Rightarrow (ii)$.

Proof.

$(i) \Rightarrow (ii)$: If at least one of the vectors, \mathbf{v}_k say, is a linear combination of the other vectors, then

$$\mathbf{v}_k = \sum_{\substack{j=1 \\ j \neq k}}^n \lambda_j \mathbf{v}_j.$$

This can be written as

$$\mathbf{0} = \sum_{\substack{j=1 \\ j \neq k}}^n \lambda_j \mathbf{v}_j - \mathbf{v}_k,$$

so if we define $\lambda_k = -1$, we get

$$\mathbf{0} = \sum_{j=1}^n \lambda_j \mathbf{v}_j.$$

Hence, $\mathbf{0}$ is a nontrivial linear combination of the vectors (it is nontrivial because $\lambda_k \neq 0$).

(ii) \Rightarrow (iii): If $\mathbf{0}$ is a nontrivial linear combination of the vectors, then we can write $\mathbf{0}$ in the form

$$\mathbf{0} = \sum_{j=1}^n \lambda_j \mathbf{v}_j,$$

where not all λ_j are zero. Let k be the largest index such that $\lambda_k \neq 0$. Then we actually have

$$\mathbf{0} = \sum_{j=1}^k \lambda_j \mathbf{v}_j, \text{ which we can also write as } \lambda_k \mathbf{v}_k = \sum_{j=1}^{k-1} -\lambda_j \mathbf{v}_j.$$

Dividing by $\lambda_k \neq 0$, we get

$$\mathbf{v}_k = \sum_{j=1}^{k-1} \left(-\frac{\lambda_j}{\lambda_k} \right) \mathbf{v}_j.$$

Hence, at least one vector, namely \mathbf{v}_k , is a linear combination of the previous ones.

(iii) \Rightarrow (i): If at least one vector is a linear combination of the previous ones, the same vector is also a linear combination of the other ones (use scalar 0 for vectors after it). \square

We have in particular proved that (i) implies (iii). So if *some* vector is a linear combination of the other ones, then *some* vector is a linear combination of the previous ones. However, this is not necessarily the same vector. As an example, consider the sequence

$$\begin{pmatrix} 1 \\ 1 \end{pmatrix}, \begin{pmatrix} 1 \\ 0 \end{pmatrix}, \begin{pmatrix} 0 \\ 1 \end{pmatrix}.$$

Here, the first vector is a linear combination of the other two, but the first vector is *not* a linear combination of the previous ones (there are no previous ones, and the vector is nonzero). Here, only the third vector is a linear combination of the previous ones.

Here are the corresponding alternative definitions of linear *independence*. These are simply obtained by taking the opposites (logical *negations*) of (i)–(iii) in Lemma 1.22: If some statements are either all true or all false, the same holds for their opposites.

We formulate the resulting definitions as a *corollary* which is a result that directly *follows* from (is implied by) a previous one (in this case Lemma 1.22), without the need for a proof (or only a very simple proof such as “take the opposites of all statements!”).

Corollary 1.23 (Alternative definitions of linear independence). *Let $\mathbf{v}_1, \mathbf{v}_2, \dots, \mathbf{v}_n \in \mathbb{R}^m$. The following statements are equivalent (meaning that they are either all true, or all false).*

- (i) *None of the vectors is a linear combination of the other ones. (This means, the vectors are linearly independent according to Definition 1.21.)*
- (ii) *There are no scalars $\lambda_1, \lambda_2, \dots, \lambda_n$ besides $0, 0, \dots, 0$ such that $\sum_{j=1}^n \lambda_j \mathbf{v}_j = \mathbf{0}$. We also say that $\mathbf{0}$ can only be written as a trivial linear combination of the vectors.*
- (iii) *None of the vectors is a linear combination of the previous ones.*

This has another important consequence: a linear combination of linearly independent vectors can be written as a linear combination *in only one way*.

Lemma 1.24. Let $\mathbf{v}_1, \mathbf{v}_2, \dots, \mathbf{v}_n \in \mathbb{R}^m$ be linearly independent, $\mathbf{v} \in \mathbb{R}^m$. Let

$$\mathbf{v} = \sum_{j=1}^n \lambda_j \mathbf{v}_j = \sum_{j=1}^n \mu_j \mathbf{v}_j$$

be two ways of writing \mathbf{v} as a linear combination of $\mathbf{v}_1, \mathbf{v}_2, \dots, \mathbf{v}_n$. Then $\lambda_j = \mu_j$ for all $j \in [n]$.

Proof. Subtracting the two linear combinations gives $\mathbf{0} = \sum_{j=1}^n (\lambda_j - \mu_j) \mathbf{v}_j$. Since $\mathbf{0}$ can only be written as a trivial linear combination of $\mathbf{v}_1, \mathbf{v}_2, \dots, \mathbf{v}_n$ by Corollary 1.23 (ii), we get $\lambda_j - \mu_j = 0$ for all j , so $\lambda_j = \mu_j$ for all j . \square

1.3.3 Span of vectors

The set of all linear combinations of some vectors is important enough to deserve a name.

Definition 1.25 (Span). Let $\mathbf{v}_1, \mathbf{v}_2, \dots, \mathbf{v}_n \in \mathbb{R}^m$. Their span is the set of all linear combinations. In formulas,

$$\text{Span}(\mathbf{v}_1, \mathbf{v}_2, \dots, \mathbf{v}_n) := \left\{ \sum_{j=1}^n \lambda_j \mathbf{v}_j : \lambda_j \in \mathbb{R} \text{ for all } j \in [n] \right\}.$$

As an example, let us consider the span of three vectors $\mathbf{v}_1, \mathbf{v}_2, \mathbf{v}_3$ in \mathbb{R}^3 . There are three cases, see Figure 1.22.

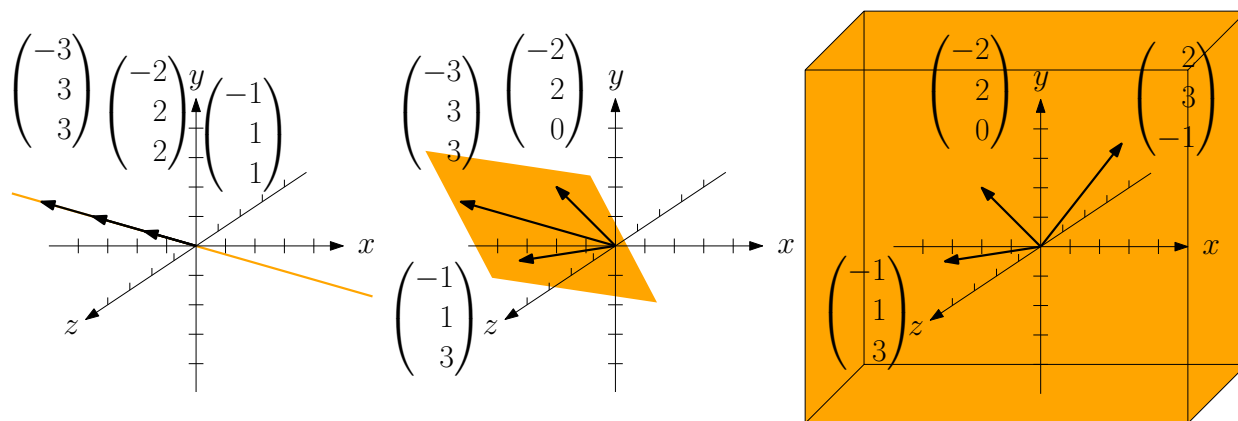


Figure 1.22: The span of three vectors in \mathbb{R}^3 : a line, a plane, or the whole space

We do not yet have the tools to prove this here, but simply appeal to the geometric intuition: the span can be a line through the origin (vectors are *collinear*), a plane through

the origin (vectors are *coplanar*), or the whole space (vectors are linearly independent). In Section 4.3.1, we will have the tools available that allow us to formalize the intuition.

The attentive reader might have noticed that there is a fourth (but pretty boring) case: if $\mathbf{v}_1 = \mathbf{v}_2 = \mathbf{v}_3 = \mathbf{0}$, then $\mathbf{0}$ is the only linear combination, so the span consists of a single point at the origin. Here are some more observations to illustrate the concept.

We always have $\mathbf{0} \in \text{Span}(\mathbf{v}_1, \mathbf{v}_2, \dots, \mathbf{v}_n)$ (obtained by setting all λ_j to 0). This even holds if $n = 0$ and there are no vectors. As we have argued in Section 1.1.5, $\mathbf{0}$ is a linear combination of the empty sequence of vectors (and the only one), so $\text{Span}() = \{\mathbf{0}\}$.

In “span language,” Fact 1.5 can be rewritten as follows:

$$\text{Span}\left(\begin{pmatrix} 2 \\ 3 \end{pmatrix}, \begin{pmatrix} 3 \\ -1 \end{pmatrix}\right) = \mathbb{R}^2.$$

The span of two nonzero vectors that are scalar multiples of each other is always a line. For example,

$$\text{Span}\left(\begin{pmatrix} 2 \\ 3 \end{pmatrix}, \begin{pmatrix} 4 \\ 6 \end{pmatrix}\right) = \left\{ \lambda \begin{pmatrix} 2 \\ 3 \end{pmatrix} : \lambda \in \mathbb{R} \right\}.$$

Figure 1.23 illustrates these two cases.

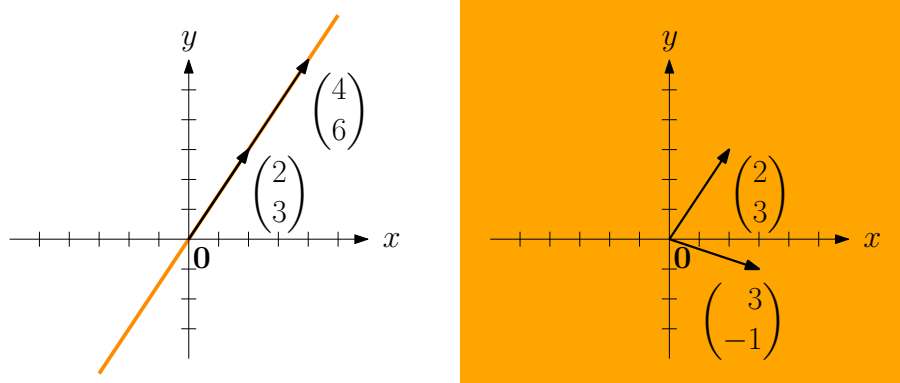


Figure 1.23: The span of two vectors in \mathbb{R}^2 : a line, or the whole space

Next we prove a very useful statement that may seem obvious from Figure 1.22 but needs a proof: the span of vectors does not change when we add a linear combination of them as a new vector.

Lemma 1.26. *Let $\mathbf{v}_1, \mathbf{v}_2, \dots, \mathbf{v}_n \in \mathbb{R}^m$, and let $\mathbf{v} \in \mathbb{R}^m$ be a linear combination of $\mathbf{v}_1, \mathbf{v}_2, \dots, \mathbf{v}_n$. Then*

$$\underbrace{\text{Span}(\mathbf{v}_1, \mathbf{v}_2, \dots, \mathbf{v}_n)}_S = \underbrace{\text{Span}(\mathbf{v}_1, \mathbf{v}_2, \dots, \mathbf{v}_n, \mathbf{v})}_T.$$

For the proof, we need to show that two sets S and T are equal. For this, we typically argue that each element of S is contained in T (then S is a *subset* of T , in formulas $S \subseteq T$)

and—vice versa—that each element of T is contained in S (then $T \subseteq S$). Having done this, there cannot be an element which is in one of the sets and not in the other one (same logic as with the implications in Lemma 1.22), so the two sets must be equal.

Proof of Lemma 1.26.

$S \subseteq T$: Each element $\mathbf{w} \in S$ is a linear combination of $\mathbf{v}_1, \mathbf{v}_2, \dots, \mathbf{v}_n$ and therefore also a linear combination of $\mathbf{v}_1, \mathbf{v}_2, \dots, \mathbf{v}_n, \mathbf{v}$ (add the scalar multiple $0\mathbf{v}$). So $\mathbf{w} \in T$.

$T \subseteq S$: each element $\mathbf{w} \in T$ is a linear combination of $\mathbf{v}_1, \mathbf{v}_2, \dots, \mathbf{v}_n, \mathbf{v}$,

$$\mathbf{w} = \sum_{j=1}^n \lambda_j \mathbf{v}_j + \lambda \mathbf{v}.$$

But since \mathbf{v} is a linear combination of $\mathbf{v}_1, \mathbf{v}_2, \dots, \mathbf{v}_n$, we can also write \mathbf{v} as

$$\mathbf{v} = \sum_{j=1}^n \mu_j \mathbf{v}_j.$$

Plugging the second equation into the first one, we get

$$\mathbf{w} = \sum_{j=1}^n \lambda_j \mathbf{v}_j + \lambda \mathbf{v} = \sum_{j=1}^n \lambda_j \mathbf{v}_j + \lambda \left(\sum_{j=1}^n \mu_j \mathbf{v}_j \right) = \sum_{j=1}^n (\lambda_j + \lambda \mu_j) \mathbf{v}_j.$$

This means that \mathbf{w} is a linear combination of $\mathbf{v}_1, \mathbf{v}_2, \dots, \mathbf{v}_n$ and hence in S . \square

The fact that the new vector is the last one in the right sequence of Lemma 1.26 has no significance. Because vector addition is commutative, the span does not depend on how we order the vectors. So another useful “backward” way of reading this lemma is the following: if some vector in a sequence is a linear combination of other vectors in the sequence, this vector can be removed from the sequence without changing the span. We formalize this in a corollary.

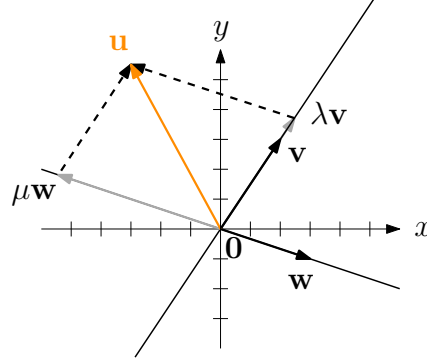
Corollary 1.27. *Let $\mathbf{v}_1, \mathbf{v}_2, \dots, \mathbf{v}_n \in \mathbb{R}^m$ and suppose that for some $k \in [n]$, \mathbf{v}_k is a linear combination of the other vectors. Then*

$$\text{Span}(\mathbf{v}_1, \mathbf{v}_2, \dots, \mathbf{v}_n) = \text{Span}(\mathbf{v}_1, \mathbf{v}_2, \dots, \mathbf{v}_{k-1}, \mathbf{v}_{k+1}, \mathbf{v}_{k+2}, \dots, \mathbf{v}_n).$$

We conclude this chapter with another important result that seems obvious from our geometric intuition but needs a proof.

Lemma 1.28 (The span of m linearly independent vectors is \mathbb{R}^m). *Let $\mathbf{v}_1, \mathbf{v}_2, \dots, \mathbf{v}_m \in \mathbb{R}^m$ be linearly independent. Then $\text{Span}(\mathbf{v}_1, \mathbf{v}_2, \dots, \mathbf{v}_m) = \mathbb{R}^m$.*

Recall Challenge 1.6 that asks you to prove that the span of two linearly independent vectors $\mathbf{v}, \mathbf{w} \in \mathbb{R}^2$ is the whole plane \mathbb{R}^2 . For two specific such vectors, we have proved this in Fact 1.5, but the same proof can be extended such that it works for any two linearly independent vectors. Geometrically, we can understand this with a column picture as in Figure 1.8: if \mathbf{v} and \mathbf{w} are not pointing into the same direction, any target vector \mathbf{u} can be expressed as the sum of scaled versions of \mathbf{v} and \mathbf{w} :



In the proof of Fact 1.5, we have found the scaling factors λ and μ by solving a system of two equations in two variables. We could follow the same approach to prove Lemma 1.28: every target vector $\mathbf{u} \in \mathbb{R}^m$ is a sum of scaled versions of the $\mathbf{v}_i, i \in [m]$, and the corresponding scaling factors $\lambda_i, i \in [m]$ can be found by solving a system of m equations in m variables. Unfortunately, we will develop the necessary theory for this only in Chapter 3. But maybe this is actually fortunate, because there is a simpler proof that we can do already now.

Proof of Lemma 1.28. The strategy is the following: we start with a sequence of m vectors whose span is obviously the whole space \mathbb{R}^m . One by one, we replace them by the \mathbf{v}_i 's and argue that the span never changes. In the end, the sequence is $\mathbf{v}_1, \mathbf{v}_2, \dots, \mathbf{v}_m \in \mathbb{R}^m$, and the span is still \mathbb{R}^m .

The “obvious” sequence is $\mathbf{e}_1, \mathbf{e}_2, \dots, \mathbf{e}_m$, consisting of the m standard unit vectors; see page 27. Every target vector $\mathbf{u} \in \mathbb{R}^m$ can be written as the linear combination

$$\mathbf{u} = \sum_{i=1}^m u_i \mathbf{e}_i,$$

so the span is indeed the whole space \mathbb{R}^m ; in \mathbb{R}^3 , this looks like

$$\begin{pmatrix} u_1 \\ u_2 \\ u_3 \end{pmatrix} = u_1 \begin{pmatrix} 1 \\ 0 \\ 0 \end{pmatrix} + u_2 \begin{pmatrix} 0 \\ 1 \\ 0 \end{pmatrix} + u_3 \begin{pmatrix} 0 \\ 0 \\ 1 \end{pmatrix}.$$

Now we consider the sequence

$$\mathbf{v}_1, \mathbf{e}_1, \mathbf{e}_2, \dots, \mathbf{e}_m,$$

obtained by adding \mathbf{v}_1 before the standard unit vectors. Since $\text{Span}(\mathbf{e}_1, \mathbf{e}_2, \dots, \mathbf{e}_m) = \mathbb{R}^m$, we know that $\mathbf{v}_1 \in \mathbb{R}^m$ is a linear combination of $\mathbf{e}_1, \mathbf{e}_2, \dots, \mathbf{e}_m$, and hence the $m+1$ vectors are linearly dependent by Definition 1.21. Then we also know that one of the vectors is a linear combination of the *previous* ones in the sequence, see Lemma 1.22 (iii). This vector cannot be \mathbf{v}_1 because \mathbf{v}_1 also starts the linearly independent sequence $\mathbf{v}_1, \mathbf{v}_2, \dots, \mathbf{v}_m$ in which no vector is a linear combination of the previous ones by Corollary 1.23 (iii). Therefore, one of the \mathbf{e}_i 's is a linear combination of the previous vectors. Removing that

vector does not change the span (see the previous Corollary 1.27), so we get a sequence $\mathbf{v}_1, \mathbf{u}_2, \mathbf{u}_3, \dots, \mathbf{u}_m$ (the \mathbf{u}_i 's name the $m - 1$ remaining standard unit vectors) with

$$\text{Span}(\mathbf{v}_1, \mathbf{u}_2, \mathbf{u}_3, \dots, \mathbf{u}_m) = \mathbb{R}^m.$$

So we have successfully replaced one of the standard unit vectors with \mathbf{v}_1 , without changing the span. Now we do the same with \mathbf{v}_2 : we add \mathbf{v}_2 directly after \mathbf{v}_1 and argue as before that the $m + 1$ vectors

$$\mathbf{v}_1, \mathbf{v}_2, \mathbf{u}_2, \mathbf{u}_3, \dots, \mathbf{u}_m$$

must be linearly dependent; therefore one of them is a linear combination of the previous ones. This can neither be \mathbf{v}_1 nor \mathbf{v}_2 , because $\mathbf{v}_1, \mathbf{v}_2$ also start a sequence of linearly independent vectors. Hence, some \mathbf{u}_i is a linear combination of the previous vectors and can be removed without changing the span. We call the $m - 2$ remaining standard unit vectors $\mathbf{u}_3, \mathbf{u}_4, \dots, \mathbf{u}_m$ (a slight abuse of notation, as this redefines the \mathbf{u}_i 's) and get

$$\text{Span}(\mathbf{v}_1, \mathbf{v}_2, \mathbf{u}_3, \mathbf{u}_4, \dots, \mathbf{u}_m) = \mathbb{R}^m.$$

By now, the pattern should be clear. After k replacement steps, we have $m - k$ remaining standard unit vectors $\mathbf{u}_{k+1}, \mathbf{u}_{k+2}, \dots, \mathbf{u}_m$ and

$$\text{Span}(\mathbf{v}_1, \mathbf{v}_2, \dots, \mathbf{v}_k, \mathbf{u}_{k+1}, \mathbf{u}_{k+2}, \dots, \mathbf{u}_m) = \mathbb{R}^m.$$

After $k = m$ steps, we get our desired result:

$$\text{Span}(\mathbf{v}_1, \mathbf{v}_2, \dots, \mathbf{v}_m) = \mathbb{R}^m.$$

What we have done in this proof (exchange vectors without changing the span) is an important technique. It will reappear later in the proof of a more general and fundamental result of linear algebra, the *Steinitz exchange lemma*, see Section 4.2.3. This particular proof is due to Oleksandr Kulkov, TA for this course in HS24.

□

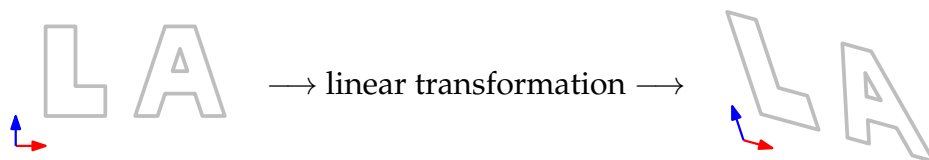
Chapter 2

Matrices

Upfront, a matrix is just a table of real numbers, for example

$$\begin{bmatrix} 3 & 5 & 2 & -1 \\ 0 & -6 & 5 & 4 \\ -3 & 2 & -6 & 8 \end{bmatrix}.$$

But unlike column or row vectors that are (for us) simply notations for vectors and co-vectors, we treat matrices as mathematical objects on their own. In linear algebra, matrices are used to compactly represent sequences of vectors (the columns of the matrix) or covectors (the rows of the matrix). But most importantly, matrices can represent *linear transformations* that act on vectors and shapes according to specific rules:



Linear transformations are everywhere. Computer graphics is a classical application. Whenever 3D objects move or rotate in a computer game, linear transformations happen in the background. Today, neural network computations in machine learning and artificial intelligence involve massive amounts of linear transformations. Consequently, *AI chips* are optimized for fast matrix computations. In this chapter, we get to know matrices in detail and understand how they represent linear transformations.

Important operations such as matrix-vector multiplication, matrix multiplication, or matrix inversion are motivated from the point of view of linear transformations.

Advanced matrix concepts such as determinants, eigenvalues and eigenvectors are not covered in this chapter; they will be introduced in the second part of the course.

It can be quite interesting to ask “the Internet” (try Google, or ChatGPT, for example) what the difference between vectors and matrices is. Here is an answer that you might not easily get but that conveys the spirit of the above introduction: vectors are the “raw material” of linear algebra, while matrices are an important part of the “toolkit” that has been developed to work with vectors.

2.1 Matrices and linear combinations

A matrix is a table of numbers. This can be seen as an efficient notation for a sequence of vectors (the columns of the matrix), or a sequence of covectors (the rows of the matrix). Therefore, matrices turn out to be very useful in arguing about or computing with sequences of (co)vectors and their linear combinations. Central matrix concepts that we introduce here are matrix-vector multiplication, the column space, the row space, the transpose, the rank, and the nullspace.

We often work with sequences of vectors. A *matrix* can be considered as a more compact notation for such a sequence. For example,

$$\begin{bmatrix} 1 & 2 \\ 3 & 4 \\ 5 & 6 \end{bmatrix} \text{ is a } 3 \times 2 \text{ matrix (3 rows, 2 columns) that represents the sequence } \left(\begin{pmatrix} 1 \\ 3 \\ 5 \end{pmatrix}, \begin{pmatrix} 2 \\ 4 \\ 6 \end{pmatrix} \right)$$

of two column vectors in \mathbb{R}^3 . Using the matrix, we save brackets and commas, but more importantly, it also represents a sequence of three covectors in $(\mathbb{R}^2)^*$ (see Section 1.2.6):

$$((1 \ 2), (3 \ 4), (5 \ 6)).$$

Definition 2.1 (Matrix). *An $m \times n$ matrix is a table of real numbers with m rows and n columns. We use upper-case letters (A, B, \dots) to denote matrices, and write their entries with the corresponding lower case letters and two indices, as in*

$$A = \begin{bmatrix} a_{11} & a_{12} & \cdots & a_{1n} \\ a_{21} & a_{22} & \cdots & a_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ a_{m1} & a_{m2} & \cdots & a_{mn} \end{bmatrix}.$$

Hence, a_{ij} is the entry in row i and column j of matrix A . The “dot-free” notation (see also Section 1.1.5) is

$$A = [a_{ij}]_{i=1, j=1}^{m, n}.$$

The set of $m \times n$ matrices is denoted by $\mathbb{R}^{m \times n}$.

If we want to talk about the columns of A as vectors or the rows of A as covectors, we use column notation or row notation,

$$A = \underbrace{\begin{bmatrix} | & | & \cdots & | \\ \mathbf{v}_1 & \mathbf{v}_2 & \cdots & \mathbf{v}_n \\ | & | & \cdots & | \end{bmatrix}}_{\substack{\text{column notation with} \\ \mathbf{v}_1, \mathbf{v}_2, \dots, \mathbf{v}_n \in \mathbb{R}^m}}, \quad A = \underbrace{\begin{bmatrix} - & \mathbf{u}_1^\top & - \\ - & \mathbf{u}_2^\top & - \\ & \vdots & \\ - & \mathbf{u}_m^\top & - \end{bmatrix}}_{\substack{\text{row notation with} \\ \mathbf{u}_1^\top, \mathbf{u}_2^\top, \dots, \mathbf{u}_m^\top \in (\mathbb{R}^n)^*}}.$$

Here,

$$\mathbf{v}_j = \begin{pmatrix} a_{1j} \\ a_{2j} \\ \vdots \\ a_{mj} \end{pmatrix} \text{ (} j\text{-th column), and } \mathbf{u}_i^\top = (a_{i1} \ a_{i2} \ \cdots \ a_{in}) \text{ (} i\text{-th row)}.$$

While a vector needs one dot symbol in “dot notation”, a matrix needs seven. This is significant enough to use the other notations (dot-free, column, row) more frequently for matrices.

Some sources use the notation $a_{i,j}$ instead of a_{ij} , and indeed, this would avoid some ambiguities, in particular for concrete values of i and j . For example, we use a_{12} for the second entry in the first row of matrix A , but this could be misunderstood as the twelfth entry of a vector \mathbf{a} . The notation $a_{1,2}$ would avoid this misunderstanding.

However, if we are in the context of matrices, *and* there are only two index symbols (such as ij or 12), then things are clear: the first symbol stands for the row, and the second one for the column. If there are more symbols, it is different: we better write $a_{37,25}$ to refer to the entry in row 37 and column 25 of A , since the ambiguities in a_{3725} cannot easily be resolved. But as more than two symbols are rare, it is customary and economical to use separating commas only in this case and not always.

What is a matrix as a mathematical object? Definition 2.1 of a matrix was rather informal (“table of real numbers”). For all practical purposes, this is sufficient and intuitive. But we can easily turn this into a formal definition: let us say that a matrix is officially a function $A : [m] \times [n] \rightarrow \mathbb{R}$: for each $i \in [m]$ (row index) and $j \in [n]$ (column index), the function value $A(i, j)$ is some real number, providing the entry in row i and column j of the matrix. Thus, the “table of real numbers” is simply the value table of the function A , and a_{ij} is a shortcut for $A(i, j)$. The symbol $\mathbb{R}^{m \times n}$ that we have introduced for the set of all matrices corresponds to this definition.

After this digression, let us turn to what we can do with matrices: Just like vectors, two matrices of the same shape can be added and multiplied with a scalar, as in the following examples:

$$\begin{bmatrix} 1 & 2 \\ 3 & 4 \end{bmatrix} + \begin{bmatrix} 5 & 6 \\ 7 & 8 \end{bmatrix} = \begin{bmatrix} 6 & 8 \\ 10 & 12 \end{bmatrix}, \quad 2 \begin{bmatrix} 1 & 2 \\ 3 & 4 \end{bmatrix} = \begin{bmatrix} 2 & 4 \\ 6 & 8 \end{bmatrix}.$$

Definition 2.2 (Matrix addition, scalar multiplication, zero matrix, square matrix). Let $A = [a_{ij}]_{i=1, j=1}^m, n$ and $B = [b_{ij}]_{i=1, j=1}^m, n$ be $m \times n$ matrices, $\lambda \in \mathbb{R}$ a scalar.

- (i) The matrix $A + B := [a_{ij} + b_{ij}]_{i=1, j=1}^m, n$ is the sum of A and B .
- (ii) The matrix $\lambda A := [\lambda a_{ij}]_{i=1, j=1}^m, n$ is a scalar multiple of A .
- (iii) The matrix $[0]_{i=1, j=1}^m, n$ is the $m \times n$ zero matrix, written as 0 .
- (iv) If $m = n$ (number of rows equals number of columns), then A is a square matrix.

The non-square matrices come in two kinds of shapes with intuitive names. We have the *tall matrices* with more rows than columns, and the *wide matrices* with more columns than rows; see Figure 2.1.

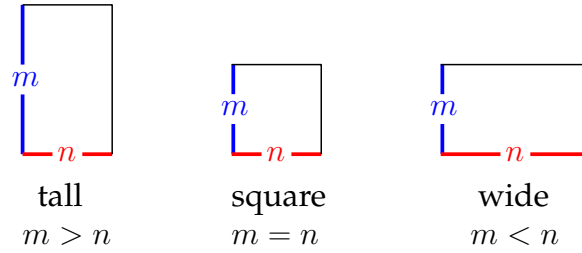


Figure 2.1: Matrix shapes

Square matrices are particularly important, and we often consider additional properties of them. Before we provide the general definitions, we give some 3×3 examples.

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

identity
matrix

$$\begin{bmatrix} 2 & 0 & 0 \\ 0 & 4 & 0 \\ 0 & 0 & 5 \end{bmatrix}$$

diagonal
matrix

$$\begin{bmatrix} 2 & 1 & 0 \\ 0 & 4 & 7 \\ 0 & 0 & 5 \end{bmatrix}$$

upper triangular
matrix

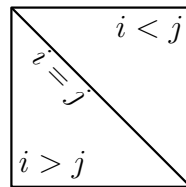
$$\begin{bmatrix} 2 & 0 & 0 \\ 1 & 4 & 0 \\ 0 & 7 & 5 \end{bmatrix}$$

lower triangular
matrix

$$\begin{bmatrix} 2 & 1 & 0 \\ 1 & 4 & 7 \\ 0 & 7 & 5 \end{bmatrix}$$

symmetric
matrix

Definition 2.3 (Square matrix classes). Let $A = [a_{ij}]_{i=1,j=1}^m$ be an $m \times m$ square matrix. If $i < j$ / $i = j$ / $i > j$, then a_{ij} is said to be above / on / below the diagonal.



- (i) If $a_{ii} = 1$ for all i (entries on the diagonal are 1) and $a_{ij} = 0$ for all $i \neq j$ (entries not on the diagonal are 0), then A is the **identity matrix**, denoted (in abuse of notation) by I for every m . A different way of defining I is as

$$I := [\delta_{ij}]_{i=1,j=1}^m.$$

Here, δ_{ij} is the Kronecker delta, defined as 1 if $i = j$ and 0 otherwise.

- (ii) If $a_{ij} = 0$ for all $i \neq j$ (entries not on the diagonal are 0), then A is a **diagonal matrix**.
- (iii) If $a_{ij} = 0$ for all $i > j$ (entries below the diagonal are 0), then A is an **upper triangular matrix**.

(iv) If $a_{ij} = 0$ for all $i < j$ (entries above the diagonal are 0), then A is a lower triangular matrix.

(v) If $a_{ij} = a_{ji}$ for all i, j , then A is a symmetric matrix.

Note that (ii)-(iv) each require that some entries are zero, but *not* that the other entries are nonzero. For example, the $m \times m$ zero matrix is at the same time diagonal, upper triangular, and lower triangular, even though we do not see a “diagonal”, or a “triangle” in it. Also whenever we say things like “for all i ”, we mean “for all applicable i ”, in this case $i \in [m]$.

2.1.1 Matrix-vector multiplication

Here is the efficient “matrix way” of writing down a linear combination as a *matrix-vector product*:

$$\underbrace{7 \begin{pmatrix} 1 \\ 3 \\ 5 \end{pmatrix} + 8 \begin{pmatrix} 2 \\ 4 \\ 6 \end{pmatrix}}_{\text{linear combination}} = \underbrace{\begin{bmatrix} 1 & 2 \\ 3 & 4 \\ 5 & 6 \end{bmatrix} \begin{pmatrix} 7 \\ 8 \end{pmatrix}}_{\text{matrix-vector product}} = \underbrace{\begin{pmatrix} 23 \\ 53 \\ 83 \end{pmatrix}}_{\text{result}}.$$

Definition 2.4 (Matrix-vector multiplication with A in column notation). *Let*

$$A = \begin{bmatrix} | & | & \cdots & | \\ \mathbf{v}_1 & \mathbf{v}_2 & \cdots & \mathbf{v}_n \\ | & | & & | \end{bmatrix} \in \mathbb{R}^{m \times n}, \quad \mathbf{x} = \begin{pmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{pmatrix} \in \mathbb{R}^n.$$

The vector

$$A\mathbf{x} := \sum_{j=1}^n x_j \mathbf{v}_j \in \mathbb{R}^m$$

is the product of A and \mathbf{x} .

This allows us to express linear combinations (Definition 1.4) and linear (in)dependence (according to Corollary 1.23 (ii)) in “matrix language”.

Observation 2.5. *Let A be an $m \times n$ matrix.*

- (i) *A vector $\mathbf{b} \in \mathbb{R}^m$ is a linear combination of the columns of A if and only if there is a vector $\mathbf{x} \in \mathbb{R}^n$ (of suitable scalars) such that $A\mathbf{x} = \mathbf{b}$.*
- (ii) *The columns of A are linearly independent if and only if $\mathbf{x} = \mathbf{0}$ is the only vector such that $A\mathbf{x} = \mathbf{0}$.*

We can also think of matrix-vector multiplication as transforming an “input vector” $\mathbf{x} \in \mathbb{R}^n$ into an “output vector” $A\mathbf{x} \in \mathbb{R}^m$, where the matrix A tells us how to do this transformation. We will explore this interpretation of the matrix-vector product in detail in Section 2.2, but it is useful to have it in mind already here. Pictorially, matrix-vector multiplication looks like this:

$$\begin{array}{c} A \quad \mathbf{x} \quad A\mathbf{x} \\ \begin{array}{|c|} \hline \text{ } \\ \hline \end{array} \begin{array}{|c|} \hline \text{ } \\ \hline \end{array} = \begin{array}{|c|} \hline \text{ } \\ \hline \end{array} \\ \begin{array}{c} m \\ n \end{array} \end{array}$$

It is also important to understand the product in the other matrix notations.

Observation 2.6 (Matrix-vector multiplication with A in table notation). *Let*

$$A = \begin{bmatrix} a_{11} & a_{12} & \cdots & a_{1n} \\ a_{21} & a_{22} & \cdots & a_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ a_{m1} & a_{m2} & \cdots & a_{mn} \end{bmatrix} = [a_{ij}]_{i=1, j=1}^m \in \mathbb{R}^{m \times n}, \quad \mathbf{x} = \begin{pmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{pmatrix} = (x_j)_{j=1}^n \in \mathbb{R}^n.$$

Then

$$A\mathbf{x} = \begin{pmatrix} a_{11}x_1 + a_{12}x_2 + \cdots + a_{1n}x_n \\ a_{21}x_1 + a_{22}x_2 + \cdots + a_{2n}x_n \\ \vdots \\ a_{m1}x_1 + a_{m2}x_2 + \cdots + a_{mn}x_n \end{pmatrix} = \left(\sum_{j=1}^n a_{ij}x_j \right)_{i=1}^m \in \mathbb{R}^m.$$

This is in many sources the official definition of matrix-vector multiplication. It does not explicitly refer to the columns or rows of A but just looks at A as a table. But we easily see that both definitions say the same, by annotating the columns in Observation 2.6:

$$A = \begin{array}{cccc} \begin{bmatrix} a_{11} & a_{12} & \cdots & a_{1n} \\ a_{21} & a_{22} & \cdots & a_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ a_{m1} & a_{m2} & \cdots & a_{mn} \end{bmatrix} & \mathbf{v}_1 & \mathbf{v}_2 & \cdots & \mathbf{v}_n \end{array}, \quad A\mathbf{x} = \begin{pmatrix} a_{11}x_1 + a_{12}x_2 + \cdots + a_{1n}x_n \\ a_{21}x_1 + a_{22}x_2 + \cdots + a_{2n}x_n \\ \vdots \\ a_{m1}x_1 + a_{m2}x_2 + \cdots + a_{mn}x_n \\ x_1\mathbf{v}_1 + x_2\mathbf{v}_2 + \cdots + x_n\mathbf{v}_n \end{pmatrix}.$$

An immediate consequence is the following.

Corollary 2.7 (Identity-vector multiplication). *Let I be the $m \times m$ identity matrix (Definition 2.3). Then $I\mathbf{x} = \mathbf{x}$ for all $\mathbf{x} \in \mathbb{R}^m$.*

Finally, we can also use row notation of A to define $A\mathbf{x}$ in terms of scalar products.

Observation 2.8 (Matrix-vector multiplication with A in row notation). *Let*

$$A = \begin{bmatrix} \text{---} & \mathbf{u}_1^\top & \text{---} \\ \text{---} & \mathbf{u}_2^\top & \text{---} \\ & \vdots & \\ \text{---} & \mathbf{u}_m^\top & \text{---} \end{bmatrix} \in \mathbb{R}^{m \times n}, \quad \mathbf{x} \in \mathbb{R}^n. \quad \text{Then } A\mathbf{x} = \underbrace{\begin{pmatrix} \mathbf{u}_1^\top \mathbf{x} \\ \mathbf{u}_2^\top \mathbf{x} \\ \vdots \\ \mathbf{u}_m^\top \mathbf{x} \end{pmatrix}}_{m \text{ scalar products}}.$$

This is seen to be correct by annotating the *rows* in Observation 2.6:

$$A = \begin{bmatrix} a_{11} & a_{12} & \cdots & a_{1n} \\ a_{21} & a_{22} & \cdots & a_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ a_{m1} & a_{m2} & \cdots & a_{mn} \end{bmatrix} \begin{matrix} \mathbf{u}_1^\top \\ \mathbf{u}_2^\top \\ \vdots \\ \mathbf{u}_m^\top \end{matrix}, \quad A\mathbf{x} = \begin{pmatrix} a_{11}x_1 + a_{12}x_2 + \cdots + a_{1n}x_n \\ a_{21}x_1 + a_{22}x_2 + \cdots + a_{2n}x_n \\ \vdots \\ a_{m1}x_1 + a_{m2}x_2 + \cdots + a_{mn}x_n \end{pmatrix} \begin{matrix} \mathbf{u}_1^\top \mathbf{x} \\ \mathbf{u}_2^\top \mathbf{x} \\ \vdots \\ \mathbf{u}_m^\top \mathbf{x} \end{matrix}.$$

Observation 2.8 formalizes the way in which many people do matrix-vector multiplication in practice: to get the i -th entry of the output vector $A\mathbf{x}$, multiply (scalar product) the i -th row of A with the input vector \mathbf{x} . Here is how this can be visualized for the initial example of this section:

$$\begin{array}{ccc} \begin{bmatrix} \mathbf{1} & \mathbf{2} \\ 3 & 4 \\ 5 & 6 \end{bmatrix} \begin{pmatrix} \mathbf{7} \\ \mathbf{8} \end{pmatrix} = \begin{pmatrix} \mathbf{23} \\ 53 \\ 83 \end{pmatrix} & \begin{bmatrix} 1 & 2 \\ \mathbf{3} & \mathbf{4} \\ 5 & 6 \end{bmatrix} \begin{pmatrix} 7 \\ \mathbf{8} \end{pmatrix} = \begin{pmatrix} 23 \\ \mathbf{53} \\ 83 \end{pmatrix} & \begin{bmatrix} 1 & 2 \\ 3 & 4 \\ \mathbf{5} & \mathbf{6} \end{bmatrix} \begin{pmatrix} 7 \\ 8 \end{pmatrix} = \begin{pmatrix} 23 \\ 53 \\ \mathbf{83} \end{pmatrix} \\ 1 \cdot 7 + 2 \cdot 8 = 23 & 3 \cdot 7 + 4 \cdot 8 = 53 & 5 \cdot 7 + 6 \cdot 8 = 83 \end{array}$$

2.1.2 Column space and rank

Definition 2.9 (Column space). *Let A be an $m \times n$ matrix. The column space $C(A)$ of A is the span (set of all linear combinations) of the columns,*

$$C(A) := \{A\mathbf{x} : \mathbf{x} \in \mathbb{R}^n\} \subseteq \mathbb{R}^m.$$

Here we use Definition 2.4 of the matrix-vector product $A\mathbf{x}$ as the linear combination of the columns with scalars from \mathbf{x} . When we consider all possible vectors $\mathbf{x} \in \mathbb{R}^n$, we obtain all possible linear combinations (the span) of the columns. Note that we always have $\mathbf{0} \in C(A)$ because $A\mathbf{0} = \mathbf{0}$.

Using the column space, the statement of Fact 1.5 can be written as

$$C\left(\begin{bmatrix} 2 & 3 \\ 3 & -1 \end{bmatrix}\right) = \text{Span}\left(\begin{pmatrix} 2 \\ 3 \end{pmatrix}, \begin{pmatrix} 3 \\ -1 \end{pmatrix}\right) = \mathbb{R}^2.$$

See Figure 1.23 (right) for an illustration. In general, whenever A is a 2×2 matrix with linearly independent columns, $C(A) = \mathbb{R}^2$ holds; see Challenge 1.6.

A crucial parameter of a matrix is its rank. The rank is defined as the number of *independent columns*. A column is called independent if it is not a linear combination of previous columns.

Definition 2.10 ((In)dependent column and (column) rank of a matrix). *Let*

$$A = \begin{bmatrix} | & | & \cdots & | \\ \mathbf{v}_1 & \mathbf{v}_2 & \cdots & \mathbf{v}_n \\ | & | & \cdots & | \end{bmatrix}$$

be an $m \times n$ matrix with columns $\mathbf{v}_1, \mathbf{v}_2, \dots, \mathbf{v}_n \in \mathbb{R}^m$. Column \mathbf{v}_j is called independent if \mathbf{v}_j is not a linear combination of $\mathbf{v}_1, \mathbf{v}_2, \dots, \mathbf{v}_{j-1}$. Otherwise, \mathbf{v}_j is called dependent. The rank of A , written as $\text{rank}(A)$, is the number of independent columns of A . The rank is sometimes also called the column rank to distinguish it more explicitly from the row rank, see Definition 2.14 below.

This means, $\text{rank}(A)$ is a number between 0 and n . We have $\text{rank}(A) = n$ exactly if no column is a linear combination of the previous ones. According to Corollary 1.23 (iii), this is the same as saying that the columns are linearly independent. The case $\text{rank}(A) = 0$ happens exactly if $A = 0$, the zero matrix. For $A = 0$, every (even the first) column is a linear combination of the previous ones, because $\mathbf{0}$ is a linear combination of the empty sequence of vectors; see Section 1.1.5.

Figure 2.2 provides two more examples.

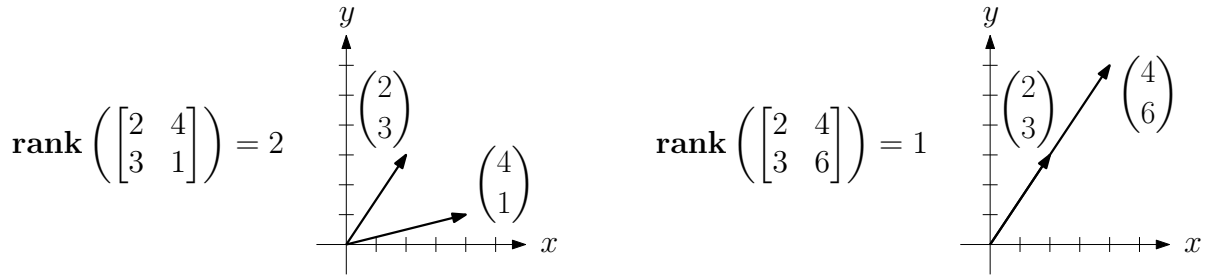


Figure 2.2: Ranks of two 2×2 matrices: 2 when both columns are independent (left), or 1 when only the first column is independent (right)

If we reorder the columns of a matrix, we may get other independent columns. For example, the two matrices

$$\begin{bmatrix} 2 & 4 \\ 3 & 6 \end{bmatrix} \text{ and } \begin{bmatrix} 4 & 2 \\ 6 & 3 \end{bmatrix}$$

have the same columns, but in different order. Each of the two matrices has one independent column, namely its first one, but these are different. Still, both matrices have the same rank 1. This is not a coincidence. We will take this up again in the example immediately preceding Section 4.2.3; a consequence of the results in Section 4.2.3 is that the rank of a matrix does not change when we reorder the columns.

The independent columns of a matrix A are also of interest, because they already span the column space of A .

Lemma 2.11 (The independent columns span the column space). *Let A be an $m \times n$ matrix with r independent columns, and let C be the $m \times r$ submatrix containing the independent columns. Then $\mathbf{C}(A) = \mathbf{C}(C)$.*

By a *submatrix* of a matrix A , we mean a matrix obtained from A by selecting some rows and/or columns. Here, C is obtained from A by selecting the independent columns.

Proof. Let $\mathbf{u}_1, \mathbf{u}_2, \dots, \mathbf{u}_r$ be the independent columns of A , and $\mathbf{w}_1, \mathbf{w}_2, \dots, \mathbf{w}_{n-r}$ the dependent columns (in the same order as they appear in A). We will prove that

$$\underbrace{\text{Span}(\mathbf{u}_1, \mathbf{u}_2, \dots, \mathbf{u}_r, \mathbf{w}_1, \mathbf{w}_2, \dots, \mathbf{w}_{n-r})}_{\mathbf{C}(A)} = \underbrace{\text{Span}(\mathbf{u}_1, \mathbf{u}_2, \dots, \mathbf{u}_r)}_{\mathbf{C}(C)}.$$

We first observe that \mathbf{w}_j is a linear combination of $\mathbf{u}_1, \mathbf{u}_2, \dots, \mathbf{u}_r, \mathbf{w}_1, \mathbf{w}_2, \dots, \mathbf{w}_{j-1}$, for all j . Indeed, by Definition 2.10, a dependent column is a linear combination of the previous columns in A , and the sequence $\mathbf{u}_1, \mathbf{u}_2, \dots, \mathbf{u}_r, \mathbf{w}_1, \mathbf{w}_2, \dots, \mathbf{w}_{j-1}$ contains all those (and possibly a few extra independent ones). Hence, if we start from the sequence $\mathbf{u}_1, \mathbf{u}_2, \dots, \mathbf{u}_r$ and then add $\mathbf{w}_1, \mathbf{w}_2, \dots, \mathbf{w}_{n-r}$ one by one, Lemma 1.26 guarantees that the span of the sequence never changes. \square

2.1.3 Row space and transpose

Recall that a matrix also represents a sequence of covectors, namely its rows. So we can also define the row space $\mathbf{R}(A)$ of a matrix: the span of its rows. We have not officially defined the span of a sequence of covectors, and we are not planning to do so. It would be possible to make “covector copies” of all our vector definitions (or silently apply the vector versions to covectors, via abuse of notation). But this is not necessary. Instead, for the definition of the row space, we treat the rows as columns (of another matrix) and not as covectors. This is more practical, and Definition 2.14 of row space below does it in a clean way. Figure 2.3 illustrates the row spaces of the matrices from Figure 2.2. But we still think about the row space of a matrix as being spanned by the rows.

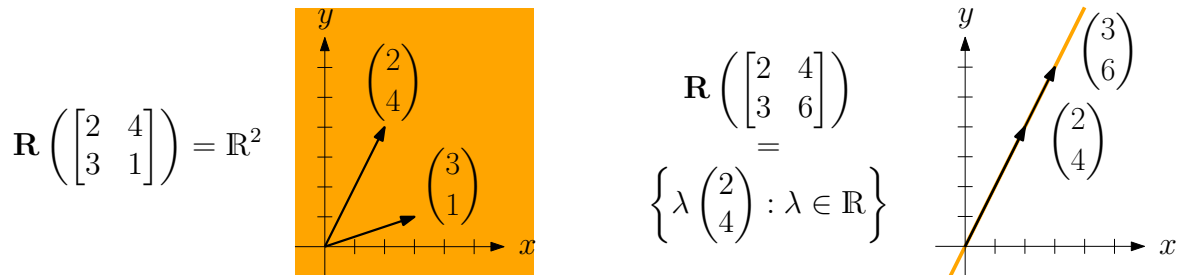


Figure 2.3: Row spaces of two 2×2 matrices: the whole plane (left) when the rows are linearly independent, or a line (right) when the rows are linearly dependent

For both matrices, the number of independent columns (rank) equals the number of independent rows (row rank). Is this a coincidence? No! It turns out that this is true for *every* matrix. This is quite surprising, and we will prove it in Section 4.3.2.

What we will do here is prepare the ground. We are still missing formal definitions of row space, independent row, and row rank of a matrix A . For this, we express the rows (covectors) of A as the columns (vectors) of another matrix A^\top , and then define the row space of A as the column space of A^\top , an independent row of A as an independent column of A^\top , and the row rank of A as the (column) rank of A^\top .

This needs the concept of *matrix transposition*. The transpose A^\top of a matrix A is another matrix, obtained by “mirroring” A along the *diagonal* “ \backslash ”, the line going through the diagonal entries a_{11}, a_{22}, \dots of A . Figure 2.4 shows a physical such mirror image and its mathematical abstraction where we do not screw up the number and bracket symbols. The effect of this mirroring is that the rows of A become the columns of A^\top .

Figure 2.4: Mirroring a matrix along the diagonal, physically and mathematically

Definition 2.12 (Transpose). Let $A = [a_{ij}]_{i=1,j=1}^{m,n}$ be an $m \times n$ matrix. The transpose of A is the $n \times m$ matrix

$$A^\top := B = [b_{ij}]_{i=1,j=1}^{n,m},$$

where $b_{ij} = a_{ji}$ for all i, j .

This means, the entry of A^\top in row i and column j is a_{ji} , the entry of A in row j and column i . Transposing a matrix thus interchanges columns with rows. Previously, we have transposed vectors to turn them into covectors and vice versa, see Section 1.2.6. For example,

$$\underbrace{\begin{pmatrix} 1 \\ 3 \\ 5 \end{pmatrix}}_{\text{vector}}^\top = \underbrace{(1 \quad 3 \quad 5)}_{\text{covector}}.$$

With matrices of the corresponding dimensions, the same happens:

$$\underbrace{\begin{bmatrix} 1 \\ 3 \\ 5 \end{bmatrix}}_{3 \times 1 \text{ matrix}}^\top = \underbrace{\begin{bmatrix} 1 & 3 & 5 \end{bmatrix}}_{1 \times 3 \text{ matrix}}.$$

This motivates the use of the same transposition symbol $^\top$ in both cases.

In column and row notation, we have

$$A = \begin{bmatrix} | & | & & | \\ \mathbf{v}_1 & \mathbf{v}_2 & \cdots & \mathbf{v}_n \\ | & | & & | \end{bmatrix} \Leftrightarrow A^\top = \begin{bmatrix} - & \mathbf{v}_1^\top & - \\ - & \mathbf{v}_2^\top & - \\ & \vdots & \\ - & \mathbf{v}_n^\top & - \end{bmatrix},$$

$$A = \begin{bmatrix} - & \mathbf{u}_1^\top & - \\ - & \mathbf{u}_2^\top & - \\ & \vdots & \\ - & \mathbf{u}_m^\top & - \end{bmatrix} \Leftrightarrow A^\top = \begin{bmatrix} | & | & \cdots & | \\ \mathbf{u}_1 & \mathbf{u}_2 & \cdots & \mathbf{u}_m \\ | & | & & | \end{bmatrix}.$$

It is easy to see that mirroring twice gives back the original. Also, the symmetric matrices are exactly the ones that are mirror images of themselves:

Observation 2.13 (Transposing twice, and transposing symmetric matrices). *Let A be an $m \times n$ matrix. Then*

$$(A^\top)^\top = A.$$

Moreover, a square matrix A is symmetric (Definition 2.3) if and only if $A = A^\top$.

Now we can define the row space of A simply as the column space of A^\top . Definitions of independent row and row rank are done in the same way.

Definition 2.14 (Row space, independent row, row rank). *Let A be an $m \times n$ matrix. The row space $\mathbf{R}(A)$ of A is the column space of the transpose,*

$$\mathbf{R}(A) := \mathbf{C}(A^\top).$$

For every i , the i -th row of A is called independent if the i -th column of A^\top is independent. The row rank of A is the (column) rank of A^\top .

Note that by this definition, the row space is a set of vectors, not of covectors. While $\mathbf{C}(A) \subseteq \mathbb{R}^m$, we have $\mathbf{R}(A) \subseteq \mathbb{R}^n$.

2.1.4 Rank-1 matrices

The statement that the number of independent columns of a matrix equals the number of independent rows can via Definition 2.14 be expressed as $\text{rank}(A) = \text{rank}(A^\top)$. Indeed, $\text{rank}(A)$ counts the independent columns of A , while $\text{rank}(A^\top)$ counts the independent columns of A^\top and therefore the independent rows of A .

Here, we will prove $\text{rank}(A) = \text{rank}(A^\top)$ for the case where A has rank 1 (the general case is handled in Theorem 4.33). The rank-1 result will be an easy consequence of the

following lemma that tells us how rank-1 matrices look like. Before that, let us consider an example of a rank-1-matrix:

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

This matrix has rank 1, since there is only one independent column (the first one), and the second and third are scalar multiples of it. In this example, there is also only one independent row (the first one), and the second one is a scalar multiple of it.

Lemma 2.15 (Rank-1 matrices). *Let A be an $m \times n$ matrix. The following two statements are equivalent.*

(i) $\text{rank}(A) = 1$.

(ii) There are nonzero vectors $\mathbf{v} \in \mathbb{R}^m$, $\mathbf{w} \in \mathbb{R}^n$ such that

$$A = [v_i w_j]_{i=1, j=1}^m n.$$

In dot notation, the rank-1 matrices are therefore exactly the nonzero matrices of the form

$$\begin{bmatrix} v_1 w_1 & v_1 w_2 & \cdots & v_1 w_n \\ v_2 w_1 & v_2 w_2 & \cdots & v_2 w_n \\ \vdots & \vdots & \ddots & \vdots \\ v_m w_1 & v_m w_2 & \cdots & v_m w_n \end{bmatrix}.$$

Proof. (i) \Rightarrow (ii): If $\text{rank}(A) = 1$, there is exactly one independent column $\mathbf{v} \neq \mathbf{0}$ (Definition 2.10). This means that all columns before \mathbf{v} are $\mathbf{0}$, and all columns after \mathbf{v} are scalar multiples of \mathbf{v} . Thus, all columns are scalar multiples of \mathbf{v} where at least one scalar (the one for column \mathbf{v} itself) is nonzero. Let $\mathbf{w} \neq \mathbf{0}$ be the vector of scalars, meaning that the j -th column of A is $w_j \mathbf{v}$. Hence, a_{ij} (the entry in row i and column j of A) equals $w_j v_i = v_i w_j$. In other words,

$$A = [v_i w_j]_{i=1, j=1}^m n.$$

(ii) \Rightarrow (i): If $A = [v_i w_j]_{i=1, j=1}^m n$ for nonzero vectors \mathbf{v}, \mathbf{w} , then column j of A is $w_j \mathbf{v}$. The first column for which $w_j \neq 0$ is independent, since $\mathbf{v} \neq \mathbf{0}$ and all columns before it are $\mathbf{0}$; all columns after it are scalar multiples of this independent column (the scalar for column $k > j$ is w_k/w_j). Hence, there is exactly one independent column, so $\text{rank}(A) = 1$. \square

Corollary 2.16 (Transpose of a rank-1 matrix). *Let A be an $m \times n$ matrix with $\text{rank}(A) = 1$. Then also $\text{rank}(A^\top) = 1$.*

Proof. By Lemma 2.15 (i) \Rightarrow (ii), there are nonzero vectors $\mathbf{v} \in \mathbb{R}^m$, $\mathbf{w} \in \mathbb{R}^n$ such that

$$A = \underbrace{[v_i w_j]_{i=1, j=1}^m n}_{a_{ij}}.$$

By Definition 2.12 of the transpose,

$$A^T = \underbrace{[v_j w_i]_{i=1, j=1}^n_m}_{a_{ji}} = [w_i v_j]_{i=1, j=1}^n_m,$$

with nonzero $w \in \mathbb{R}^n$, $v \in \mathbb{R}^m$. Using Lemma 2.15 (ii) \Rightarrow (i) for A^T gives $\text{rank}(A^T) = 1$. \square

2.1.5 Nullspace

After column and row space, here is the third fundamental space associated with a matrix.

Definition 2.17 (Nullspace). *Let A be an $m \times n$ matrix. The nullspace of A is the set*

$$\mathbf{N}(A) = \{\mathbf{x} \in \mathbb{R}^n : A\mathbf{x} = \mathbf{0}\} \subseteq \mathbb{R}^n.$$

Coming back to the interpretation of A as transforming “input vector” \mathbf{x} to “output vector” $A\mathbf{x}$, we can say that the nullspace contains all input vectors that lead to output vector $\mathbf{0}$. We always have $\mathbf{0} \in \mathbf{N}(A)$, and sometimes this is the only vector in $\mathbf{N}(A)$. We in fact understand the situation precisely: by Observation 2.5 (ii), the matrices with a “trivial” nullspace (only containing $\mathbf{0}$) are exactly the ones with linearly independent columns. At the other extreme, if $A = \mathbf{0}$ (the $m \times n$ zero matrix), then $\mathbf{N}(A) = \mathbb{R}^n$, because every input leads to output $\mathbf{0}$.

Let us also look at the nullspace of the rank-1 matrix whose column and row spaces we have previously examined in Figures 1.23 (left) and Figure 2.3 (right). This is the set

$$\mathbf{N}\left(\begin{bmatrix} 2 & 4 \\ 3 & 6 \end{bmatrix}\right) = \left\{ \begin{pmatrix} x \\ y \end{pmatrix} \in \mathbb{R}^2 : \begin{bmatrix} 2 & 4 \\ 3 & 6 \end{bmatrix} \begin{pmatrix} x \\ y \end{pmatrix} = \mathbf{0} \right\}.$$

By Definition 2.4 of matrix-vector multiplication, the vector equation defining the nullspace can be written as the two “normal” equations $2x + 4y = 0$ and $3x + 6y = 0$. These two equations actually say the same (you get one equation from the other by multiplication with a factor of $3/2$ or $2/3$). So we can ignore the second one, and simplify the first one to get $y = -x/2$. This is the equation of a line through the origin; see Figure 2.5.

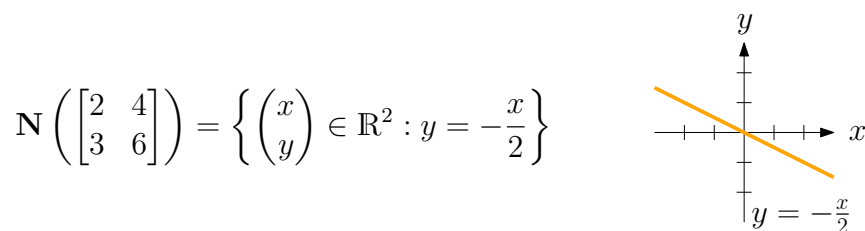


Figure 2.5: Nullspace of a 2×2 matrix of rank 1: a line

2.2 Matrices and linear transformations

When we multiply an $m \times n$ matrix A with a vector \mathbf{x} in \mathbb{R}^n , we get a “transformed” vector $A\mathbf{x} \in \mathbb{R}^m$. Here, we look at the properties of and the theory behind such matrix transformations. They are for example used to draw 3-dimensional objects in 2-dimensional space, and they have many other applications. The main insight is that matrix transformations are the same objects as linear transformations. The latter are not defined via matrices, but via a condition that we call *linearity*.

2.2.1 Matrix transformations

An $m \times n$ matrix A defines a function that “transforms” an input vector $\mathbf{x} \in \mathbb{R}^n$ into an output vector $A\mathbf{x} \in \mathbb{R}^m$. In the context of functions, we would normally say that \mathbf{x} is mapped to $A\mathbf{x}$; the term “transformation” comes from a geometric viewpoint where we think of the output as a transformed (for example rotated) version of the input. We will explore the geometric viewpoint in some more detail below.

Definition 2.18 (Matrix transformation). *Let A be an $m \times n$ matrix. The function $T_A : \mathbb{R}^n \rightarrow \mathbb{R}^m$ defined by*

$$T_A : \mathbf{x} \mapsto A\mathbf{x}$$

is the matrix transformation with matrix A .

For example, if

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

then

$$T_A : \begin{pmatrix} x_1 \\ x_2 \end{pmatrix} \mapsto \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix} \begin{pmatrix} x_1 \\ x_2 \end{pmatrix} = \begin{pmatrix} x_2 \\ x_1 \end{pmatrix},$$

the function that swaps the two coordinates of its 2-dimensional input vector. A does not have to be a square matrix. For example, if $A = \begin{bmatrix} 1 & 1 \end{bmatrix}$ (a 1×2 matrix), then

$$T_A : \begin{pmatrix} x_1 \\ x_2 \end{pmatrix} \mapsto \begin{bmatrix} 1 & 1 \end{bmatrix} \begin{pmatrix} x_1 \\ x_2 \end{pmatrix} = (x_1 + x_2) \in \mathbb{R}^1.$$

The following lemma looks a bit technical at first sight but has a natural interpretation.

Lemma 2.19 (Linearity of matrix transformations). *Let A be an $m \times n$ matrix, $\mathbf{x}_1, \mathbf{x}_2 \in \mathbb{R}^n$ and $\lambda_1, \lambda_2 \in \mathbb{R}$. Then*

$$A(\lambda_1 \mathbf{x}_1 + \lambda_2 \mathbf{x}_2) = \lambda_1 A\mathbf{x}_1 + \lambda_2 A\mathbf{x}_2.$$

This says the following: taking the linear combination $\lambda_1 \mathbf{x}_1 + \lambda_2 \mathbf{x}_2$ and then applying T_A to get the output $A(\lambda_1 \mathbf{x}_1 + \lambda_2 \mathbf{x}_2)$ gives the same result as applying T_A to both vectors individually and then taking the linear combination $\lambda_1 A\mathbf{x}_1 + \lambda_2 A\mathbf{x}_2$ of the individual

outputs. We can visualize this with a *commutative diagram*:

$$\begin{array}{ccc}
 & \text{linear combination} & \\
 \mathbf{x}_1, \mathbf{x}_2 & \longrightarrow & \lambda_1 \mathbf{x}_1 + \lambda_2 \mathbf{x}_2 \\
 T_A \downarrow & & \downarrow T_A \\
 A\mathbf{x}_1, A\mathbf{x}_2 & \longrightarrow & \lambda_1 A\mathbf{x}_1 + \lambda_2 A\mathbf{x}_2 = A(\lambda_1 \mathbf{x}_1 + \lambda_2 \mathbf{x}_2) \\
 & \text{linear combination} &
 \end{array}$$

We also say that the diagram commutes (although it does not do anything by itself). The arrows in a commutative diagram correspond to certain operations, and we can follow any path through the diagram (apply the operations in any order), and the result is always the same.

Proof of Lemma 2.19. A natural approach is to first prove the following two simpler statements that deal with vector addition and scalar multiplication separately: for all $\mathbf{x}, \mathbf{x}' \in \mathbb{R}^n$ and all $\lambda \in \mathbb{R}$,

- (i) $A(\mathbf{x} + \mathbf{x}') = A(\mathbf{x}) + A(\mathbf{x}')$, and
- (ii) $A(\lambda \mathbf{x}) = \lambda A(\mathbf{x})$.

These equalities follow quite directly from the rules of vector addition (Definition 1.2), scalar multiplication (Definition 1.3) and matrix-vector multiplication (Definition 2.4); we omit the easy calculations. Combining (i) and (ii), we then get linearity:

$$A(\lambda_1 \mathbf{x}_1 + \lambda_2 \mathbf{x}_2) \stackrel{(i)}{=} A(\lambda_1 \mathbf{x}_1) + A(\lambda_2 \mathbf{x}_2) \stackrel{(ii)}{=} \lambda_1 A(\mathbf{x}_1) + \lambda_2 A(\mathbf{x}_2).$$

□

Vice versa, from linearity we also get (i) and (ii) (think about how, or skip ahead to the proof of Lemma 2.23), so linearity is in fact equivalent to the combination of (i) and (ii). Some sources refer to this combination as linearity.

The geometry of matrix transformations. To understand what a matrix transformation T_A “does”, it is useful to not only look at individual inputs, but also at a whole set of inputs. For a set of input vectors $X \subseteq \mathbb{R}^n$, we define $A(X) := \{A\mathbf{x} : \mathbf{x} \in X\} \subseteq \mathbb{R}^m$ as the set of transformed output vectors. Note that $A(\mathbb{R}^n) = \mathbf{C}(A)$, the column space of A ; see Definition 2.9.

In the following examples, we see how different A ’s transform the standard unit vectors $\mathbf{e}_1, \mathbf{e}_2$, and a set $X \subseteq \mathbb{R}^2$ (gray L-shaped *polygon*); see Figure 2.6 (middle).

Figure 2.6 (right) corresponds to our first example above where A swaps the two coordinates of the input vector. The geometric effect is that the input is mirrored along the diagonal \swarrow of the coordinate system.

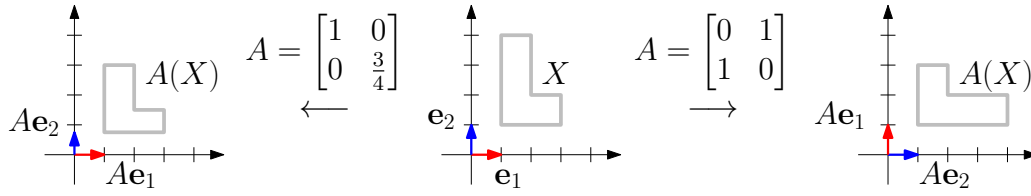


Figure 2.6: Middle: The input; Right: Mirroring the input along the diagonal. Left: Stretching the input by a factor of $\frac{3}{4}$ along the second coordinate.

Figure 2.6 (left) gives an example of *stretching*, which means to make the input longer or shorter along some or all of the coordinates. If the *stretching factors* are the same for all coordinates, we have a *scaling*, resulting in a larger or smaller copy of the input.

Let us investigate this in some more detail. We first observe that in both examples of Figure 2.6, the output vector Ae_1 is the first column of A , while output vector Ae_2 is the second column of A . This is not a coincidence. Generally, multiplying a matrix with the j -th standard unit vector (that has a 1 at index j and 0's everywhere else) gives us the j -th column of the matrix; this is an immediate consequence of Definition 2.4 (matrix-vector multiplication in column notation).

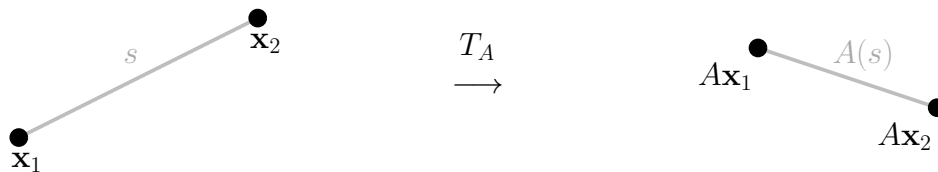
The polygon X to which we apply A in Figure 2.6 has 6 corners and 6 line segments connecting the corners. This is an infinite set, but to compute $A(X)$, we only need to apply A to the corners; each line segment will “follow” its two corners. This is a consequence of linearity: consider a line segment s connecting two corners \mathbf{x}_1 and \mathbf{x}_2 . In Section 1.1.4, we have seen that s is the set of *convex combinations* of \mathbf{x}_1 and \mathbf{x}_2 ,

$$s = \{\lambda_1 \mathbf{x}_1 + \lambda_2 \mathbf{x}_2 : \lambda_1 + \lambda_2 = 1, \lambda_1 \geq 0, \lambda_2 \geq 0\}.$$

Hence,

$$\begin{aligned} A(s) &= \{A(\lambda_1 \mathbf{x}_1 + \lambda_2 \mathbf{x}_2) : \lambda_1 + \lambda_2 = 1, \lambda_1 \geq 0, \lambda_2 \geq 0\} \\ &= \{\lambda_1 A\mathbf{x}_1 + \lambda_2 A\mathbf{x}_2 : \lambda_1 + \lambda_2 = 1, \lambda_1 \geq 0, \lambda_2 \geq 0\} \quad (\text{linearity, Lemma 2.19}). \end{aligned}$$

This means that the transformed line segment $A(s)$ is simply the line segment connecting the transformed corners $A\mathbf{x}_1$ and $A\mathbf{x}_2$:



If $A = I$, the identity matrix, then the matrix transformation T_A simply outputs the input, without transforming it (due to $I\mathbf{x} = \mathbf{x}$ for all \mathbf{x} ; see Corollary 2.7). Figure 2.7 shows two more examples of matrix transformations, a *shear*, and a *rotation*.

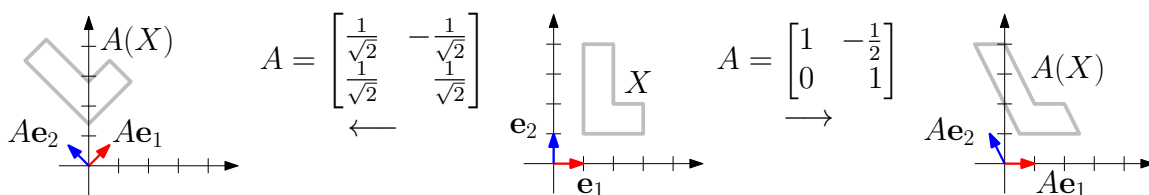


Figure 2.7: Middle: The input; Right: Shearing the input parallel to the first coordinate. Left: Rotating the input by 45 degrees.

If A is a 2×3 matrix, then T_A is a *projection* from \mathbb{R}^3 to \mathbb{R}^2 . Such projections can be used to draw 3-dimensional objects in 2-dimensional space, for example the cube in the margin.



You can recognize the cube, although what you really see is just a 2-dimensional image, a *parallel projection*. Figure 2.8 shows how such a parallel projection is obtained through a matrix transformation T_A .

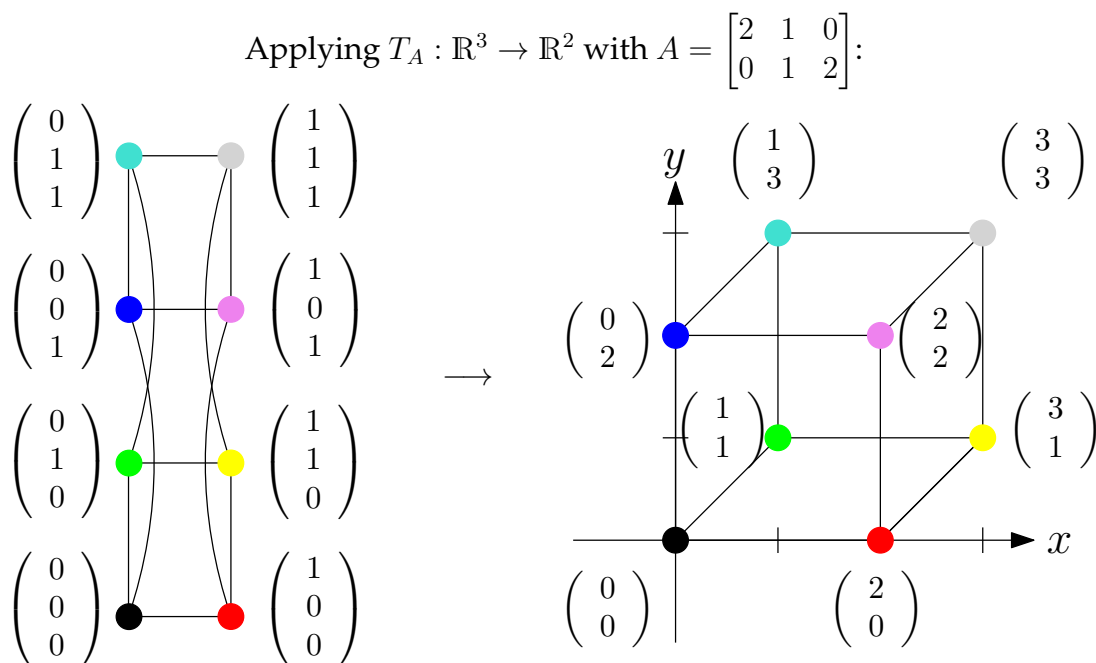


Figure 2.8: Parallel projection of a 3-dimensional cube. The left figure shows the 8 corners of the 3-dimensional *unit cube* as vectors in \mathbb{R}^3 . Two corners are connected in the cube if they differ in exactly one coordinate. The right figure is a 2-dimensional drawing, resulting from applying T_A to the cube corners (the connections follow the corners). With different matrices A , we get different drawings; see Figure 2.9 for another drawing.

If you take a photograph of a real 3-dimensional cube, you will also get a 2-dimensional image of it; this one will not be a parallel projection, though, but a *perspective projection*.

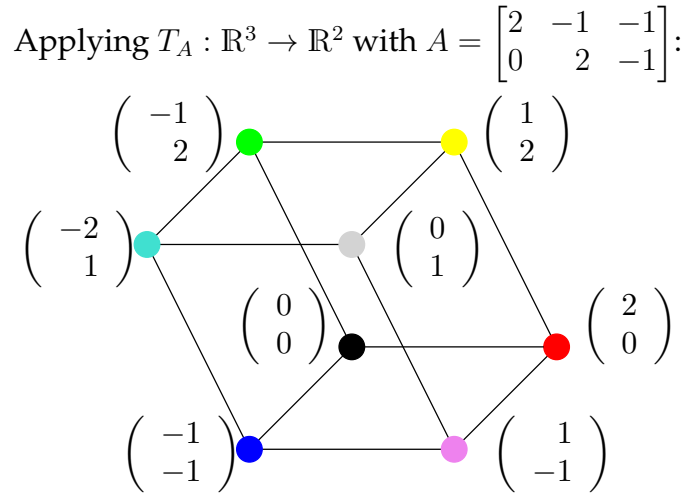


Figure 2.9: Projecting the cube in Figure 2.8 (left) using a different matrix

Figure 2.10 shows such an image. There is no matrix transformation that can realize this projection. The reason is that a matrix transformation T_A has the following property: if two line segments s, s' (for example two vertical edges of the original 3-dimensional cube) are parallel, then the transformed line segments $A(s)$ and $A(s')$ are also parallel. Exercise 2.20 asks you to prove this. But in Figure 2.10, parallel cube edges get transformed to edges that are not parallel. For example, look at the three “vertical” edges in Figure 2.10. Your brain may tell you that they are parallel, because it “sees” the original cube, but in the actual image, they are definitely not parallel.



Figure 2.10: Perspective projection of a cube as obtained through a photograph

Exercise 2.20. Let $s \subseteq \mathbb{R}^n$ be a line segment with endpoints $\mathbf{x}_1, \mathbf{x}_2 \in \mathbb{R}^n$ and $s' \subseteq \mathbb{R}^n$ a line segment with endpoints $\mathbf{x}'_1, \mathbf{x}'_2 \in \mathbb{R}^n$. Let A be an $m \times n$ matrix.

On page 60, we have argued (and you can use this here) that the transformed line segment $A(s) \subseteq \mathbb{R}^m$ has endpoints $A\mathbf{x}_1, A\mathbf{x}_2 \in \mathbb{R}^m$, and the transformed line segment $A(s') \subseteq \mathbb{R}^m$ has endpoints $A\mathbf{x}'_1, A\mathbf{x}'_2 \in \mathbb{R}^m$.

We call two line segments s (with endpoints $\mathbf{x}_1, \mathbf{x}_2$) and s' (with endpoints $\mathbf{x}'_1, \mathbf{x}'_2$) parallel if both $\mathbf{x}_1 - \mathbf{x}_2$ and $\mathbf{x}'_1 - \mathbf{x}'_2$ are scalar multiples of some vector \mathbf{v} , i.e. $\mathbf{x}_1 - \mathbf{x}_2 = \lambda\mathbf{v}$ and $\mathbf{x}'_1 - \mathbf{x}'_2 = \lambda'\mathbf{v}$ for some real numbers λ, λ' .

Prove the following statement: if the line segments s, s' are parallel, then the transformed line segments $A(s)$ and $A(s')$ are also parallel.

2.2.2 Linear transformations and linear functionals

Recall that a matrix transformation satisfies linearity (transformation commutes with linear combination, see Lemma 2.19). When we look at *all* the functions $T : \mathbb{R}^n \rightarrow \mathbb{R}^m$ that satisfy linearity, we arrive at the class of linear transformations. A function $T : \mathbb{R}^n \rightarrow \mathbb{R}$ satisfying linearity is called a linear functional.

Definition 2.21 (Linear transformation, linear functional). *A function $T : \mathbb{R}^n \rightarrow \mathbb{R}^m$ / $T : \mathbb{R}^n \rightarrow \mathbb{R}$ is called a linear transformation / linear functional if the following linearity axiom holds for all $\mathbf{x}_1, \mathbf{x}_2 \in \mathbb{R}^n$ and all $\lambda_1, \lambda_2 \in \mathbb{R}$.*

$$T(\lambda_1\mathbf{x}_1 + \lambda_2\mathbf{x}_2) = \lambda_1T(\mathbf{x}_1) + \lambda_2T(\mathbf{x}_2).$$

An axiom is a *defining property* of a class of mathematical objects. Exactly the objects satisfying the property belong to the class.

Since $T_A(\mathbf{x}) = A\mathbf{x}$ for a matrix transformation, the following is an immediate consequence of Lemma 2.19.

Observation 2.22. *Every matrix transformation is a linear transformation.*

The following provides an alternative definition of linearity in terms of *two* linearity axioms. For matrix transformations, we have already touched upon this in the proof of Lemma 2.19. This alternative definition is in some sources the main definition.

Lemma 2.23. *A function $T : \mathbb{R}^n \rightarrow \mathbb{R}^m$ / $T : \mathbb{R}^n \rightarrow \mathbb{R}$ is a linear transformation / linear functional if and only if the following two linearity axioms hold for all $\mathbf{x}, \mathbf{x}' \in \mathbb{R}^n$ and all $\lambda \in \mathbb{R}$.*

$$(i) \ T(\mathbf{x} + \mathbf{x}') = T(\mathbf{x}) + T(\mathbf{x}'), \text{ and}$$

$$(ii) \ T(\lambda\mathbf{x}) = \lambda T(\mathbf{x}).$$

It is quite common that a class of objects is being defined via axioms in different ways, where each way serves its own purpose. For example, if we want to prove that some function is a linear transformation or a linear functional, the two simpler linearity axioms (i) and (ii) in Lemma 2.23 are usually easier to check than the more “complicated” single linearity axiom in Definition 2.21. On the other hand, if we want to apply linearity to prove something else, the single axiom that combines (i) and (ii) may lead to shorter proofs.

Proof of Lemma 2.23. If we have linearity according to Definition 2.21, we can apply it to suitable vectors and scalars in order to derive the two simpler linearity axioms (i) and (ii): With $\mathbf{x}_1 = \mathbf{x}$, $\mathbf{x}_2 = \mathbf{x}'$, $\lambda_1 = \lambda_2 = 1$, we get (i). Using $\mathbf{x}_1 = \mathbf{x}$, $\mathbf{x}_2 = \mathbf{0}$, $\lambda_1 = \lambda$, $\lambda_2 = 0$, we obtain (ii). Vice versa, if (i) and (ii) hold, we can derive linearity according to Definition 2.21:

$$T(\lambda_1 \mathbf{x}_1 + \lambda_2 \mathbf{x}_2) \stackrel{(i)}{=} T(\lambda_1 \mathbf{x}_1) + T(\lambda_2 \mathbf{x}_2) \stackrel{(ii)}{=} \lambda_1 T(\mathbf{x}_1) + \lambda_2 T(\mathbf{x}_2).$$

□

Let us look at a few examples. Let $\mathbf{v} \in \mathbb{R}^m$ be a vector. The function $T : \mathbb{R}^m \rightarrow \mathbb{R}$,

$$T : \mathbf{x} \mapsto \sum_{i=1}^m v_i x_i$$

is a linear functional (also known as the covector \mathbf{v}^\top , see Definition 1.20). It is often a bit clearer to verify the two simple linearity axioms (i) and (ii) in Lemma 2.23. Here, this is done as follows:

$$\begin{aligned} T(\mathbf{x} + \mathbf{x}') &= \sum_{i=1}^m v_i (x_i + x'_i) = \sum_{i=1}^m v_i x_i + \sum_{i=1}^m v_i x'_i = T(\mathbf{x}) + T(\mathbf{x}'), \text{ and} \\ T(\lambda \mathbf{x}) &= \sum_{i=1}^m v_i \lambda x_i = \lambda \sum_{i=1}^m v_i x_i = \lambda T(\mathbf{x}). \end{aligned}$$

In these derivations, we use well-known properties (commutativity, distributivity) of addition and multiplication of real numbers, as well as Definition 1.2 of vector addition and Definition 1.3 of scalar multiplication (in the first equality of each line).

The function $T : \mathbb{R}^3 \rightarrow \mathbb{R}^2$,

$$T : \mathbf{x} \mapsto \begin{pmatrix} x_1 + 2x_2 \\ 3x_2 + 4x_3 \end{pmatrix}$$

is a linear transformation. We could again check this directly, but a more elegant proof is to observe that T is actually a matrix transformation and therefore a linear transformation by Theorem 2.26. Indeed, it is easy to see that $T = T_A$ for

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

Another (somewhat trivial but still instructive) example of a matrix transformation is $T : \mathbb{R}^n \rightarrow \mathbb{R}^m$, $\mathbf{x} \mapsto \mathbf{0}$. Here, $\mathbf{0}$ is the m -dimensional zero vector; linearity obviously holds, and we can also express T as the matrix transformation T_A with $A = \mathbf{0} \in \mathbb{R}^{m \times n}$. As our last example, we consider the *identity* $T : \mathbb{R}^m \rightarrow \mathbb{R}^m$, $\mathbf{x} \mapsto \mathbf{x}$. This is also a linear transformation and equal to the matrix transformation T_I where I is the $m \times m$ identity matrix; see Corollary 2.7.

Now for some counterexamples. Consider $T : \mathbb{R}^m \rightarrow \mathbb{R}$,

$$T : \mathbf{x} \mapsto \sum_{i=1}^m |x_i| = \|\mathbf{x}\|_1.$$

This outputs the 1-norm of \mathbf{x} (see Section 1.2.2) but is not a linear functional. To show this, we need to find a violation of linearity. For example, if $\mathbf{x} \neq \mathbf{0}$ and $\lambda < 0$, then $T(\lambda\mathbf{x}) > 0$ but $\lambda T(\mathbf{x}) < 0$; so linearity (ii), $T(\lambda\mathbf{x}) = \lambda T(\mathbf{x})$ in Lemma 2.23, does not always hold.

Slightly changing the third example above, we can produce another counterexample. Consider $T : \mathbf{x} \mapsto \mathbf{v}$, where $\mathbf{v} \in \mathbb{R}^m$ is some fixed nonzero vector. This is not a linear transformation, because linearity (i), $T(\mathbf{x} + \mathbf{x}') = T(\mathbf{x}) + T(\mathbf{x}')$ in Lemma 2.23, always fails: we have $T(\mathbf{x} + \mathbf{x}') = \mathbf{v}$ and $T(\mathbf{x}) + T(\mathbf{x}') = 2\mathbf{v} \neq \mathbf{v}$.

In this counterexample, we have $T(\mathbf{0}) \neq \mathbf{0}$, and this is already enough to conclude that T is not a linear transformation. Indeed, we have

Lemma 2.24. *Let $T : \mathbb{R}^n \rightarrow \mathbb{R}^m$ / $T : \mathbb{R}^n \rightarrow \mathbb{R}$ be a linear transformation / linear functional. Then $T(\mathbf{0}) = \mathbf{0}$ / $T(\mathbf{0}) = 0$.*

Proof. We use linearity (ii) in Lemma 2.23 with any \mathbf{x} : $T(\mathbf{0}) = T(0\mathbf{x}) = 0T(\mathbf{x}) = \mathbf{0}$ / 0 . \square

Linearity generalizes to linear combinations of more (or less) than two vectors, and this turns out to be the key for fully understanding linear transformations.

Lemma 2.25. *Let $T : \mathbb{R}^n \rightarrow \mathbb{R}^m$ / $T : \mathbb{R}^n \rightarrow \mathbb{R}$ be a linear transformation / linear functional, let $\mathbf{x}_1, \mathbf{x}_2, \dots, \mathbf{x}_\ell \in \mathbb{R}^n$ and $\lambda_1, \lambda_2, \dots, \lambda_\ell \in \mathbb{R}$. Then*

$$T\left(\sum_{j=1}^{\ell} \lambda_j \mathbf{x}_j\right) = \sum_{j=1}^{\ell} \lambda_j T(\mathbf{x}_j).$$

Proof (with dots). We use linearity (i) and (ii) from Lemma 2.23 to obtain

$$T\left(\sum_{j=1}^{\ell} \lambda_j \mathbf{x}_j\right) = T\left(\sum_{j=1}^{\ell-1} \lambda_j \mathbf{x}_j + \lambda_\ell \mathbf{x}_\ell\right) \stackrel{(i)}{=} T\left(\sum_{j=1}^{\ell-1} \lambda_j \mathbf{x}_j\right) + T(\lambda_\ell \mathbf{x}_\ell) \stackrel{(ii)}{=} T\left(\sum_{j=1}^{\ell-1} \lambda_j \mathbf{x}_j\right) + \lambda_\ell T(\mathbf{x}_\ell).$$

Doing the same for $\ell - 1$, we further get

$$T\left(\sum_{j=1}^{\ell} \lambda_j \mathbf{x}_j\right) = \underbrace{T\left(\sum_{j=1}^{\ell-1} \lambda_j \mathbf{x}_j\right)}_{T(\sum_{j=1}^{\ell-1} \lambda_j \mathbf{x}_j)} + \lambda_{\ell-1} T(\mathbf{x}_{\ell-1}) + \lambda_\ell T(\mathbf{x}_\ell).$$

Repeating this for $\ell - 2, \dots, 1$, we finally obtain

$$T\left(\sum_{j=1}^{\ell} \lambda_j \mathbf{x}_j\right) = \underbrace{T\left(\sum_{j=1}^0 \lambda_j \mathbf{x}_j\right)}_{T(\mathbf{0})} + \lambda_1 T(\mathbf{x}_1) + \dots + \lambda_{\ell-1} T(\mathbf{x}_{\ell-1}) + \lambda_\ell T(\mathbf{x}_\ell) = \sum_{j=1}^{\ell} \lambda_j T(\mathbf{x}_j),$$

because $T(\mathbf{0}) = \mathbf{0}$ by Lemma 2.24. \square

This proof has the usual dots in $\lambda_1 T(\mathbf{x}_1) + \cdots + \lambda_{\ell-1} T(\mathbf{x}_{\ell-1}) + \lambda_\ell T(\mathbf{x}_\ell)$, but also dots of a different quality, namely the ones in $\ell - 2, \dots, 1$. These dots indicate a repeating pattern in the proof itself. In Section 1.1.5, we have seen dot-free notations for sequences and sums and argued that they are more precise than the ones with the dots; is there also a dot-free notation for proofs such as the one above? Yes, and this notation is known as *proof by induction*, an important proof technique in mathematics. The concept of induction is the following: We want to prove that some statement holds for all natural numbers n . For example, that

$$\sum_{j=1}^n j = \frac{n(n+1)}{2}.$$

This is a famous formula often attributed to the German mathematician *Carl Friedrich Gauss*; legend has it that he discovered the formula in school, trying to avoid a very boring task assigned to him and his classmates by the teacher: sum up all numbers from 1 to 100! Indeed, using the formula, one immediately sees that the sum is 5,050.

Back to induction: We first check the *base case*, meaning that the statement holds for the first natural number $n = 0$. Indeed, for $n = 0$, both sides of Gauss' formula give 0.

For $n > 0$, we perform the *induction step*. This proves an implication: *if* the statement is true for the number $n - 1$ (this is the *induction hypothesis*), *then* it is also true for n (this concludes the induction step). In our example, if Gauss' formula is true for $n - 1$, we can compute

$$\sum_{j=1}^n j = \left(\sum_{j=1}^{n-1} j \right) + n = \frac{(n-1)n}{2} + n = \frac{n(n+1)}{2}.$$

\uparrow
induction hypothesis

\uparrow
easy calculation

So if Gauss' formula is true for $n - 1$, it is indeed also true for n .

Once we have completed base case and induction step, we have proved the statement for all natural numbers. Why is that? We have proved it for the first natural number (base case), and the induction step lets us conclude that it is also true for the second natural number (we know that if it is true for the first, then it is also true for the second). So we have proved it for the second natural number, and the induction step lets us conclude that it is also true for the third natural number. And so on (these are the dots, but now they do not appear in the proof, but in its justification). Every natural number is eventually reached by this sequence of steps, so we have proved it for all natural numbers. Even though there are infinitely many natural numbers, every natural number itself is finite, so we eventually get to it.

This is often illustrated with the *domino effect*. Suppose you have an infinite sequence of dominoes, numbered $0, 1, 2, \dots$, and arranged in line. For every n , you know that if domino $n - 1$ falls, then it knocks over domino n (induction step). Hence, if you knock over domino 0 (base case), then every domino will eventually fall.

Let us exercise induction for the statement of Lemma 2.25. Here, the natural number symbol is not n but ℓ .

Proof of Lemma 2.25 (by induction). Base case: For $\ell = 0$, the statement reads as $T(\mathbf{0}) = \mathbf{0}$ which is true by Lemma 2.24. For $\ell > 0$, we perform the induction step: if the statement is true for $\ell - 1$, we compute

$$\begin{aligned}
 T\left(\sum_{j=1}^{\ell} \lambda_j \mathbf{x}_j\right) &= T\left(\sum_{j=1}^{\ell-1} \lambda_j \mathbf{x}_j + \lambda_{\ell} \mathbf{x}_{\ell}\right) \stackrel{(i)}{=} T\left(\sum_{j=1}^{\ell-1} \lambda_j \mathbf{x}_j\right) + T(\lambda_{\ell} \mathbf{x}_{\ell}) \\
 &\stackrel{(ii)}{=} T\left(\sum_{j=1}^{\ell-1} \lambda_j \mathbf{x}_j\right) + \lambda_{\ell} T(\mathbf{x}_{\ell}) \\
 &= \sum_{j=1}^{\ell-1} \lambda_j T(\mathbf{x}_j) + \lambda_{\ell} T(\mathbf{x}_{\ell}) \quad (\text{induction hypothesis}) \\
 &= \sum_{j=1}^{\ell} \lambda_j T(\mathbf{x}_j).
 \end{aligned}$$

So if the statement is true for $\ell - 1$, it is indeed true for ℓ . □

The proof is conceptually the same as the previous one, but replaces “repeating this for $\ell - 2, \dots, 1$ ” by a single step. Whenever you see a proof using such repetitions, or saying “and so on...”, you can be pretty sure that this is an informal proof by induction. There is nothing wrong with a proof using “and so on”, as long as you can turn it into a formal proof by induction, if needed.

Looking back at the proofs of Lemma 1.28 and Lemma 2.11, these were also inductions in disguise, where we have used “one by one” to indicate a repeating step in a proof. In these cases, we would actually apply *limited induction*, meaning that we do not prove a result for all natural numbers, but only the ones that are at most as large as some given number. But the principle is the same.

2.2.3 The matrix of a linear transformation

We already know that every matrix transformation is a linear transformation (Observation 2.22). Now we will prove that every linear transformation (Definition 2.21) is also a matrix transformation (Definition 2.18). This means that there are no “fancy” linear transformations beyond the ones that we already know, namely the matrix transformations.

Theorem 2.26. *Let $T : \mathbb{R}^n \rightarrow \mathbb{R}^m$ be a linear transformation. There is a unique $m \times n$ matrix A such that $T = T_A$ (meaning that $T(\mathbf{x}) = T_A(\mathbf{x})$ for all $\mathbf{x} \in \mathbb{R}^n$). This matrix is*

$$A = \begin{bmatrix} \left| \begin{array}{c} T(\mathbf{e}_1) \\ \vdots \end{array} \right| & \left| \begin{array}{c} T(\mathbf{e}_2) \\ \vdots \end{array} \right| & \cdots & \left| \begin{array}{c} T(\mathbf{e}_n) \\ \vdots \end{array} \right| \end{bmatrix}.$$

Proof. In order for $T = T_A$ to hold, we must in particular have $T(\mathbf{e}_j) = T_A(\mathbf{e}_j) = A\mathbf{e}_j$ (which is the j -th column of A) for all $j \in [n]$. Hence, the only candidate for A is the one mentioned in the statement of the theorem, the matrix whose columns are the m -dimensional output vectors that we get when we apply T to the n -dimensional standard unit vectors as inputs. With this matrix A , we indeed get $T = T_A$, because for all $\mathbf{x} \in \mathbb{R}^n$,

$$T_A(\mathbf{x}) = A\mathbf{x} = \sum_{j=1}^n x_j T(\mathbf{e}_j) = T\left(\sum_{j=1}^n x_j \mathbf{e}_j\right) = T(\mathbf{x}).$$

In the second equality, we use Definition 2.4 of matrix-vector multiplication with our matrix A as defined above. In the third equality, we apply Lemma 2.25, and the last equality uses that every vector \mathbf{x} is a linear combination of the standard unit vectors, where the scalars are simply the entries of \mathbf{x} , as in

$$\begin{pmatrix} x_1 \\ x_2 \\ x_3 \end{pmatrix} = x_1 \underbrace{\begin{pmatrix} 1 \\ 0 \\ 0 \end{pmatrix}}_{\mathbf{e}_1} + x_2 \underbrace{\begin{pmatrix} 0 \\ 1 \\ 0 \end{pmatrix}}_{\mathbf{e}_2} + x_3 \underbrace{\begin{pmatrix} 0 \\ 0 \\ 1 \end{pmatrix}}_{\mathbf{e}_3}.$$

□

A consequence of this lemma is the following: every linear transformation is completely determined by its behavior on the standard unit vectors. This also explains why we have paid special attention to this behavior in Figures 2.6 and 2.7.

2.2.4 Kernel and Image

For every linear transformation, there are two important sets of vectors.

Definition 2.27 (Kernel and image). *Let $T : \mathbb{R}^n \rightarrow \mathbb{R}^m$ be a linear transformation. The set*

$$\mathbf{Ker}(T) := \{\mathbf{x} \in \mathbb{R}^n : T(\mathbf{x}) = \mathbf{0}\} \subseteq \mathbb{R}^n$$

is the kernel of T . The set

$$\mathbf{Im}(T) := \{T(\mathbf{x}) : \mathbf{x} \in \mathbb{R}^n\} \subseteq \mathbb{R}^m$$

is the image of T .

The image of T is the set of all outputs that T can produce. This is actually a familiar concept.

Observation 2.28. *Let $T : \mathbb{R}^n \rightarrow \mathbb{R}^m$ be a linear transformation and A the unique $m \times n$ matrix (that exists by Theorem 2.26) such that $T = T_A$. Then*

$$\mathbf{Im}(T) = \mathbf{C}(A),$$

the column space of A .

This immediately follows from Definition 2.9 of $C(A)$ and $T(\mathbf{x}) = A\mathbf{x}$. In light of this, some sources call the column space of A the image of A (although it is officially the image of T_A).

For the kernel, we have a similar statement.

Observation 2.29. *Let $T : \mathbb{R}^n \rightarrow \mathbb{R}^m$ be a linear transformation and A the unique $m \times n$ matrix (that exists by Theorem 2.26) such that $T = T_A$. Then*

$$\mathbf{Ker}(T) = \mathbf{N}(A),$$

the nullspace of A ; see Definition 2.17.

Again, some sources call the nullspace of A the kernel of A (although it is officially the kernel of T_A).

Table 2.1 provides kernel and image of three linear transformations that we have considered as examples on page 64.

T	$\mathbf{Ker}(T)$	$\mathbf{Im}(T)$
$\mathbb{R}^n \rightarrow \mathbb{R}^1, \quad \mathbf{x} \mapsto \sum_{i=1}^n x_i$	$\{\mathbf{x} \in \mathbb{R}^n : \sum_{i=1}^n x_i = 0\}$	\mathbb{R}^1
$\mathbb{R}^n \rightarrow \mathbb{R}^m, \quad \mathbf{x} \mapsto \mathbf{0}$	\mathbb{R}^n	$\{\mathbf{0}\}$
$\mathbb{R}^m \rightarrow \mathbb{R}^m, \quad \mathbf{x} \mapsto \mathbf{x}$	$\{\mathbf{0}\}$	\mathbb{R}^m

Table 2.1: Kernel and image of some linear transformations

Exercise 2.30. *Let A be an $m \times m$ (square) matrix, $\lambda \in \mathbb{R}$ a scalar. Prove that the function $T : \mathbb{R}^m \rightarrow \mathbb{R}^m, \mathbf{x} \mapsto A\mathbf{x} + \lambda\mathbf{x}$ is a linear transformation!*

Exercise 2.31. *Prove that the function $T : \mathbb{R}^2 \rightarrow \mathbb{R}, \mathbf{x} \mapsto x_1x_2$ is not a linear functional!*

Exercise 2.32. *Let $\mathbf{v} \in \mathbb{R}^m, \mathbf{w} \in \mathbb{R}^n$ be nonzero vectors and consider the $m \times n$ matrix $A = [v_i w_j]_{i=1, j=1}^m, n$ (this matrix has rank 1 by Lemma 2.15). Give formulas for $\mathbf{Ker}(T_A)$ and $\mathbf{Im}(T_A)$, depending on the vectors \mathbf{v} and \mathbf{w} !*

2.3 Matrix multiplication

If we combine two matrix transformations T_A and T_B into one transformation (“first do T_B , then T_A ”), it turns out that we get another matrix transformation. Its matrix can be obtained by *multiplying* A and B in a suitable way. Here, we formally introduce this kind of matrix multiplication and derive its essential properties. Matrix multiplication naturally comes up in matrix decompositions where we write a matrix as a product of other matrices with the goal of revealing some structural properties. We present the CR decomposition in this section; other decompositions will appear in subsequent chapters.

2.3.1 Combining matrix transformations

We want to show that combining two matrix transformations gives another matrix transformation. Before we do this, let us look at two examples. In Figure 2.6, we have seen two matrix transformations, mirroring and stretching. Figure 2.11 shows what happens if we combine them by first stretching the input to obtain an intermediate result, and then mirroring the intermediate result to get the output. We can also check that there is a single matrix transformation T_C that gets us from the input to the output in one step. We can read off the matrix C from what happens to the unit vectors e_1 (red vector in the input) and e_2 (blue vector in the input) when we apply T_C : We know from Theorem 2.26 that the first column of C is $T_C(e_1)$ (the red vector in the output), and the second column of C is $T_C(e_2)$ (the blue vector in the output).

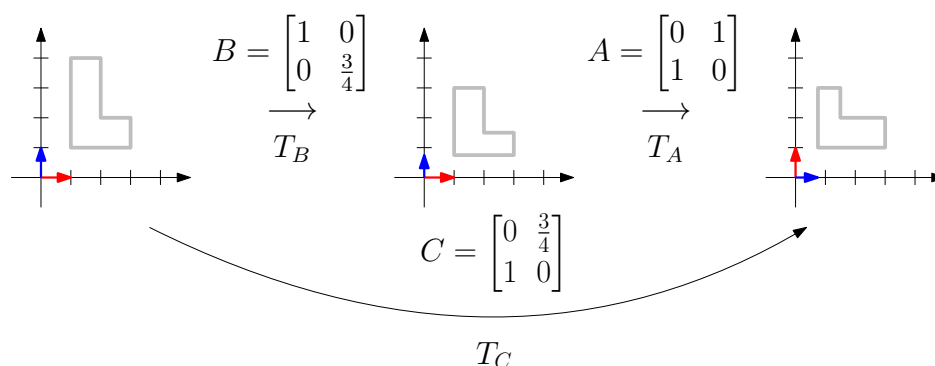
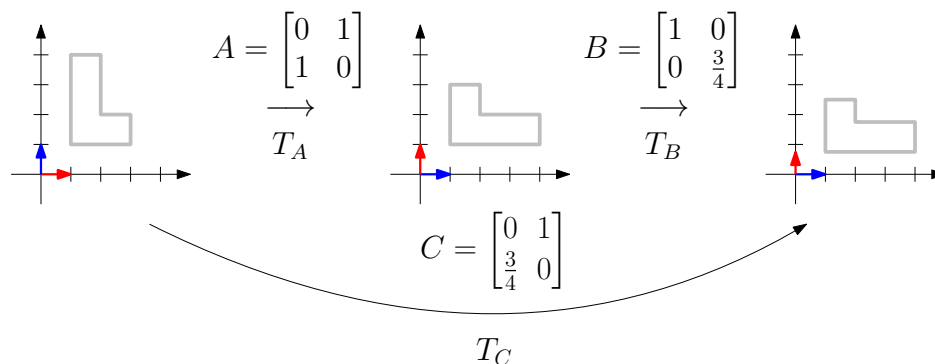


Figure 2.11: Left: The input. Middle: Stretching the input by a factor of $\frac{3}{4}$ along the second coordinate. Right: The output, obtained from mirroring the intermediate result along the diagonal. Bottom: Combining the two transformations into one.

We can also combine the two matrix transformations in reverse order: first mirror the input, then stretch the intermediate result:



The output is different from the one in Figure 2.11, so the combination order matters; but also here, we get a single matrix transformation T_C that combines the two steps into one. Again, we can read off C from how the two standard unit vectors get transformed.

Next we formalize this kind of combination that generally works in the context of functions.

Definition 2.33 (Composition of functions). *Let $g : X \rightarrow Y$ and $f : Y \rightarrow Z$ be two functions where X, Y, Z are arbitrary sets. The function $h : X \rightarrow Z$,*

$$h : x \mapsto f(g(x))$$

is the composition of f and g , written as $f \circ g$ (“first apply g , then f ”).

This works because the set Y is both the codomain of g and the domain of f . In other words, every possible output of g is a valid input for f . Here is what happens when we compose two matrix transformations.

Lemma 2.34 (Composition of matrix transformations). *Let $T_B : \mathbb{R}^b \rightarrow \mathbb{R}^n$ and $T_A : \mathbb{R}^n \rightarrow \mathbb{R}^a$ be two matrix transformations. The composition $T_A \circ T_B : \mathbb{R}^b \rightarrow \mathbb{R}^a$ (“first do T_B , then T_A ”) is another matrix transformation.*

Proof. We only need to show that $T_A \circ T_B$ is a linear transformation according to Definition 2.21. Then it follows from Theorem 2.26 that $T_A \circ T_B$ is also a matrix transformation. To verify the linearity axiom in Definition 2.21, we use Definition 2.33 of composition and linearity of both T_B and T_A :

$$\begin{aligned} (T_A \circ T_B)(\lambda_1 \mathbf{x}_1 + \lambda_2 \mathbf{x}_2) &= T_A(T_B(\lambda_1 \mathbf{x}_1 + \lambda_2 \mathbf{x}_2)) \text{ (composition)} \\ &= T_A(\lambda_1 T_B(\mathbf{x}_1) + \lambda_2 T_B(\mathbf{x}_2)) \text{ (linearity of } T_B) \\ &= \lambda_1 T_A(T_B(\mathbf{x}_1)) + \lambda_2 T_A(T_B(\mathbf{x}_2)) \text{ (linearity of } T_A) \\ &= \lambda_1 (T_A \circ T_B)(\mathbf{x}_1) + \lambda_2 (T_A \circ T_B)(\mathbf{x}_2). \text{ (composition)} \end{aligned}$$

□

2.3.2 Definition and basic properties

Suppose we are given an $a \times n$ matrix A and an $n \times b$ matrix B . From Lemma 2.34, we know that $T_A \circ T_B = T_C$ for some $a \times b$ matrix C . We can think of T_C as “first do T_B , then T_A ”. Equality between two functions ($T_A \circ T_B$ and T_C in our case) means that for every possible input, both functions produce the same output. Here is how we get the matrix C from this.

Lemma 2.35 (Matrix of the composition). *Let A be an $a \times n$ matrix and*

$$B = \begin{bmatrix} | & | & & | \\ \mathbf{x}_1 & \mathbf{x}_2 & \cdots & \mathbf{x}_b \\ | & | & & | \end{bmatrix}$$

an $n \times b$ matrix. The $a \times b$ matrix

$$C = \begin{bmatrix} | & | & & | \\ A\mathbf{x}_1 & A\mathbf{x}_2 & \cdots & A\mathbf{x}_b \\ | & | & & | \end{bmatrix}$$

is the unique matrix that satisfies $T_C = T_A \circ T_B$.

Proof. We already know from Lemma 2.34 that there is a matrix C with $T_C = T_A \circ T_B$. Both T_C and $T_A \circ T_B$ must produce the same output on every standard unit vector:

$$T_C(\mathbf{e}_j) = (T_A \circ T_B)(\mathbf{e}_j) \stackrel{\text{Def. 2.33}}{=} T_A(T_B(\mathbf{e}_j)) \stackrel{\text{Def. 2.18}}{=} AT_B(\mathbf{e}_j), \quad j = 1, 2, \dots, b. \quad (2.1)$$

We know that $T_C(\mathbf{e}_j) = C\mathbf{e}_j$ is the j -th column of C , and $T_B(\mathbf{e}_j) = B\mathbf{e}_j$ is the j -th column of B that we called \mathbf{x}_j . Hence, (2.1) says that the j -th column of C is $A\mathbf{x}_j$ for all j , and this gives

$$C = \begin{bmatrix} | & | & & | \\ A\mathbf{x}_1 & A\mathbf{x}_2 & \cdots & A\mathbf{x}_b \\ | & | & & | \end{bmatrix}.$$

□

We will now call this matrix C the *product* of A and B , and write it as AB .

Definition 2.36 (Matrix multiplication in column notation). Let A be an $a \times n$ matrix and

$$B = \begin{bmatrix} | & | & & | \\ \mathbf{x}_1 & \mathbf{x}_2 & \cdots & \mathbf{x}_b \\ | & | & & | \end{bmatrix}$$

an $n \times b$ matrix. The $a \times b$ matrix

$$AB := \begin{bmatrix} | & | & & | \\ A\mathbf{x}_1 & A\mathbf{x}_2 & \cdots & A\mathbf{x}_b \\ | & | & & | \end{bmatrix}$$

is the product of A and B .

Note that for the product AB to be defined, the number of columns of A needs to match the number of rows of B ; this is the number n in Definition 2.36. This naturally connects to the fact that the composition $T_A \circ T_B$ is only defined if the output of T_B can be used as input for T_A ; in other words, we must have $T_B : \cdots \rightarrow \mathbb{R}^n$ and $T_A : \mathbb{R}^n \rightarrow \cdots$ for some n . We can nicely summarize Lemma 2.35 and Definition 2.36 as

Corollary 2.37 (Matrix of the composition). Let A be an $a \times n$ matrix and B be an $n \times b$ matrix. Then $T_A \circ T_B = T_{AB}$.

One can define the product AB directly, without referring to matrix-vector multiplication: in order to compute the entry of AB in row i and column j , we take the scalar product of row i of A and column j of B .

Observation 2.38. *Let*

$$A = \begin{bmatrix} - & \mathbf{u}_1^\top & - \\ - & \mathbf{u}_2^\top & - \\ & \vdots & \\ - & \mathbf{u}_a^\top & - \end{bmatrix} \in \mathbb{R}^{a \times n}, \quad B = \begin{bmatrix} | & | & \cdots & | \\ \mathbf{x}_1 & \mathbf{x}_2 & \cdots & \mathbf{x}_b \\ | & | & \cdots & | \end{bmatrix} \in \mathbb{R}^{n \times b}.$$

Then

$$AB = \underbrace{\begin{bmatrix} \mathbf{u}_1^\top \mathbf{x}_1 & \mathbf{u}_1^\top \mathbf{x}_2 & \cdots & \mathbf{u}_1^\top \mathbf{x}_b \\ \mathbf{u}_2^\top \mathbf{x}_1 & \mathbf{u}_2^\top \mathbf{x}_2 & \cdots & \mathbf{u}_2^\top \mathbf{x}_b \\ \vdots & \vdots & \ddots & \vdots \\ \mathbf{u}_a^\top \mathbf{x}_1 & \mathbf{u}_a^\top \mathbf{x}_2 & \cdots & \mathbf{u}_a^\top \mathbf{x}_b \end{bmatrix}}_{ab \text{ scalar products}} = [\mathbf{u}_i^\top \mathbf{x}_j]_{i=1, j=1}^{a \quad b} \in \mathbb{R}^{a \times b}.$$

This works because $A\mathbf{x}_j$, the j -th column of AB , can be written as $A\mathbf{x}_j = [\mathbf{u}_i^\top \mathbf{x}_j]_{i=1}^a$; see Observation 2.8 (matrix-vector multiplication with A in row notation).

The previous definition of AB corresponds to how matrix multiplication is usually done, mentally; see Figure 2.12.

$$\begin{array}{ccc} \begin{bmatrix} \mathbf{2} & \cdots & \mathbf{3} \\ 3 & -1 \end{bmatrix} \blacktriangleright \begin{bmatrix} -3 & \mathbf{1} & 3 \\ 2 & -\mathbf{1} & 0 \end{bmatrix} = \begin{bmatrix} 0 & -1 & 6 \\ -11 & 4 & 9 \end{bmatrix} & \begin{bmatrix} 2 & 3 \\ \mathbf{3} & -\mathbf{1} \end{bmatrix} \blacktriangleright \begin{bmatrix} -3 & 1 & \mathbf{3} \\ 2 & -1 & \mathbf{0} \end{bmatrix} = \begin{bmatrix} 0 & -1 & 6 \\ -11 & 4 & \mathbf{9} \end{bmatrix} \\ \mathbf{2} \cdot \mathbf{1} + \mathbf{3} \cdot (-1) = -1 & \mathbf{3} \cdot \mathbf{3} + (-1) \cdot \mathbf{0} = \mathbf{9} \end{array}$$

Figure 2.12: Matrix multiplication: multiply (scalar product) each row of the first matrix with each column of the second matrix! In this example, this needs six scalar products, two of which are shown.

Finally, here is the dot-free definition of AB .

Observation 2.39. *Let $A = [a_{ij}]_{i=1, j=1}^{a \quad n}$, $B = [b_{ij}]_{i=1, j=1}^{n \quad b}$. Then*

$$AB = \left[\sum_{\ell=1}^n a_{i\ell} b_{\ell j} \right]_{i=1, j=1}^{a \quad b}.$$

This is just a different way of writing Observation 2.38, since $\sum_{\ell=1}^n a_{i\ell} b_{\ell j}$ equals $\mathbf{u}_i^\top \mathbf{x}_j$ (“row i of A times column j of B ”).

As two more examples, let us come back to the matrices

$$A = \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix}, \quad B = \begin{bmatrix} 1 & 0 \\ 0 & \frac{3}{4} \end{bmatrix}.$$

of Figure 2.11. We have

$$AB = \begin{bmatrix} (0 \ 1) \begin{pmatrix} 1 \\ 0 \end{pmatrix} & (0 \ 1) \begin{pmatrix} 0 \\ \frac{3}{4} \end{pmatrix} \\ (1 \ 0) \begin{pmatrix} 1 \\ 0 \end{pmatrix} & (1 \ 0) \begin{pmatrix} 0 \\ \frac{3}{4} \end{pmatrix} \end{bmatrix} = \begin{bmatrix} 0 & \frac{3}{4} \\ 1 & 0 \end{bmatrix}$$

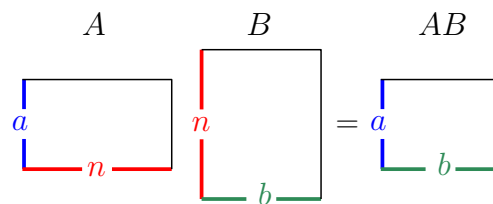
and

$$BA = \begin{bmatrix} (1 \ 0) \begin{pmatrix} 0 \\ 1 \end{pmatrix} & (1 \ 0) \begin{pmatrix} 1 \\ 0 \end{pmatrix} \\ (0 \ \frac{3}{4}) \begin{pmatrix} 0 \\ 1 \end{pmatrix} & (0 \ \frac{3}{4}) \begin{pmatrix} 1 \\ 0 \end{pmatrix} \end{bmatrix} = \begin{bmatrix} 0 & 1 \\ \frac{3}{4} & 0 \end{bmatrix}.$$

We see that matrix multiplication is not commutative, since we have $BA \neq AB$ here. On page 70, we already saw that it makes a difference whether we apply T_A or T_B first. Also in life, the order of combining two activities typically matters. You first put your socks on, and then the shoes. The reverse order would have a quite different effect. On the other hand, there are activities that do commute, for example, it does not matter whether you put your left or right shoe on first. Similarly, we do have $AB = BA$ in some cases.

If A and B are not square matrices, it may happen that AB is defined and BA is not. Or that both products are defined, but are of different shape.

It is not hard to understand the situation exactly. For AB to be defined, A must be $a \times n$ and B must be $n \times b$, for some number n . For BA to be defined as well, we also need $a = b$. This means, A must be $m \times n$ and B must be $n \times m$, for some number m . Then AB is $m \times m$ and BA is $n \times n$, so both products are square matrices. If $m = n$, they have the same shape, otherwise, one is larger than the other. Here is a pictorial view of matrix multiplication:



An easy but important fact is how matrix multiplication interacts with transposition (Definition 2.12).

Lemma 2.40. *Let A be an $a \times n$ matrix and B an $n \times b$ matrix. Then*

$$(AB)^\top = B^\top A^\top.$$

Proof. Since B^\top is a $b \times n$ matrix and A^\top an $n \times a$ matrix, the product $B^\top A^\top$ is a $b \times a$ matrix and therefore has the same shape as $(AB)^\top$.

Next, we compare the two matrices

$$C = (AB)^\top = [c_{ij}]_{i=1, j=1}^b, a \quad \text{and} \quad D = B^\top A^\top = [d_{ij}]_{i=1, j=1}^b, a$$

entry by entry. By Definition 2.12 of the transpose, c_{ij} is the entry in row j and column i of AB and therefore the scalar product of the j -th row of A and the i -th column of B ; see Observation 2.38:

$$c_{ij} = (j\text{-th row of } A) \cdot (i\text{-th column of } B).$$

By the same observation,

$$d_{ij} = (i\text{-th row of } B^\top) \cdot (j\text{-th column of } A^\top).$$

To conclude $c_{ij} = d_{ij}$, it remains to note that

$$(j\text{-th row of } A) = (j\text{-th column of } A^\top) \text{ and } (i\text{-th column of } B) = (i\text{-th row of } B^\top).$$

□

A proof by induction can be used to show that this generalizes to more matrices, for example

$$(ABC)^\top = C^\top B^\top A^\top.$$

As seen before in Corollary 2.7 for matrix-vector multiplication, identity matrices also have no effect in matrix multiplications. We leave the proof of this as an (easy) exercise.

Corollary 2.41. *Let I be the $m \times m$ identity matrix. Then $IA = A$ for all $m \times n$ matrices, and $AI = A$ for all $n \times m$ matrices.*

2.3.3 Distributivity and associativity

While matrix multiplication is not commutative (we may have $AB \neq BA$), two other important laws hold.

Lemma 2.42. *Let A, B, C be three matrices. Whenever the respective sums and products in the following are defined, we have*

$$(i) \quad A(B + C) = AB + AC \text{ and } (A + B)C = AC + BC; \quad (\text{distributivity})$$

$$(ii) \quad (AB)C = A(BC). \quad (\text{associativity})$$

Proof. We omit the mechanical proof of distributivity, based on Definition 2.2 (i) of matrix addition, Definition 2.36 of matrix multiplication, and distributivity of the real numbers: $a(b + c) = ab + ac$ and $(a + b)c = ac + bc$.

For associativity, the proof is more interesting. We use the fact that matrix multiplication according to Definition 2.36 has been designed in such a way that AB is the matrix of the linear transformation $T_A \circ T_B$, the composition of T_A and T_B ; see Corollary 2.37. Using this once more, we see that $(AB)C$ is the matrix of $(T_A \circ T_B) \circ T_C$. Similarly, $A(BC)$ is the matrix of $T_A \circ (T_B \circ T_C)$.

It remains to observe that $(T_A \circ T_B) \circ T_C$ and $T_A \circ (T_B \circ T_C)$ are the same linear transformations and therefore also have the same matrices $(AB)C = A(BC)$ by Theorem 2.26.

Indeed, applying Definition 2.33 of composition twice for each transformation, we get that for every input \mathbf{x} , both functions produce the same output:

$$\begin{aligned} ((T_A \circ T_B) \circ T_C)(\mathbf{x}) &= (T_A \circ T_B)(T_C(\mathbf{x})) = T_A(T_B(T_C(\mathbf{x}))), \\ (T_A \circ (T_B \circ T_C))(\mathbf{x}) &= T_A((T_B \circ T_C)(\mathbf{x})) = T_A(T_B(T_C(\mathbf{x}))). \end{aligned}$$

□

Knowing that matrix multiplication is associative allows us to write ABC for a product of three matrices. As it does not matter whether we compute this as $(AB)C$ or $A(BC)$, we can as well omit the brackets.

This also works for more matrices and is then called *generalized associativity*. For example, $(AB)(CD) = A((BC)D) = \dots = ABCD$. This can be proved along the same lines. $(AB)(CD)$ is the matrix of $(T_A \circ T_B) \circ (T_C \circ T_D)$, while $A((BC)D)$ is the matrix of $T_A \circ ((T_B \circ T_C) \circ T_D)$. Again, both linear transformations have the same matrix, since they are the same transformations: on every input \mathbf{x} , both produce the same output $T_A(T_B(T_C(T_D(\mathbf{x}))))$ as we can see by repeatedly applying Definition 2.33 of composition.

2.3.4 Everything “is” matrix multiplication

Let us look at a matrix multiplication AB in the special case where the matrix B has just one column. For example,

$$\underbrace{\begin{bmatrix} 0 & 1 \\ 2 & 3 \\ 4 & 5 \end{bmatrix}}_{A, 3 \times 2} \underbrace{\begin{bmatrix} 1 \\ 1 \end{bmatrix}}_{B, 2 \times 1} = \underbrace{\begin{bmatrix} 1 \\ 5 \\ 9 \end{bmatrix}}_{AB, 3 \times 1}.$$

Except for the shapes of the brackets, this looks exactly like the matrix-vector multiplication

$$\underbrace{\begin{bmatrix} 0 & 1 \\ 2 & 3 \\ 4 & 5 \end{bmatrix}}_{A, 3 \times 2} \underbrace{\begin{pmatrix} 1 \\ 1 \end{pmatrix}}_{\mathbf{x} \in \mathbb{R}^2} = \underbrace{\begin{pmatrix} 1 \\ 5 \\ 9 \end{pmatrix}}_{A\mathbf{x} \in \mathbb{R}^3}.$$

This easily follows from Definition 2.36 of matrix multiplication: when B has just one column \mathbf{x} , then AB also has just one column, the matrix-vector product $A\mathbf{x}$.

In fact, for *every* matrix multiplication where at least one of the matrices is “flat” (has only one row or one column), we have a corresponding version involving vectors, covectors, or scalars, see Table 2.2 on the next page. *covector-matrix multiplication* as well as the *outer product* are new, and we will formally define them next. We will also explore why this “zoo” of rather diverse multiplications contains exactly the “animals” in the table, and why they all work like matrix multiplications. For just using them in the following, only the latter point (they work like matrix multiplications) is important and Table 2.2 is all that you need.

matrix multiplication	covector / vector / scalar version		
Example	Example	operation	
$\underbrace{\begin{bmatrix} 0 & 1 \\ 2 & 3 \\ 4 & 5 \end{bmatrix}}_{3 \times 2} \underbrace{\begin{bmatrix} 1 \\ 1 \end{bmatrix}}_{2 \times 1} = \underbrace{\begin{bmatrix} 1 \\ 5 \\ 9 \end{bmatrix}}_{3 \times 1}$	$\underbrace{\begin{bmatrix} 0 & 1 \\ 2 & 3 \\ 4 & 5 \end{bmatrix}}_{3 \times 2} \underbrace{\begin{pmatrix} 1 \\ 1 \end{pmatrix}}_{\in \mathbb{R}^2} = \underbrace{\begin{pmatrix} 1 \\ 5 \\ 9 \end{pmatrix}}_{\in \mathbb{R}^3}$	$\underbrace{A\mathbf{x}}_{\in \mathbb{R}^m}$	matrix-vector multiplication $A \in \mathbb{R}^{m \times n}, \mathbf{x} \in \mathbb{R}^n$ (Definition 2.4)
$\underbrace{\begin{bmatrix} 1 & 2 \end{bmatrix}}_{1 \times 2} \underbrace{\begin{bmatrix} 3 \\ 4 \end{bmatrix}}_{2 \times 1} = \underbrace{\begin{bmatrix} 11 \end{bmatrix}}_{1 \times 1}$	$\underbrace{\begin{pmatrix} 1 & 2 \end{pmatrix}}_{\in (\mathbb{R}^2)^*} \underbrace{\begin{pmatrix} 3 \\ 4 \end{pmatrix}}_{\in \mathbb{R}^2} = \underbrace{11}_{\in \mathbb{R}}$	$\underbrace{\mathbf{v}^\top \mathbf{w}}_{\in \mathbb{R}}$	scalar product $\mathbf{v}, \mathbf{w} \in \mathbb{R}^m$ (Definition 1.9, Definition 1.19)
$\underbrace{\begin{bmatrix} 1 & 1 & 1 \end{bmatrix}}_{1 \times 3} \underbrace{\begin{bmatrix} 0 & 1 \\ 2 & 3 \\ 4 & 5 \end{bmatrix}}_{3 \times 2} = \underbrace{\begin{bmatrix} 6 & 9 \end{bmatrix}}_{1 \times 2}$	$\underbrace{\begin{pmatrix} 1 & 1 & 1 \end{pmatrix}}_{\in (\mathbb{R}^3)^*} \underbrace{\begin{bmatrix} 0 & 1 \\ 2 & 3 \\ 4 & 5 \end{bmatrix}}_{3 \times 2} = \underbrace{\begin{pmatrix} 6 & 9 \end{pmatrix}}_{\in (\mathbb{R}^2)^*}$	$\underbrace{\mathbf{y}^\top A}_{\in (\mathbb{R}^n)^*}$	covector-matrix multiplication $\mathbf{y}^\top \in (\mathbb{R}^m)^*, A \in \mathbb{R}^{m \times n}$ (Definition 2.43)
$\underbrace{\begin{bmatrix} 3 \\ 4 \\ 5 \end{bmatrix}}_{3 \times 1} \underbrace{\begin{bmatrix} 1 & 2 \end{bmatrix}}_{1 \times 2} = \underbrace{\begin{bmatrix} 3 & 6 \\ 4 & 8 \\ 5 & 10 \end{bmatrix}}_{3 \times 2}$	$\underbrace{\begin{pmatrix} 3 \\ 4 \\ 5 \end{pmatrix}}_{\in \mathbb{R}^3} \underbrace{\begin{pmatrix} 1 & 2 \end{pmatrix}}_{\in (\mathbb{R}^2)^*} = \underbrace{\begin{bmatrix} 3 & 6 \\ 4 & 8 \\ 5 & 10 \end{bmatrix}}_{3 \times 2}$	$\underbrace{\mathbf{v}\mathbf{w}^\top}_{m \times n}$	outer product $\mathbf{v} \in \mathbb{R}^m, \mathbf{w} \in \mathbb{R}^n$ (Definition 2.44)
$\underbrace{\begin{bmatrix} 3 \\ 4 \end{bmatrix}}_{2 \times 1} \underbrace{\begin{bmatrix} 2 \end{bmatrix}}_{1 \times 1} = \underbrace{\begin{bmatrix} 6 \\ 8 \end{bmatrix}}_{2 \times 1}$	$\underbrace{\begin{pmatrix} 3 \\ 4 \end{pmatrix}}_{\in \mathbb{R}^2} \underbrace{2}_{\in \mathbb{R}} = \underbrace{\begin{pmatrix} 6 \\ 8 \end{pmatrix}}_{\in \mathbb{R}^2}$	$\underbrace{\mathbf{v}\lambda}_{\in \mathbb{R}^m}$	scalar multiplication (the other way around) $\mathbf{v} \in \mathbb{R}^m, \lambda \in \mathbb{R}$ (Definition 1.3).
$\underbrace{\begin{bmatrix} 2 \end{bmatrix}}_{1 \times 1} \underbrace{\begin{bmatrix} 3 & 4 \end{bmatrix}}_{1 \times 2} = \underbrace{\begin{bmatrix} 6 & 8 \end{bmatrix}}_{1 \times 2}$	$\underbrace{2}_{\in \mathbb{R}} \underbrace{\begin{pmatrix} 3 & 4 \end{pmatrix}}_{\in (\mathbb{R}^2)^*} = \underbrace{\begin{pmatrix} 6 & 8 \end{pmatrix}}_{\in (\mathbb{R}^2)^*}$	$\lambda \mathbf{v}^\top$	scalar multiplication (covector version) $\mathbf{v}^\top \in \mathbb{R}^m, \lambda \in \mathbb{R}$
$\underbrace{\begin{bmatrix} 2 \end{bmatrix}}_{1 \times 1} \underbrace{\begin{bmatrix} 3 \end{bmatrix}}_{1 \times 1} = \underbrace{\begin{bmatrix} 6 \end{bmatrix}}_{1 \times 1}$	$\underbrace{2}_{\in \mathbb{R}} \underbrace{3}_{\in \mathbb{R}} = \underbrace{6}_{\in \mathbb{R}}$	$\lambda\mu$	standard multiplication $\lambda, \mu \in \mathbb{R}$

Table 2.2: “Mixed” multiplications that work like matrix multiplications

Definition 2.43 (Covector-matrix multiplication). *Let*

$$\mathbf{y}^\top = (y_1 \ y_2 \ \cdots \ y_m) \in (\mathbb{R}^m)^*, A = \begin{bmatrix} | & | & & | \\ \mathbf{v}_1 & \mathbf{v}_2 & \cdots & \mathbf{v}_n \\ | & | & & | \end{bmatrix} \in \mathbb{R}^{m \times n}.$$

The covector

$$\mathbf{y}^\top A = \underbrace{(\mathbf{y}^\top \mathbf{v}_1 \ \mathbf{y}^\top \mathbf{v}_2 \ \cdots \ \mathbf{y}^\top \mathbf{v}_n)}_{n \text{ scalar products}} \in (\mathbb{R}^n)^*$$

is the product of \mathbf{y}^\top and A .

By the definition of matrix multiplication via scalar products (Observation 2.38), this is indeed the covector version of the matrix multiplication

$$\underbrace{\begin{bmatrix} - & \mathbf{y}^\top & - \end{bmatrix}}_{1 \times m} A = \underbrace{\begin{bmatrix} \mathbf{y}^\top \mathbf{v}_1 & \mathbf{y}^\top \mathbf{v}_2 & \cdots & \mathbf{y}^\top \mathbf{v}_n \end{bmatrix}}_{1 \times n}.$$

Now for the outer product.

Definition 2.44. Let $\mathbf{v} \in \mathbb{R}^m$, $\mathbf{w} \in \mathbb{R}^n$. The outer product $\mathbf{v}\mathbf{w}^\top$ of \mathbf{v} and \mathbf{w} is the $m \times n$ matrix

$$\mathbf{v}\mathbf{w}^\top := \begin{bmatrix} v_1 w_1 & v_1 w_2 & \cdots & v_1 w_n \\ v_2 w_1 & v_2 w_2 & \cdots & v_2 w_n \\ \vdots & \vdots & \ddots & \vdots \\ v_m w_1 & v_m w_2 & \cdots & v_m w_n \end{bmatrix} = [v_i w_j]_{i=1, j=1}^{m, n}.$$

The outer product is sometimes also written as $\mathbf{v} \otimes \mathbf{w}$ but we will not use this notation.

This is the vector / covector version of the matrix multiplication

$$\begin{bmatrix} | \\ \mathbf{v} \\ | \end{bmatrix} \begin{bmatrix} - & \mathbf{w}^\top & - \end{bmatrix} = \begin{bmatrix} v_1 w_1 & v_1 w_2 & \cdots & v_1 w_n \\ v_2 w_1 & v_2 w_2 & \cdots & v_2 w_n \\ \vdots & \vdots & \ddots & \vdots \\ v_m w_1 & v_m w_2 & \cdots & v_m w_m \end{bmatrix},$$

as we can see from the definition of matrix multiplication in terms of scalar products (Observation 2.38). Here, $v_i w_j$ is the scalar product of the i -th row of \mathbf{v} (which has just one entry v_i) and the j -th column of \mathbf{w}^\top (which has also just one entry w_j).

The outer product provides a different way of thinking about rank-1 matrices: according to Lemma 2.15, a rank-1 matrix can now equivalently be characterized as the outer product of two vectors.

Using the arsenal of “mixed” multiplications from Table 2.2, we can form “mixed” expressions, involving matrices, vectors, covectors, and scalars (assuming the dimensions are such that all products are defined). A frequent example of such a mixed expression is

$$\mathbf{x}^\top A^\top A \mathbf{x},$$

where $\mathbf{x} \in \mathbb{R}^n$ is a vector, $\mathbf{x}^\top \in (\mathbb{R}^n)^*$ its transpose (a covector), A an $m \times n$ matrix, and A^\top its transpose (an $n \times m$ matrix). The result is a scalar.

We can evaluate such mixed expressions—using matrix multiplication—as if

- vectors in \mathbb{R}^m were $m \times 1$ matrices: $\begin{pmatrix} x_1 \\ x_2 \\ \vdots \\ x_m \end{pmatrix} = \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_m \end{bmatrix};$

- covectors in $(\mathbb{R}^n)^*$ were $1 \times n$ matrices: $(x_1 \ x_2 \ \cdots \ x_n) \text{''}=\text{''} [x_1 \ x_2 \ \cdots \ x_n]$; and
- scalars were 1×1 matrices: $\lambda \text{''}=\text{''} [\lambda]$.

Since it is only “as if”, the result may differ from the matrix multiplication result in the surrounding brackets. But associativity still holds, so evaluation order does not matter (Lemma 2.42), and when we transpose a product, we can instead transpose the individual factors and multiply them in reverse order (Lemma 2.40). Transposing a scalar has no effect, so $\lambda^\top = \lambda$.

For example, we can evaluate $\mathbf{x}^\top A^\top A \mathbf{x}$ as

$$\underbrace{\underbrace{\mathbf{x}^\top A}_{\text{covector}} \underbrace{A \mathbf{x}}_{\text{vector}}}_{\text{scalar}},$$

and according to Table 2.2, the result is the scalar product of the vectors $(\mathbf{x}^\top A)^\top = A \mathbf{x}$ (the transpose of the covector $\mathbf{x}^\top A$) and $A \mathbf{x}$; so we have the scalar product of a vector with itself. This means that

$$\mathbf{x}^\top A^\top A \mathbf{x} = \|A \mathbf{x}\|^2,$$

the squared Euclidean norm (length) of $A \mathbf{x}$; see Definition 1.11.

But we could also evaluate $\mathbf{x}^\top A^\top A \mathbf{x}$ as

$$\underbrace{\underbrace{\mathbf{x}^\top}_{\text{covector}} \underbrace{A^\top A}_{\text{matrix}}}_{\text{covector}} \underbrace{\mathbf{x}}_{\text{vector}} \quad \text{or} \quad \underbrace{\underbrace{\mathbf{x}^\top}_{\text{covector}} \underbrace{A^\top A}_{\text{matrix}}}_{\text{vector}} \underbrace{\mathbf{x}}_{\text{vector}}$$

In all cases, the result is the same. Let us look at another example, $\mathbf{v}^\top \mathbf{w} \mathbf{v}^\top \mathbf{w}$ where $\mathbf{v}, \mathbf{w} \in \mathbb{R}^m$. This is

$$\underbrace{\underbrace{\mathbf{v}^\top \mathbf{w}}_{\text{scalar}} \underbrace{\mathbf{v}^\top \mathbf{w}}_{\text{scalar}}}_{\text{scalar}},$$

the square of the scalar product $\mathbf{v}^\top \mathbf{w}$. But we could also evaluate this as

$$\underbrace{\underbrace{\mathbf{v}^\top}_{\text{covector}} \underbrace{\mathbf{w} \mathbf{v}^\top}_{\text{matrix}}}_{\text{vector}} \underbrace{\mathbf{w}}_{\text{vector}}.$$

Although the final result is the same, we better not compute it this way: as an intermediate step, this evaluation order computes an $m \times m$ matrix, the outer product $\mathbf{w} \mathbf{v}^\top$ of \mathbf{w} and \mathbf{v} . If $m = 1,000$, this matrix has 1,000,000 (one million) entries, so its computation alone requires 1,000,000 multiplications $w_i v_j$, one for each entry. Using the first evaluation order, the final result $(\mathbf{v}^\top \mathbf{w})^2$ can be computed with 1,001 multiplications only.

As everything “is” matrix multiplication, the general question behind this is the following: given k matrices A_1, A_2, \dots, A_k of dimensions $a \times n_1, n_1 \times n_2, \dots, n_{k-1} \times b$, what is the most efficient evaluation order for computing their product, the $a \times b$ matrix $A_1 A_2 \dots A_k$? In other words, how shall we put the brackets so that we need the least effort? Already for $k = 3$, we can convince ourselves that the two possible evaluation orders $(A_1 A_2) A_3$ and $A_1 (A_2 A_3)$ usually require different efforts, so which one is more efficient? As k grows, the number of possible evaluation orders quickly becomes too large to assess the effort for all of them individually, so we need to be smarter. Fortunately, there is a *dynamic programming* algorithm that—given $a, n_1, n_2, \dots, n_{k-1}, b$ —finds the most efficient evaluation order quickly. This is covered in algorithms textbooks; see for example *Introduction to Algorithms* [CLRS22, Section 14.2].

Everything “is” composition. Table 2.2 presents several kinds of multiplications that look like matrix multiplications in the sense that they produce the same numbers; but since they involve vectors, covectors, and scalars, they are logically quite different from matrix multiplications. But we can develop a unifying view of all these multiplications that also explains why the same numbers result from logically different multiplications. The key insight is that all objects involved (matrices, vectors, covectors, and scalars) correspond to transformations in a natural way, and with the property that all the multiplications in Table 2.2 correspond to compositions of these transformations. Table 2.3 presents the transformations.

Object			Corresponding transformation				
Matrix	A	$\in \mathbb{R}^{m \times n}$	T_A	$:$	$\mathbb{R}^n \rightarrow \mathbb{R}^m$,	$\mathbf{x} \mapsto A\mathbf{x}$
Vector	\mathbf{v}	$\in \mathbb{R}^m$	$T_{\mathbf{v}}$	$:$	$\mathbb{R} \rightarrow \mathbb{R}^m$,	$x \mapsto x\mathbf{v}$
Covector	\mathbf{v}^\top	$\in \mathbb{R}^m$	$T_{\mathbf{v}^\top}$	$:$	$\mathbb{R}^m \rightarrow \mathbb{R}$,	$\mathbf{x} \mapsto \sum_{i=1}^m v_i x_i$
Scalar	λ	$\in \mathbb{R}$	T_λ	$:$	$\mathbb{R} \rightarrow \mathbb{R}$,	$x \mapsto \lambda x$

Table 2.3: Transformations corresponding to matrices, vectors, covectors, and scalars

For covectors, this is nothing new, since a covector \mathbf{v}^\top is by Definition 1.20 already the transformation $T_{\mathbf{v}^\top}$ in Table 2.3. For a matrix A , the corresponding transformation in the table is also well-known to us: it is the matrix transformation T_A , see Definition 2.18. In case of vectors and scalars, we were so far not familiar with their corresponding transformations. For example, if $\mathbf{v} = \begin{pmatrix} 1 \\ 2 \end{pmatrix} \in \mathbb{R}^2$, then

$$T_{\mathbf{v}} : \mathbb{R} \rightarrow \mathbb{R}^2, \quad x \mapsto x \begin{pmatrix} 1 \\ 2 \end{pmatrix} = \begin{pmatrix} x \\ 2x \end{pmatrix}. \quad (2.2)$$

Now the following can be proved (but we will not do this): whenever α and β are some objects such that their product $\alpha\beta$ is defined according to Table 2.2, it holds that

$$T_{\alpha\beta} = T_\alpha \circ T_\beta.$$

This means that the composition of T_α and T_β gives the transformation corresponding to the product of α and β .

If $\alpha = A$ and $\beta = B$ are matrices, we already know that $T_{AB} = T_A \circ T_B$, see Corollary 2.37, because we have defined the product AB precisely in such a way that this holds; see Definition 2.36.

For the other products in Table 2.2, they also have to be like that in order to fulfill $T_{\alpha\beta} = T_\alpha \circ T_\beta$ in all cases. Once we have this, associativity of mixed expressions (all evaluation orders lead to the same final result) follows in the same way as previously for matrix multiplication; see the proof of Lemma 2.42 (ii).

We can also see from Table 2.3 that the transformations corresponding to flat matrices differ from the ones for scalars, vectors, and covectors only in a small detail: whenever the former have \mathbb{R}^1 as domain or codomain, it is \mathbb{R} for the latter. Formally, $\mathbb{R} = \{\lambda : \lambda \in \mathbb{R}\}$ and $\mathbb{R}^1 = \{(\lambda) : \lambda \in \mathbb{R}\}$ are different sets, but the difference is just a matter of putting brackets or not around a number; this explains why in Table 2.2, the flat matrix multiplications produce the same numbers as their corresponding vector / covector / scalar versions, and only the brackets are different. Taking up the example in (2.2), here is the flat matrix version of it: if $A = \begin{bmatrix} 1 \\ 2 \end{bmatrix}$, then

$$T_A : \mathbb{R}^1 \rightarrow \mathbb{R}^2, \quad (x) \mapsto \begin{bmatrix} 1 \\ 2 \end{bmatrix} (x) = \begin{pmatrix} x \\ 2x \end{pmatrix}.$$

We can now also understand which mixed expressions are well-formed and what their types are. For example, if $\mathbf{x} \in \mathbb{R}^n$ and A is an $m \times n$ matrix, we can write $\mathbf{x}^\top A^\top A \mathbf{x}$, because

$$T_{\mathbf{x}^\top A^\top A \mathbf{x}} : \mathbb{R} \leftarrow \mathbb{R}^n \leftarrow \mathbb{R}^m \leftarrow \mathbb{R}^n \leftarrow \mathbb{R}.$$

In the composition $T_{\mathbf{x}^\top A^\top A \mathbf{x}} = T_{\mathbf{x}^\top} \circ T_{A^\top} \circ T_A \circ T_{\mathbf{x}}$, the codomain of each transformation equals the domain of the next transformation (left of it). As the result is $T_{\mathbf{x}^\top A^\top A \mathbf{x}} : \mathbb{R} \rightarrow \mathbb{R}$, Table 2.3 tells us that $\mathbf{x}^\top A^\top A \mathbf{x}$ is a scalar.

Considering just $\mathbf{x}^\top A^\top$, we have

$$T_{\mathbf{x}^\top A^\top} : \mathbb{R} \leftarrow \mathbb{R}^n \leftarrow \mathbb{R}^m,$$

so $T_{\mathbf{x}^\top A^\top} : \mathbb{R}^m \rightarrow \mathbb{R}$, and $\mathbf{x}^\top A^\top \in (\mathbb{R}^m)^*$ is a covector by Table 2.3.

But the mixed expression $\mathbf{x} A^\top$, for example, is *ill-formed* (incorrectly structured) and does not correspond to a multiplication in Table 2.2. This is because we get a mismatch between the codomain of T_{A^\top} and the domain of $T_{\mathbf{x}}$:

$$T_{\mathbf{x} A^\top} : \mathbb{R}^n \leftarrow \mathbb{R} \neq \mathbb{R}^n \leftarrow \mathbb{R}^m.$$

We conclude this section with yet another way to do matrix multiplication, namely via covector-matrix multiplications. Recall that Definition 2.36 has introduced matrix

multiplication via matrix-vector multiplication, expressing the result in column notation. Symmetrically, we can now also define it via covector-matrix multiplication and express the result in row notation.

Observation 2.45 (Matrix multiplication in row notation). *Let*

$$A = \begin{bmatrix} - & \mathbf{u}_1^\top & - \\ - & \mathbf{u}_2^\top & - \\ & \vdots & \\ - & \mathbf{u}_a^\top & - \end{bmatrix}$$

be an $a \times n$ matrix and B an $n \times b$ matrix. The product AB is the $a \times b$ matrix

$$\begin{bmatrix} - & \mathbf{u}_1^\top B & - \\ - & \mathbf{u}_2^\top B & - \\ & \vdots & \\ - & \mathbf{u}_a^\top B & - \end{bmatrix}. \quad (2.3)$$

Proof. Writing B in column notation as

$$B = \begin{bmatrix} | & | & \cdots & | \\ \mathbf{x}_1 & \mathbf{x}_2 & \cdots & \mathbf{x}_b \\ | & | & \cdots & | \end{bmatrix},$$

Definition 2.43 of covector-matrix multiplication shows that row i of (2.3) is

$$[\mathbf{u}_i^\top \mathbf{x}_1 \quad \mathbf{u}_i^\top \mathbf{x}_2 \quad \cdots \quad \mathbf{u}_i^\top \mathbf{x}_n],$$

and this is exactly what AB has in row i according to matrix multiplication in scalar product form (Observation 2.38). \square

2.3.5 CR decomposition

After having talked a lot about matrix transformations and linear transformations, we come back to some “pure” matrix theory as we have started it in Section 2.1.

You know the prime factor decomposition of a natural number. For example, this writes the number 1001 as

$$1001 = 7 \cdot 11 \cdot 13.$$

This decomposition tells you something about the “structure” of the number that you cannot easily tell from just looking at the number.

In linear algebra, it is mostly matrix decompositions that are of interest. This means, we want to write a matrix A as a product of other matrices, with the goal of learning more about A , and potentially speeding up computations involving A .

In this section, we will see a first such decomposition.

Theorem 2.46 (CR decomposition). *Let A be an $m \times n$ matrix of rank r (Definition 2.10). Let C be the $m \times r$ submatrix of A containing the independent columns. Then there is a unique $r \times n$ matrix R' such that*

$$A = CR'.$$

It will become clear in Theorem 3.18 why we call the matrix R' and not simply R . Before we go to the proof, let us do an example. Consider the rank-1 matrix

$$A = \begin{bmatrix} 2 & 4 & 6 \\ 3 & 6 & 9 \end{bmatrix}$$

with one independent column (the first one). In this case, the CR decomposition is

$$A = \underbrace{\begin{bmatrix} 2 \\ 3 \end{bmatrix}}_{C, 2 \times 1} \underbrace{\begin{bmatrix} 1 & 2 & 3 \end{bmatrix}}_{R', 1 \times 3}.$$

The columns of R' contain the scalars that we need in order to write each column as a scalar multiple of the first one. Something very similar is going on in general.

Proof of Theorem 2.46. We know that A and C have the same column space (Lemma 2.11), so every column of A can be written as a linear combination of the columns of C . If A has columns $\mathbf{v}_1, \mathbf{v}_2, \dots, \mathbf{v}_n \in \mathbb{R}^m$, we therefore have $\mathbf{v}_j = C\mathbf{x}_j$ for all j , where $\mathbf{x}_j \in \mathbb{R}^r$ is a vector of r scalars. This uses that matrix-vector multiplication is simply another notation for linear combinations; see Definition 2.4. But then we get

$$A = \begin{bmatrix} | & | & \cdots & | \\ \mathbf{v}_1 & \mathbf{v}_2 & \cdots & \mathbf{v}_n \\ | & | & \cdots & | \end{bmatrix} = C \underbrace{\begin{bmatrix} | & | & \cdots & | \\ \mathbf{x}_1 & \mathbf{x}_2 & \cdots & \mathbf{x}_n \\ | & | & \cdots & | \end{bmatrix}}_{R' \in \mathbb{R}^{r \times n}} = CR',$$

using Definition 2.36 of matrix multiplication. So we have a decomposition of the required form, where the matrix C contains the independent columns of A ; the equation $A = CR'$ means that the matrix R' contains scalars that express all columns as linear combinations of the independent columns. Since the columns of C are linearly independent, there are unique such scalars by Lemma 1.24, so there is a unique matrix R' that satisfies the equation $A = CR'$. \square

Here is an example where A has rank 2:

$$\underbrace{\begin{bmatrix} 1 & 2 & 0 & 3 \\ 2 & 4 & 1 & 4 \\ 3 & 6 & 2 & 5 \end{bmatrix}}_{A, 3 \times 4} = \underbrace{\begin{bmatrix} 1 & 0 \\ 2 & 1 \\ 3 & 2 \end{bmatrix}}_{C, 3 \times 2} \underbrace{\begin{bmatrix} 1 & 2 & 0 & 3 \\ 0 & 0 & 1 & -2 \end{bmatrix}}_{R', 2 \times 4}.$$

To see that this is the CR decomposition, we look at the matrix A column by column. Each of the four columns $\mathbf{v}_j, j = 1, 2, 3, 4$, is either independent (and then $\mathbf{v}_j = 1\mathbf{v}_j$ is the unique linear combination), or it is dependent and therefore a unique linear combination of the previous independent columns. In the example, the first and the third column are independent, a fact that you can simply believe at this point. We do not yet know how to systematically find the independent columns of a matrix, and the scalars that express a dependent column as a linear combination of the previous independent ones.

		columns of A			
		\mathbf{v}_1	\mathbf{v}_2	\mathbf{v}_3	\mathbf{v}_4
$A = \begin{bmatrix} 1 & 2 & 0 & 3 \\ 2 & 4 & 1 & 4 \\ 3 & 6 & 2 & 5 \end{bmatrix}$	\parallel	\parallel	\parallel	\parallel	
	$\begin{bmatrix} 1 \\ 2 \\ 3 \end{bmatrix}$	$\begin{bmatrix} 2 \\ 4 \\ 6 \end{bmatrix}$	$\begin{bmatrix} 0 \\ 1 \\ 2 \end{bmatrix}$	$\begin{bmatrix} 3 \\ 4 \\ 5 \end{bmatrix}$	
	\parallel	\parallel	\parallel	\parallel	
	\mathbf{v}_1	$1\mathbf{v}_1$	$2\mathbf{v}_1$	$3\mathbf{v}_1$	
	\mathbf{v}_2				
	\mathbf{v}_3		$1\mathbf{v}_3$	$-2\mathbf{v}_3$	
	\mathbf{v}_4				
	independent?	yes	no	yes	no

The resulting CR decomposition is therefore indeed

$$\underbrace{\begin{bmatrix} 1 & 2 & 0 & 3 \\ 2 & 4 & 1 & 4 \\ 3 & 6 & 2 & 5 \end{bmatrix}}_{A, 3 \times 4} = \underbrace{\begin{bmatrix} 1 & 0 \\ 2 & 1 \\ 3 & 2 \end{bmatrix}}_{C, 3 \times 2} \underbrace{\begin{bmatrix} 1 & 2 & 0 & 3 \\ 0 & 0 & 1 & -2 \end{bmatrix}}_{R', 2 \times 4}.$$

We will see in Theorem 3.18 how the CR decomposition can systematically be computed; in fact, we get it as by-product of transforming A into a *standard form* R . Lemma 4.19 and Theorem 4.32 interpret C and R' as compact representations of the column space and the row space of A . Here, we conclude with a “data compression” view of the CR decomposition.

Matrix compression. An $m \times n$ matrix A has mn entries, and if m and n are large, mn can be *very* large. As a result, storing A or transmitting A over some network link might be quite costly. If A has low rank r , the CR decomposition provides a *compression* of A such that it takes up less space. To compress A , we do not store A but the two matrices C and R' of its CR decomposition. From C and R' , we can reconstruct A whenever we want via the matrix multiplication $A = CR'$; so no information is lost. But since C has mr entries, and R' has rn entries, we need to store only $(m+n)r$ entries instead of mn . This is not always a saving (in the example above it is not), but if r is small compared to m and n , the savings can be substantial.

For example, if $m = n = 1,000$, A has 1,000,000 (one million) entries. Now assume that A has only rank 10. Then C and R' together only have $2,000 \cdot 10 = 20,000$ entries. This is fifty times less than A has.

Exercise 2.47. What are the matrices C and R' in Theorem 2.46 if A is an $m \times m$ matrix of rank m (the columns are linearly independent)? What are the matrices C and R' if A is the $m \times n$ zero matrix? The second question also requires you to think about $m \times 0$ and $0 \times n$ matrices. While these are not extremely relevant in practice, it is good to make sure that our definitions and proofs also work for them.

2.4 Invertible and Inverse matrices

We introduce the important class of invertible matrices, the ones that lead to an “undoable” matrix transformation $T_A : \mathbf{x} \mapsto A\mathbf{x}$. It turns out that these are exactly the square matrices with linearly independent columns. Moreover, the transformation that is undoing T_A is also a matrix transformation, and we call its matrix the inverse of A . The inverse A^{-1} of an invertible matrix A is the higher-dimensional analog of the reciprocal $1/a$ of a nonzero real number a that is also written as a^{-1} .

2.4.1 Undoing matrix transformations

In Figures 2.6 and 2.7, we have seen four matrix transformations $T_A : \mathbb{R}^2 \rightarrow \mathbb{R}^2$, and what they geometrically “do” (scale the input, mirror it, rotate it, or shear it). In each of these four cases, we can “undo” the action (i.e. transform the output back to the input) with another matrix transformation whose matrix we call A^{-1} ; Figure 2.13 shows this for the scaling and mirroring transformations from Figure 2.6. You are invited to also find the matrices A^{-1} for the transformations from Figure 2.7.

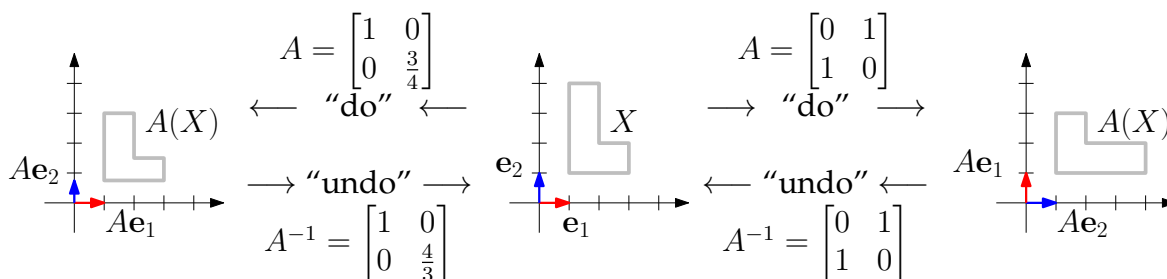


Figure 2.13: Left: To undo stretching by a factor of $3/4$, stretch by the reciprocal factor of $4/3$! Right: to undo mirroring, mirror again!

Not every matrix transformation is undoable. Consider $A = 0$ in which case $T_A(\mathbf{x}) = \mathbf{0}$ for all \mathbf{x} . Since the output is always $\mathbf{0}$, we cannot uniquely reconstruct the input.

Before we try to understand which matrix transformations are undoable, let us formalize the concept of undoable functions in general.

Definition 2.48 (Injective, surjective, and bijective functions). *Let X, Y be sets and $f : X \rightarrow Y$ a function.*

- (i) *f is called injective if for every $y \in Y$, there is at most one $x \in X$ with $f(x) = y$.
("For every possible output, at most one input leads to it.")*
- (ii) *f is called surjective if for every $y \in Y$, there is at least one $x \in X$ with $f(x) = y$.
("For every possible output, at least one input leads to it.")*
- (iii) *f is called bijective (undoable) if f is both injective and surjective.
("For every possible output, exactly one input leads to it.")*
- (iv) *The inverse of a bijective function f is the function*

$$f^{-1} : Y \rightarrow X, \quad y \mapsto \text{the unique } x \in X \text{ such that } f(x) = y.$$

If $f(x) = y$, then $f^{-1}(y) = x$ by definition of f^{-1} , so f^{-1} is indeed undoing f . This can also be written as $f^{-1}(f(x)) = x$ for all $x \in X$, or (using Definition 2.33 of composition) as $f^{-1} \circ f = \text{id}$, where $\text{id} : X \rightarrow X, x \mapsto x$ is the *identity function* (in abuse of notation, we use the same symbol id for every domain).

Vice versa, if $f^{-1}(y) = x$, then $f(x) = y$ by definition of f^{-1} , so f is undoing f^{-1} , as you may have expected. This can also be written as $f(f^{-1}(y)) = y$ for all $y \in Y$, or $f \circ f^{-1} = \text{id}$.

Along these lines, it is easy to prove that f^{-1} is also bijective, and that its inverse is f . We summarize this in the following fact.

Fact 2.49 (Bijective functions and their inverses). *If $f : X \rightarrow Y$ is bijective, then $f^{-1} : Y \rightarrow X$ is also bijective, and $(f^{-1})^{-1} = f$. Moreover, $f^{-1} \circ f = \text{id}$ (f^{-1} is undoing f) and $f \circ f^{-1} = \text{id}$ (f is undoing f^{-1}).*

Next we look at matrix transformations $T_A : \mathbb{R}^n \rightarrow \mathbb{R}^m$. If $m \neq n$, T_A can never be bijective. We separately prove this for the two cases $m < n$ (when A is wide) and $m > n$ (when A is tall); see Figure 2.1 for these matrix shapes.

Lemma 2.50 (Wide matrix transformations are not injective). *Let $T_A : \mathbb{R}^n \rightarrow \mathbb{R}^m$ be a matrix transformation, and $m < n$. Then T_A is not injective (and therefore also not bijective).*

Intuitively, this seems clear. If $m < n$, then T_A is "squeezing" a high-dimensional space \mathbb{R}^n into a low dimensional space \mathbb{R}^m , in which case overlaps (inputs with the same output) seem unavoidable. One has to be careful with this intuition, though. It is true for "nice" functions such as matrix transformations, but not for all functions.

Proof of Lemma 2.50. We show that there is more than one input leading to output $\mathbf{0}$. Towards this, the main thing to observe is that a wide matrix A has linearly dependent columns. To see this, we consider the first m columns. Either these are already linearly dependent, so all columns are linearly dependent; or the first m columns are linearly

independent. But then they span \mathbb{R}^m by Lemma 1.28, so the next column is a linear combination of the first m columns, and again, all columns are linearly dependent.

Knowing this, we can express $\mathbf{0}$ as a nontrivial linear combination of the columns, so we have $T_A(\mathbf{x}) = A\mathbf{x} = \mathbf{0}$ for some nonzero vector $\mathbf{x} \in \mathbb{R}^n$ of scalars; see also Observation 2.5 (ii). But this shows that T_A is not injective: we have two different inputs, $\mathbf{0}$ and some $\mathbf{x} \neq \mathbf{0}$, leading to the same output $T_A(\mathbf{0}) = T_A(\mathbf{x}) = \mathbf{0}$. \square

Lemma 2.51 (Tall matrix transformations are not surjective). *Let $T_A : \mathbb{R}^n \rightarrow \mathbb{R}^m$ be a matrix transformation, and $m > n$. Then T_A is not surjective (and therefore also not bijective).*

Again, this seems quite intuitive: If $m > n$, then T_A is “embedding” a low-dimensional space \mathbb{R}^n into a high-dimensional space \mathbb{R}^m , but such an embedding cannot fill the whole space (think of a line or a plane in 3-dimensional space). Also here, this intuition is only true for nice functions.

Proof of Lemma 2.51. We construct a vector $\mathbf{y} \in \mathbb{R}^m$ such that no input leads to output \mathbf{y} . As we argued in the previous proof of Lemma 2.50, every wide matrix has linearly dependent columns; this in particular applies to A^\top , the transpose of A , so we can express $\mathbf{0}$ as a nontrivial linear combination of the columns of A^\top . In other words, there is a nonzero vector $\mathbf{y} \in \mathbb{R}^m$ such that $A^\top \mathbf{y} = \mathbf{0}$. We now want to show that there is no input $\mathbf{x} \in \mathbb{R}^n$ with output $T_A(\mathbf{x}) = A\mathbf{x} = \mathbf{y}$.

For this, we use a *proof by contradiction*. This works as follows. If we want to prove that some statement is true, we first assume the opposite. If we can logically derive some nonsense from this assumption, we know that the assumption must have been false; so the statement is true after all. The full story is quite a bit more subtle than we tell it here, but we will not go into this and simply use the proof by contradiction principle.

In our case, we want to prove that there is no \mathbf{x} such that $A\mathbf{x} = \mathbf{y}$, so let us first assume the opposite, namely that there is such an \mathbf{x} . Also recall that we have constructed $\mathbf{y} \neq \mathbf{0}$ such that $A^\top \mathbf{y} = \mathbf{0}$. Now we consider the number

$$\mathbf{y}^\top A\mathbf{x}.$$

This is the product of a covector, a matrix, and a vector; we refer back to Section 2.3.4 where we have introduced such “mixed” multiplications and explained how they behave like matrix multiplications. Hence, using associativity and the transpose-of-a-product Lemma 2.40, we get

$$\mathbf{y}^\top A\mathbf{x} = \underbrace{\mathbf{y}^\top A}_{(A^\top \mathbf{y})^\top = \mathbf{0}^\top} \mathbf{x} = \mathbf{0}^\top \mathbf{x} = 0.$$

On the other hand, by our opposite assumption and using the other evaluation order, we get

$$\mathbf{y}^\top A\mathbf{x} = \mathbf{y}^\top \underbrace{A\mathbf{x}}_{\mathbf{y}} = \mathbf{y}^\top \mathbf{y} = \|\mathbf{y}\|^2 \neq 0,$$

since $\mathbf{y} \neq \mathbf{0}$, so its Euclidean norm (length; see Definition 1.11) is also nonzero. But now we have logically derived the desired nonsense: $\mathbf{y}^\top A\mathbf{x}$ is at the same time zero and nonzero.

So the assumption that some input \mathbf{x} leads to output $T_A(\mathbf{x}) = A\mathbf{x} = \mathbf{y}$ must have been false, and in reality no input leads to output \mathbf{y} . Hence, T_A is not surjective. \square

As a consequence of Lemma 2.50 and Lemma 2.51, the only candidates for bijective (undoable) matrix transformations are the ones of the form $T_A : \mathbb{R}^m \rightarrow \mathbb{R}^m$ (for which A is an $m \times m$ square matrix). Not every $m \times m$ matrix A leads to a bijective T_A (consider $A = 0 \in \mathbb{R}^{m \times m}$ for $m > 0$ as the easiest counterexample).

Now we want to show that *if* such a T_A is bijective, then the inverse function $T_A^{-1} : \mathbb{R}^m \rightarrow \mathbb{R}^m$ is again a matrix transformation. Since matrix transformations and linear transformations are the same objects (Observation 2.22 and Theorem 2.26), we argue with linear transformations, which saves us from having to “invent” the matrix of T_A^{-1} manually.

Lemma 2.52 (The inverse of a bijective linear transformation). *Let $T : \mathbb{R}^m \rightarrow \mathbb{R}^m$ be a bijective linear transformation. Then its inverse $T^{-1} : \mathbb{R}^m \rightarrow \mathbb{R}^m$ is also a linear transformation (and bijective by Fact 2.49).*

Proof. We prove linearity of T^{-1} according to Definition 2.21. For this, we must show that

$$T^{-1}(\lambda_1 \mathbf{y}_1 + \lambda_2 \mathbf{y}_2) = \lambda_1 T^{-1}(\mathbf{y}_1) + \lambda_2 T^{-1}(\mathbf{y}_2), \quad (2.4)$$

for all $\mathbf{y}_1, \mathbf{y}_2 \in \mathbb{R}^m$ and all $\lambda_1, \lambda_2 \in \mathbb{R}$. To see this, we define $\mathbf{x}_1 := T^{-1}(\mathbf{y}_1)$, $\mathbf{x}_2 := T^{-1}(\mathbf{y}_2)$. Since T is undoing T^{-1} (Fact 2.49), we have $T(\mathbf{x}_1) = \mathbf{y}_1$ and $T(\mathbf{x}_2) = \mathbf{y}_2$. Linearity of T thus gives

$$T(\lambda_1 \underbrace{T^{-1}(\mathbf{y}_1)}_{\mathbf{x}_1} + \lambda_2 \underbrace{T^{-1}(\mathbf{y}_2)}_{\mathbf{x}_2}) = \lambda_1 \underbrace{\mathbf{y}_1}_{T(\mathbf{x}_1)} + \lambda_2 \underbrace{\mathbf{y}_2}_{T(\mathbf{x}_2)}.$$

Applying T^{-1} to both sides is undoing T on the left side (Fact 2.49, again), and we obtain

$$\lambda_1 T^{-1}(\mathbf{y}_1) + \lambda_2 T^{-1}(\mathbf{y}_2) = T^{-1}(\lambda_1 \mathbf{y}_1 + \lambda_2 \mathbf{y}_2).$$

This is the desired linearity (2.4) of T^{-1} . \square

Using this, we can now prove the following result that characterizes the undoable matrix transformations in terms of matrix properties.

Lemma 2.53 (Bijective matrix transformations). *Let A be an $m \times m$ matrix. The following three statements are equivalent.*

- (i) $T_A : \mathbb{R}^m \rightarrow \mathbb{R}^m$ is bijective.
- (ii) There is an $m \times m$ matrix B such that $BA = I$.
- (iii) The columns of A are linearly independent.

Proof. We prove this via the three circular implications (i) \Rightarrow (ii) \Rightarrow (iii) \Rightarrow (i).

(i) \Rightarrow (ii): If T_A is bijective, it has an inverse T_A^{-1} , see Definition 2.48 (iv). Since T_A is a linear transformation by Observation 2.22, we know from Lemma 2.52 that T_A^{-1} is another linear transformation and therefore also a matrix transformation T_B by Theorem 2.26.

For a bijective function f , we have $f^{-1} \circ f = \text{id}$ (f^{-1} is undoing f ; see Fact 2.49). Here, this reads as

$$\underbrace{T_B \circ T_A}_{\text{Cor. 2.37 } T_{BA}} = \underbrace{\text{id}}_{T_I}.$$

This writes the same linear transformation as a matrix transformation in two different ways, and by Theorem 2.26, both ways must result in the same matrix. So we get

$$BA = I.$$

(ii) \Rightarrow (iii): If there is an $m \times m$ matrix B such that $BA = I$, we will argue that $\mathbf{x} = \mathbf{0}$ is the only vector such that $A\mathbf{x} = \mathbf{0}$. This means that the columns of A are linearly independent; see Observation 2.5 (ii). For the argument, consider any $\mathbf{x} \in \mathbb{R}^m$ with $A\mathbf{x} = \mathbf{0}$. We then get $\mathbf{x} = (BA)\mathbf{x} = B(A\mathbf{x}) = B\mathbf{0} = \mathbf{0}$.

(iii) \Rightarrow (i): If the columns of A are linearly independent, we will deduce that T_A is both injective and surjective, and hence bijective by Definition 2.48. Let us start with surjectivity: given any possible output $\mathbf{y} \in \mathbb{R}^m$, we need to find some input $\mathbf{x} \in \mathbb{R}^m$ such that $T_A(\mathbf{x}) = A\mathbf{x} = \mathbf{y}$. In other words, we need to express \mathbf{y} as a linear combination of the columns of A ; see Observation 2.5 (i). But this can be done since the m linearly independent columns of A span the whole space \mathbb{R}^m by Lemma 1.28.

For injectivity, we still need to show that there is only one input leading to output \mathbf{y} . In other words, \mathbf{y} can be expressed as a linear combination of the columns of A in only one way. This is true by Lemma 1.24. \square

Because matrix multiplication is in general not commutative, the following is an interesting property.

Lemma 2.54. *Let A, B be $m \times m$ matrices such that $BA = I$. Then also $AB = I$.*

Proof. By Lemma 2.53, $BA = I$ means that T_A is bijective, in particular surjective. So for every $\mathbf{y} \in \mathbb{R}^m$ we have some $\mathbf{x} \in \mathbb{R}^m$ with $A\mathbf{x} = \mathbf{y}$ and therefore get

$$AB\mathbf{y} = AB(A\mathbf{x}) = A(BA)\mathbf{x} = A\mathbf{x} = \mathbf{y}.$$

So $T_{AB} : \mathbb{R}^m \rightarrow \mathbb{R}^m$ is the (linear) identity transformation T_I , and since a linear transformation has a unique matrix by Theorem 2.26, we must have $AB = I$. \square

2.4.2 Definitions and basic properties

Lemma 2.53 characterizes the matrices whose transformations are undoable. Here we give those matrices a name and look into their properties.

Definition 2.55 (Invertible and singular matrix). Let A be an $m \times m$ matrix. A is called invertible if there is an $m \times m$ matrix B such that $BA = I$. (By Lemma 2.53 this is the case if and only if A has linearly independent columns.) Otherwise, A is called singular.

It is also fine to call a singular matrix *non-invertible*. You also see invertible matrices being called *non-singular*. In some sources, the condition $AB = I$ is used instead, in yet others, it is $AB = BA = I$. Because of Lemma 2.54, this makes no difference.

Observation 2.56 (Alternative definitions of an invertible matrix). Let A be an $m \times m$ matrix. A is invertible if and only if there is an $m \times m$ matrix B satisfying one (and therefore all) of the following three equivalent conditions.

(i) $AB = I$.

(ii) $BA = I$. (This means that A is invertible according to Definition 2.55.)

(iii) $AB = BA = I$.

Moreover, the matrix B is unique.

Proof. By Lemma 2.54, we have (i) \Leftrightarrow (ii), and via this also get (i) \Rightarrow (iii) and (ii) \Rightarrow (iii). The reverse implications (iii) \Rightarrow (i) and (iii) \Rightarrow (ii) are obvious, so we get equivalence of all three conditions: (i) \Leftrightarrow (iii) \Leftrightarrow (ii). To see uniqueness, consider two matrices B, B' satisfying the conditions. Then

$$B = IB \stackrel{(ii)}{=} (B'A)B = B'(AB) \stackrel{(i)}{=} B'I = B'.$$

□

We can now summarize the situation and define the inverse of an invertible matrix.

Definition 2.57 (Inverse matrix). Let A be an $m \times m$ matrix. A is invertible if and only if there exists an $m \times m$ matrix B such that $BA = I$ (or $AB = I$, or $AB = BA = I$). In this case, the matrix B is unique and called the inverse of A . We denote it by A^{-1} .

As a consequence, A^{-1} is also invertible, and its inverse is the original matrix A .

Observation 2.58 (Inverse of the inverse). Let A be an invertible $m \times m$ matrix. Then A^{-1} is also invertible and

$$(A^{-1})^{-1} = A.$$

Proof. We have $AA^{-1} = A^{-1}A = I$ by Definition 2.57, so A satisfies the conditions for the inverse of A^{-1} . □

Let us look at the situation for small m . If $m = 1$, then $A = [a]$ for some $a \in \mathbb{R}$, and $I = [1]$. According to Definition 2.57, the invertibility condition is that there is some $B = [b]$, $b \in \mathbb{R}$, such that

$$\underbrace{\begin{bmatrix} b \end{bmatrix} \begin{bmatrix} a \end{bmatrix}}_{[ba]} = \begin{bmatrix} 1 \end{bmatrix}.$$

In other words, $ba = 1$, so b must be the reciprocal of a which exists exactly if $a \neq 0$. Hence, $[a]$ is invertible exactly if $a \neq 0$, and then its inverse is $[1/a]$.

Case 1×1 .

$$A = [a] \Rightarrow A^{-1} = \left[\frac{1}{a}\right] \quad (\text{if } a \neq 0).$$

For $m = 2$, the situation is as follows.

Case 2×2 .

$$A = \begin{bmatrix} a & b \\ c & d \end{bmatrix} \Rightarrow A^{-1} = \frac{1}{ad - bc} \begin{bmatrix} d & -b \\ -c & a \end{bmatrix} \quad (\text{if } ad - bc \neq 0).$$

To see that this formula for A^{-1} is correct, you can simply check that A^{-1} satisfies one of the conditions for B in Definition 2.57. Again, A is not always invertible, but the restriction given here ($ad - bc \neq 0$) is less obvious than for $m = 1$. It is not hard to see (try it!) that this condition—expressed in words—reads as follows: The two columns of A are linearly independent.

If $ad - bc = 0$, we only see that the given formula for A^{-1} fails due to division by zero, but there could theoretically be another formula. But putting together a few things that we already know, we see that A is indeed singular if $ad - bc = 0$: in this case, the columns of A are linearly dependent, so they span at most a line; see for example Figure 1.23 (left). Any $\mathbf{b} \in \mathbb{R}^2$ outside of that line is not a linear combination of the columns, so there are no scalars $\mathbf{x} \in \mathbb{R}^2$ with $A\mathbf{x} = \mathbf{b}$; see Observation 2.5. In still other words, the matrix transformation $T_A : \mathbf{x} \mapsto A\mathbf{x}$ cannot produce output \mathbf{b} , so it is not surjective and therefore also not bijective (undoable). Then Lemma 2.53 tells us that there can be no matrix B satisfying the conditions in Definition 2.57.

It is possible to check directly that there is no matrix B with $BA = I$ if $ad - bc = 0$; this boils down to showing that a system of linear equations in four variables (the entries of B) has no solution. While this can certainly be done in some ad-hoc way, it is not very insightful. The systematic approach to check for invertibility goes via the concept of *determinants* to which we will only get in the second part of the lecture.

For invertible $m \times m$ matrices with $m > 2$, there is also a formula for the inverse, based on *Cramer's rule*, but also this requires determinants.

The next lemma lets us compute the inverse of a *product* of two invertible matrices.

Lemma 2.59. *Let A and B be invertible $m \times m$ matrices. Then AB is also invertible, and*

$$(AB)^{-1} = B^{-1}A^{-1}.$$

Recall that AB is the matrix of the linear transformation $T_{AB} = T_A \circ T_B$ (first do T_B , then T_A); see Corollary 2.37. To undo this, we need to apply the inverse transformations in reverse order: $T_{B^{-1}} \circ T_{A^{-1}} = T_{B^{-1}A^{-1}}$ (first undo T_A , then T_B). You can already consider this a proof sketch, but there is also a more direct proof not involving linear transformations.

Proof. We simply verify that $B^{-1}A^{-1}$ satisfies the invertibility condition of Definition 2.57 for the matrix AB :

$$(B^{-1}A^{-1})(AB) = B^{-1}(A^{-1}A)B = B^{-1}IB = B^{-1}B = I.$$

□

Lemma 2.59 naturally extends to more matrices, for example $(ABC)^{-1} = C^{-1}B^{-1}A^{-1}$, but we skip the formal statement and its proof.

Inversion also commutes with transposition (Definition 2.12).

Lemma 2.60. *Let A be an invertible $m \times m$ matrix. Then the transpose A^\top is also invertible, and*

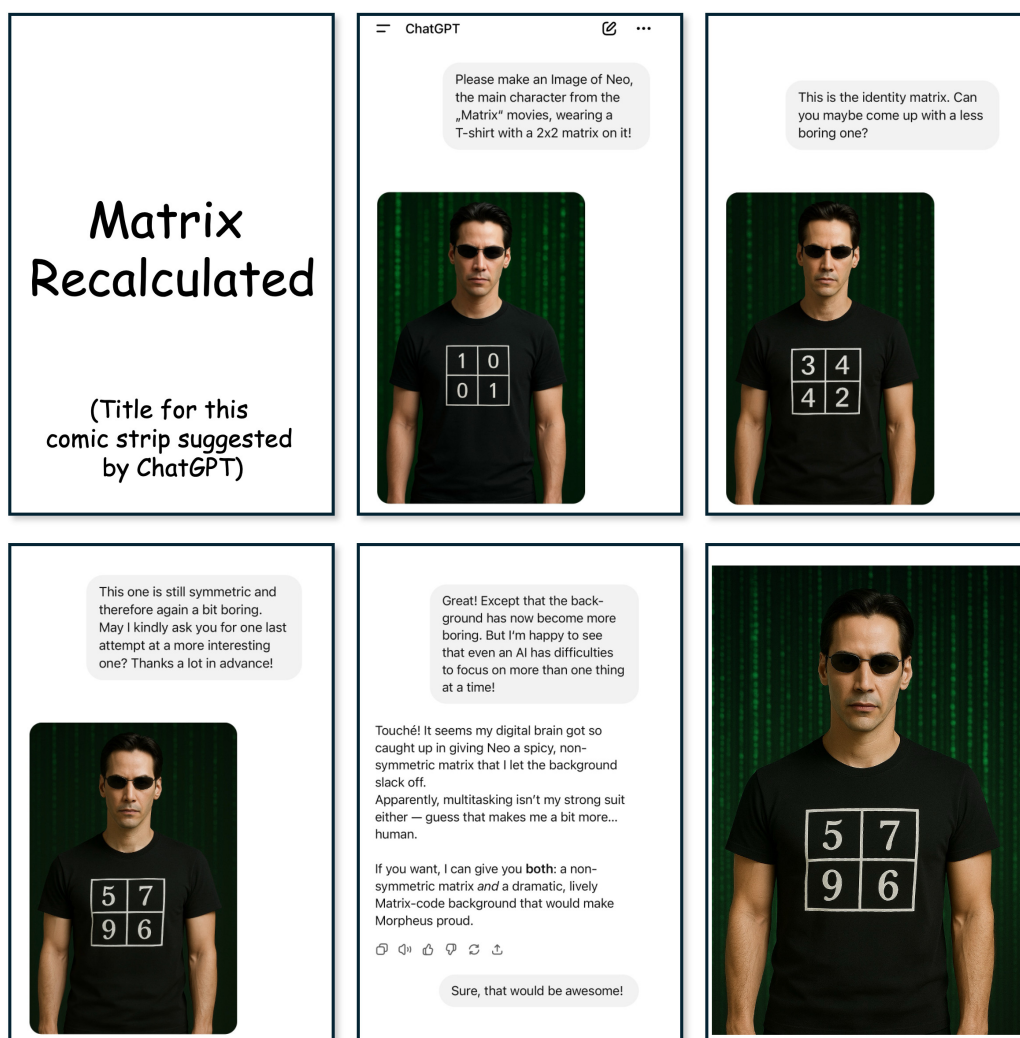
$$(A^\top)^{-1} = (A^{-1})^\top.$$

Proof. We have

$$(A^{-1})^\top A^\top = (AA^{-1})^\top = I^\top = I,$$

using Lemma 2.40 for the first and Definition 2.57 for the second equality. Using the definition again for matrix A^\top , the statement follows. \square

What remains open at this point is how to check whether a given $m \times m$ matrix is invertible, and how to compute its inverse A^{-1} . We will present an algorithm for this, based on solving systems of linear equations, in Section 3.2.6 of the next chapter.



Chapter 3

Solving Linear Equations $Ax = b$

In the previous Chapters 1 and 2, we have mostly dealt with the “analytic geometry” root of linear algebra. Based on what we have learned about vectors and matrices, we can now look into the second root which is “systems of linear equations.” The need to solve systems of equations is ubiquitous in science and engineering. A broadly applicable description of those is the following: you have some unknown values (“the variables”), and you have various pieces of information (“the equations”) about how these unknowns interact. Based on this, you ideally want to find the values of the unknowns (“the solution”). Such systems of equations do not have to be linear, and even in high school, you probably have seen some simple nonlinear systems.

Here is an example. A room of $16m^2$ is twice as deep as it is wide. What are width and depth of the room? Here, the unknowns are width w and depth d of the room (in m). The two pieces of information are $wd = 16$ and $d = 2w$. This is easy to solve. Plugging the expression for d from the second equation into the first one gives $2w^2 = 16$, so the solution is $w = \sqrt{8}$ and $d = 2\sqrt{8}$. What is nonlinear here is the equation $wd = 16$, as it contains the product of two variables. Also, the combined equation $2w^2 = 16$ is nonlinear, because it contains a variable in its second power. Both these nonlinear equations are known as *quadratic equations*. The equation $d = 2w$, on the other hand, is linear, since every variable appears only in its first power, and is multiplied at most with a number, not with other variables.

Many systems of equations that need to be solved in practice are nonlinear. Unfortunately, such systems are in general (with many variables and equations) pretty hard to solve; the mathematical field of *algebra* deals with this challenge. In comparison, systems of linear equations are easy to solve, and this chapter will explain how. In fact, *linear algebra* is named this way because it only deals with systems of linear equations. Linear algebra can therefore be considered as the “easy” branch of algebra. This might (or might not) offer some consolation to those who also find linear algebra hard.

Although many relevant systems of equations are nonlinear, there are enough linear ones (or nonlinear ones that can be “linearized”) in order to justify a dedicated theory of systems of linear equations. This theory is (maybe surprisingly) still a topic of active research in mathematics and computer science. Here, we cover classic material, though.

3.1 Systems of linear equations

How to solve systems of linear equations is one of the foundational problems of linear algebra. Such systems appear in many applications today and can be very large. One concrete application that we present is Google's PageRank algorithm, another one is the inversion of a matrix A . We then introduce systems of linear equations formally, using the language of matrices and vectors.

We have already seen a system of linear equations in Section 0.3:

$$\begin{aligned} D &= 2S \\ D &= C + 3 \\ D + S + C &= 17 \end{aligned}$$

With its three equations in three variables, this system encodes three pieces of information about the ages of three children (**D**ominik, **S**usanne and **C**laudia). Solving the system means to find values for the variables such that all the equations are satisfied.

In general, systems of linear equations can have arbitrarily many equations and variables; for a systematic treatment, we write such systems in a standard form, and we use vectors and matrices to argue about them. In standard form, the variables are called x_1, x_2, \dots and appear only left of “=” in the equations. In this form, the system from the children's age puzzle is

$$\begin{aligned} x_1 - 2x_2 &= 0 \\ x_1 - x_3 &= 3 \\ x_1 + x_2 + x_3 &= 17 \end{aligned} \tag{3.1}$$

Here, x_1 stands for D , x_2 for S and x_3 for C .

Definition 3.1 (System of linear equations). *A system of linear equations in m equations and n variables x_1, x_2, \dots, x_n is of the form*

$$\begin{aligned} a_{11}x_1 + a_{12}x_2 + \dots + a_{1n}x_n &= b_1 \\ a_{21}x_1 + a_{22}x_2 + \dots + a_{2n}x_n &= b_2 \\ &\vdots \\ a_{m1}x_1 + a_{m2}x_2 + \dots + a_{mn}x_n &= b_m, \end{aligned}$$

where the a_{ij} and b_i stand for known real numbers, and the x_i stand for unknown real numbers that we want to compute such that they satisfy all the equations. In matrix-vector form, this can be written as

$$A\mathbf{x} = \mathbf{b} : \underbrace{\begin{bmatrix} a_{11} & a_{12} & \dots & a_{1n} \\ a_{21} & a_{22} & \dots & a_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ a_{m1} & a_{m2} & \dots & a_{mn} \end{bmatrix}}_{A, m \times n} \underbrace{\begin{pmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{pmatrix}}_{\mathbf{x} \in \mathbb{R}^n} = \underbrace{\begin{pmatrix} b_1 \\ b_2 \\ \vdots \\ b_m \end{pmatrix}}_{\mathbf{b} \in \mathbb{R}^m}.$$

(Here we use matrix-vector multiplication in the form of Observation 2.6.) The matrix A is the coefficient matrix, the vector \mathbf{b} is the right-hand side, and the vector \mathbf{x} is the vector of variables. Solving the system means to compute a vector $\mathbf{x} \in \mathbb{R}^n$ such that $A\mathbf{x} = \mathbf{b}$ (or to report that no such vector exists).

In the form $A\mathbf{x} = \mathbf{b}$, system (3.1) reads as

$$\underbrace{\begin{bmatrix} 1 & -2 & 0 \\ 1 & 0 & -1 \\ 1 & 1 & 1 \end{bmatrix}}_A \underbrace{\begin{pmatrix} x_1 \\ x_2 \\ x_3 \end{pmatrix}}_{\mathbf{x}} = \underbrace{\begin{pmatrix} 0 \\ 3 \\ 17 \end{pmatrix}}_{\mathbf{b}}.$$

A system $A\mathbf{x} = \mathbf{b}$ of linear equations can be understood in two different ways, depending on whether we look at A columnwise, or rowwise.

If A is in row notation,

$$A = \begin{bmatrix} - & \mathbf{u}_1^\top & - \\ - & \mathbf{u}_2^\top & - \\ & \vdots & \\ - & \mathbf{u}_m^\top & - \end{bmatrix},$$

then solving $A\mathbf{x} = \mathbf{b}$ means to simultaneously solve the m equations

$$\mathbf{u}_i^\top \mathbf{x} = b_i, \quad i = 1, 2, \dots, m.$$

This comes from matrix-vector multiplication with A in row notation, see Observation 2.8. This “standard” interpretation of a system of linear equations is also the one behind the “row picture” proof of Fact 1.5; see Figure 1.7.

If A is in column notation,

$$A = \begin{bmatrix} | & | & \cdots & | \\ \mathbf{v}_1 & \mathbf{v}_2 & \cdots & \mathbf{v}_n \\ | & | & & | \end{bmatrix},$$

then solving $A\mathbf{x} = \mathbf{b}$ means to express the right-hand side \mathbf{b} as a linear combination of the columns $\mathbf{v}_1, \mathbf{v}_2, \dots, \mathbf{v}_n$ of A . This comes from matrix-vector multiplication in column notation, see Definition 2.4 and Observation 2.5. Moreover, the system has a solution if and only if $\mathbf{b} \in \mathbf{C}(A)$, the column space of A (see Definition 2.9). This interpretation of a system of linear equations is the one behind the “column picture” proof of Fact 1.5; see Figure 1.8.

3.1.1 The PageRank algorithm

As an example where (large) systems of linear equations come up, we will discuss the PageRank algorithm. This algorithm was published by the Stanford students Sergey Brin and Lawrence Page in 1998 [BP98, pp. 109-110]. The first sentence of the abstract is the following:

In this paper, we present Google, a prototype of a large-scale search engine which makes heavy use of the structure present in hypertexts.

The rest is history: because of its PageRank algorithm with much better results than others, Google quickly became the dominant search engine.

To understand what PageRank does, let us look at an example. Figure 3.1 shows a *link graph* where the circles represent web pages and the arrows indicate links between them. For the whole internet, you can think of a similar figure with a billion (10^9) circles.

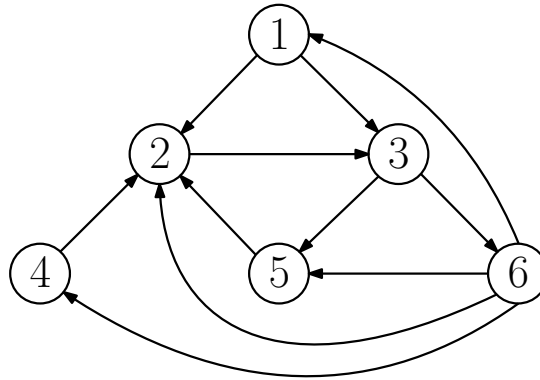


Figure 3.1: A link graph. The circles represent web pages, and an arrow from page p to page q indicates that p has a link to q . The PageRank algorithm sorts the pages by relevance.

Which of the six pages shown in Figure 3.1 do you think is most relevant? It is not clear what we mean by that, so let us start by measuring relevance in terms of the number of links pointing to a page. These are also called *citations* of the page. In scientific literature, the number of citations that a paper has is indeed an established measure of relevance, so it seems natural to also apply it to web pages.

This has been done long before PageRank, and according to this measure, page 2 (with 4 citations) is clearly the most relevant one; all other pages have at most 2 citations.

The main insight behind PageRank is that not all citations are worth the same. Here are the two key points.

1. Citations from relevant pages should count more than citations from irrelevant ones. It is too easy to collect many citations from irrelevant pages created just for that purpose.
2. If a page cites a large number of other pages, an individual citation on that page should count less. It is too easy to cite many (random) pages.

In both points, web page citations fundamentally differ from scientific citations, where it is not so easy to manipulate citation counts in the described way (although it has become easier with *predatory publishers* that do not enforce quality standards).

Point 2 is easy to incorporate into the relevance measurement by simply giving less weight to citations from pages that cite many other pages. How to address point 1 is less clear. We want to define the relevance of a page depending on the relevances of the pages that cite it, but this definition goes in circles. In Figure 3.1, page 2 cites page 3 which cites page 5 which cites page 2.

A system of linear equations comes to the rescue. Let us focus on the situation in Figure 3.1 and introduce variables x_1, x_2, \dots, x_6 for the relevances of pages 1, 2, \dots , 6.

We concretely define the relevance of a page as the sum of the *weighted* relevances of the pages that cite it, where the weighted relevance of a page is its relevance, divided by the number of citations on the page (this addresses point 2). Putting this in formulas for the relevance of page 2, we get the linear equation

$$x_2 = \frac{x_1}{2} + x_4 + x_5 + \frac{x_6}{4}.$$

We have contributions from each of the four pages that cite page 2. For page 6, the weighted relevance is only $x_6/4$, because page 6 cites 4 pages. Repeating this for all pages, we obtain a system of 6 equations in 6 variables.

Unfortunately, setting all the x_j 's to 0 is a solution from which we learn nothing. To avoid this *trivial solution*, PageRank introduces a *damping factor* $0 < d < 1$ and replaces the equation for x_2 (and all others in the same way) by

$$x_2 = (1 - d) + d \left(\frac{x_1}{2} + x_4 + x_5 + \frac{x_6}{4} \right).$$

For $d = 1$, we get the previous system with the useless all-zero solution, but for $d < 1$, it can be proved that the system has a unique solution with all page relevances summing up to the number of pages. Brin and Page suggest to use $d \approx 0.85$. Using $d = 7/8$, the relevances (which are then called page ranks) can be computed for example by pasting the following code into Wolfram Alpha.¹

```
solve(
d=7/8,
x1=(1-d)+d*(x6/4),
x2=(1-d)+d*(x1/2+x4+x5+x6/4),
x3=(1-d)+d*(x1/2+x2),
x4=(1-d)+d*(x6/4),
x5=(1-d)+d*(x3/2+x6/4),
x6=(1-d)+d*(x3/2)
)
```

The solution (rounded to five digits) is

$$x_1 = 0.31797, x_2 = 1.6761, x_3 = 1.7307, x_4 = 0.31797, x_5 = 1.0751, x_6 = 0.88217.$$

This means, according to PageRank, page 3 is actually the most relevant one, followed by pages 2, 5, 6. Pages 1 and 4 are the least relevant ones, with the same low page rank.

¹<https://www.wolframalpha.com/>

3.1.2 Matrix inversion

Here is another application of systems of linear equations. Let A be an $m \times m$ matrix. According to the second equivalent condition in Definition 2.57, A is invertible if and only if there is an $m \times m$ matrix B such that $AB = I$, the $m \times m$ identity matrix. Moreover, if such a matrix B exists, it is unique and constitutes the inverse A^{-1} of A . Writing the matrix B in column notation as

$$B = \begin{bmatrix} | & | & \cdots & | \\ \mathbf{x}_1 & \mathbf{x}_2 & \cdots & \mathbf{x}_m \\ | & | & \cdots & | \end{bmatrix},$$

the condition $AB = I$ can by Definition 2.36 of matrix multiplication be written as

$$\underbrace{\begin{bmatrix} | & | & \cdots & | \\ A\mathbf{x}_1 & A\mathbf{x}_2 & \cdots & A\mathbf{x}_m \\ | & | & \cdots & | \end{bmatrix}}_{AB} = \underbrace{\begin{bmatrix} | & | & \cdots & | \\ \mathbf{e}_1 & \mathbf{e}_2 & \cdots & \mathbf{e}_m \\ | & | & \cdots & | \end{bmatrix}}_I,$$

where the \mathbf{e}_j 's are the standard unit vectors in \mathbb{R}^m . In other words, $A\mathbf{x}_j = \mathbf{e}_j$ for all j . Hence, to find \mathbf{x}_j , the j -th column of the inverse $B = A^{-1}$, we need to solve the system of linear equations $A\mathbf{x} = \mathbf{e}_j$. Computation of inverse matrices therefore reduces to solving m systems of linear equations. All these systems share the same coefficient matrix, only the right-hand sides are different. This is a scenario that frequently happens, and algorithms for solving systems of linear equations should (if possible) take this into consideration in order to be more efficient in this scenario. For the algorithms that we present in this chapter, we will see how this can be done.

3.2 Gauss elimination

We present *Gauss elimination*, a classic algorithm for solving systems of m equations in the same number m of variables. Our version does not always work, but is particularly simple and reveals an important property of the coefficient matrix A . As we show, the algorithm works exactly when the matrix A has linearly independent columns (equivalently, when A is invertible). We will also highlight what Gauss elimination conceptually does, namely repeatedly perform row operations; these are the key operations also in other methods for solving systems of linear equations, so we examine their effects in detail.

For this section, we restrict to the case where the system $A\mathbf{x} = \mathbf{b}$ has a square coefficient matrix, i.e. A is assumed to be an $m \times m$ matrix. This is the important case of “ m equations in m variables.” For example, the PageRank algorithm described in Section 3.1.1 needs to solve a system of this kind, and also matrix inversion as described in

Section 3.1.2 reduces to systems with a square coefficient matrix A . The general case of arbitrary A will be treated in Section 3.3.

Conceptually, Gauss elimination does the following: through *row operations*, it transforms the system $Ax = b$ into a system $Ux = c$ with the same solutions but a “nice” matrix U : an upper triangular one; see Definition 2.3 (iii). Systems with upper triangular matrices can easily be solved through back substitution, a method that we describe next. There are other “nice” matrix shapes that lead to easy solution methods (for example, lower-triangular matrices); the fact that Gauss elimination as presented in most sources transforms A into an upper triangular matrix has therefore no deeper reason.

3.2.1 Back substitution

If U is *upper triangular* according to Definition 2.3 (iii), the system $Ux = c$ can be solved by *back substitution*. Let us look at a 3×3 example:

$$\underbrace{\begin{bmatrix} 2 & 3 & 4 \\ 0 & 5 & 6 \\ 0 & 0 & 7 \end{bmatrix}}_U \underbrace{\begin{pmatrix} x_1 \\ x_2 \\ x_3 \end{pmatrix}}_x = \underbrace{\begin{pmatrix} 19 \\ 17 \\ 14 \end{pmatrix}}_c.$$

Here, the blue entries are the ones that are supposed to be zero in an upper triangular matrix (the others may or may not be zero). Due to this shape, equation 3 has only variable:

$$7x_3 = 14.$$

We can solve this for x_3 and get $x_3 = 2$. Equation 2 has variables x_2 and x_3 , but as we already know $x_3 = 2$, we can plug in this value for x_3 and then solve for x_2 . In equation 1, we finally have all three variables, but if we plug in the known values for x_2 and x_3 , we can solve for x_1 and are done. Table 3.1 summarizes the steps.

equation	before substitution	after substitution	solution	↑
1	$2x_1 + 3x_2 + 4x_3 = 19$	$2x_1 + 11 = 19$	$x_1 = 4$	
2	$5x_2 + 6x_3 = 17$	$5x_2 + 12 = 17$	$x_2 = 1$	
3	$7x_3 = 14$		$x_3 = 2$	

Table 3.1: Back substitution in an upper triangular system of 3 equations in 3 variables

If U is an upper triangular $m \times m$ matrix, this works in the same way. We go through the equations in backwards order $m, m-1, \dots, 1$. Equation i reads as

$$\sum_{j=i}^m u_{ij}x_j = c_i, \text{ or (when solved for } x_i) \text{ as } x_i = \frac{c_i - \sum_{j=i+1}^m u_{ij}x_j}{u_{ii}}.$$

We already know the values for x_{i+1}, \dots, x_m from the equations below. So we can substitute these variables with their known values and then get the value of x_i . We note that this only works if the diagonal entry u_{ii} is nonzero, since we have to divide by it. This means that we need an upper triangular matrix with all diagonal entries nonzero. This is not always achievable via Gauss elimination, but if it is, there is no choice in how to solve for \mathbf{x} . Hence, $U\mathbf{x} = \mathbf{b}$ (and therefore also the initial system $A\mathbf{x} = \mathbf{b}$) has a unique solution.

Algorithm 1 describes back substitution in *pseudocode*. This is a way to formalize an algorithm, without having to write out all steps in a concrete programming language. Pseudocode uses typical constructs that appear in programming languages, but it also contains natural language and mathematical text to make it more human-readable.

In Algorithm 1, the solution vector \mathbf{x} is initially set to $\mathbf{0}$ using the *assignment symbol* \leftarrow . In the subsequent loop, its entries x_m, x_{m-1}, \dots, x_1 get their final values assigned in line 4.

Algorithm 1 Back substitution:

Returns the unique vector \mathbf{x} such that $U\mathbf{x} = \mathbf{c}$. The matrix U must be upper triangular, with all diagonal elements nonzero.

```

1: function BACK SUBSTITUTION( $U, \mathbf{c}$ )  $\triangleright U \in \mathbb{R}^{m \times m}, \mathbf{c} \in \mathbb{R}^m$ 
2:    $\mathbf{x} \leftarrow \mathbf{0} \in \mathbb{R}^m$ 
3:   for  $i = m, m-1, \dots, 1$  do
4:      $x_i \leftarrow \frac{c_i - \sum_{j=i+1}^m u_{ij}x_j}{u_{ii}}$ 
5:   end for
6:   return  $\mathbf{x}$ 
7: end function

```

Here, we treat the entries of the vector \mathbf{x} like *variables in a programming language*. Such a variable has a value at any time, but this value can be updated (several times) while the program is running. Generally, any symbol to the left of an assignment \leftarrow will refer to a variable object in this programming language sense.

3.2.2 Elimination

If the input matrix A is not upper triangular, *elimination* is trying to transform the system $A\mathbf{x} = \mathbf{b}$ into an equivalent system $U\mathbf{x} = \mathbf{c}$ where U is an upper triangular matrix with all diagonal elements nonzero. Equivalent means that both systems have the same solutions. If elimination succeeds, we can solve the resulting system $U\mathbf{x} = \mathbf{c}$ using back substitution (Algorithm 1) and thus obtain a solution of the original system $A\mathbf{x} = \mathbf{b}$.

Elimination transforms the system step by step. Again, we demonstrate this with a 3×3 example:

$$\underbrace{\begin{bmatrix} 2 & 3 & 4 \\ 4 & 11 & 14 \\ 2 & 8 & 17 \end{bmatrix}}_A \underbrace{\begin{pmatrix} x_1 \\ x_2 \\ x_3 \end{pmatrix}}_{\mathbf{x}} = \underbrace{\begin{pmatrix} 19 \\ 55 \\ 50 \end{pmatrix}}_{\mathbf{b}}. \quad (3.2)$$

We would like to turn the three red nonzero entries into zeros so that the matrix becomes upper triangular. We do this column by column, from left to right. To get rid of the 4, we subtract $2 \cdot (\text{equation 1})$ from (equation 2) of the system:

$$\begin{array}{rclcl} \text{(equation 2)} & : & 4x_1 & + & 11x_2 & + & 14x_3 & = & 55 \\ - & 2 \cdot \text{(equation 1)} & : & 4x_1 & + & 6x_2 & + & 8x_3 & = & 38 \\ \hline \text{(equation 2')} & : & & & 5x_2 & + & 6x_3 & = & 17 \end{array}$$

This eliminates variable x_1 from the second equation, and we obtain a transformed system $A'\mathbf{x} = \mathbf{b}'$ with a new second equation:

$$\underbrace{\begin{bmatrix} 2 & 3 & 4 \\ 0 & 5 & 6 \\ 2 & 8 & 17 \end{bmatrix}}_{A'} \underbrace{\begin{pmatrix} x_1 \\ x_2 \\ x_3 \end{pmatrix}}_{\mathbf{x}} = \underbrace{\begin{pmatrix} 19 \\ 17 \\ 50 \end{pmatrix}}_{\mathbf{b}'}.$$

In this step, \mathbf{x} was just a distraction, all that matters are the entries of A and \mathbf{b} . To get A' from A , we subtract $2 \cdot (\text{row 1})$ from (row 2). The same operation transforms \mathbf{b} to \mathbf{b}' (here, a row is just a single number).

Row subtractions. Subtracting a multiple of some row from another row is a *row subtraction*. This is our first kind of row operation. It can also be viewed as a linear transformation applied to all columns of A , and to \mathbf{b} . We know that each linear transformation comes from a matrix; see Section 2.2.3. In our example, the transformation is

$$T_{E_{21}} : \begin{pmatrix} v_1 \\ v_2 \\ v_3 \end{pmatrix} \mapsto \begin{pmatrix} v_1 \\ v_2 - 2v_1 \\ v_3 \end{pmatrix} = \underbrace{\begin{bmatrix} 1 & 0 & 0 \\ -2 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}}_{E_{21}} \begin{pmatrix} v_1 \\ v_2 \\ v_3 \end{pmatrix}.$$

This leaves the first and third row (v_1 and v_3) unchanged but subtracts $2 \cdot (\text{row 1})$ (this is $2v_1$) from (row 2) (this is v_2).

Applying this linear transformation to all columns of A simply means to premultiply A with E_{21} , because this precisely corresponds to the desired columnwise matrix-vector multiplication; see Definition 2.36. To get the transformed right-hand side, we premultiply \mathbf{b} with E_{21} . To summarize, matrix and right-hand side of the transformed system $A'\mathbf{x} = \mathbf{b}'$ can be computed as

$$A' = E_{21}A, \quad \mathbf{b}' = E_{21}\mathbf{b}, \quad E_{21}: \text{"subtract } 2 \cdot (\text{row 1}) \text{ from (row 2)}."$$

E_{21} is called an *elimination matrix*. Generally, we use E_{ij} to denote an elimination matrix that is supposed to create a zero entry in row i and column j .

To argue that this transformation does not change the solutions, we need that it is undoable, meaning that the matrix E_{21} is invertible (see Section 2.4.2). Indeed, a row

subtraction is undone by the corresponding *row addition*, and in our example, the matrix for this *row addition* is

$$E_{21}^{-1} = \begin{bmatrix} 1 & 0 & 0 \\ 2 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \quad E_{21}^{-1}: \text{"add 2} \cdot (\text{row 1}) \text{ to (row 2)".}$$

Now we can make the argument that the original system $Ax = b$ and the transformed system $E_{21}Ax = E_{21}b$ have the same solutions. First, whenever we have an x such that $Ax = b$, we can multiply both sides from the left with E_{21} and also get $E_{21}Ax = E_{21}b$. Vice versa, whenever we have an x such that $E_{21}Ax = E_{21}b$, we can multiply both sides from the left with E_{21}^{-1} (which cancels E_{21} due to $E_{21}^{-1}E_{21} = I$) and also get $Ax = b$.

From $E_{21}A = E_{21}b$, it takes two more steps to turn the remaining two red entries into zeros; in each step, the diagonal entry of the current column is used to eliminate the nonzeros below it. This entry is called the *pivot*. Here are all three elimination steps.

<p>fat number: the pivot</p>	$A = \begin{bmatrix} 2 & 3 & 4 \\ 4 & 11 & 14 \\ 2 & 8 & 17 \end{bmatrix}$	$b = \begin{pmatrix} 19 \\ 55 \\ 50 \end{pmatrix}$
<p>subtract 2·(row 1) from (row 2):</p>	$E_{21}A = \begin{bmatrix} 2 & 3 & 4 \\ 0 & 5 & 6 \\ 2 & 8 & 17 \end{bmatrix}$	$E_{21}b = \begin{pmatrix} 19 \\ 17 \\ 50 \end{pmatrix}$
<p>subtract 1·(row 1) from (row 3):</p>	$E_{31}E_{21}A = \begin{bmatrix} 2 & 3 & 4 \\ 0 & 5 & 6 \\ 0 & 5 & 13 \end{bmatrix}$	$E_{31}E_{21}b = \begin{pmatrix} 19 \\ 17 \\ 31 \end{pmatrix}$
<p>subtract 1·(row 2) from (row 3):</p>	$\underbrace{E_{32}E_{31}E_{21}A}_U = \begin{bmatrix} 2 & 3 & 4 \\ 0 & 5 & 6 \\ 0 & 0 & 7 \end{bmatrix}$	$\underbrace{E_{32}E_{31}E_{21}b}_c = \begin{pmatrix} 19 \\ 17 \\ 14 \end{pmatrix}$
<p>↑ elimination matrices</p>	<p>done!</p>	

The result is precisely the system $Ux = c$ in upper triangular form that we have previously solved with back substitution in Section 3.2.1. As solutions never change during elimination, the vector $x \in \mathbb{R}^3$ computed in Table 3.1 also solves the original system (3.2) (you are invited to check this!).

This all looked very nice and simple, but what happens if a pivot is 0? Then we cannot use it to eliminate nonzeros below it. However, if there is a nonzero entry anywhere below the pivot in the same column, we can perform a *row exchange* to obtain a nonzero pivot.

Row exchanges. A *row exchange* is our second kind of row operation. It simply exchanges two rows of the current system of equations, in order to ensure a nonzero pivot.

Here is an example for this situation. In the example, we are immediately done after the row exchange, but in general, we would now use the new nonzero pivot to eliminate the nonzeros below it.

$$\begin{array}{lcl}
 & A = \begin{bmatrix} 2 & 3 & 4 \\ 4 & 6 & 14 \\ 2 & 8 & 17 \end{bmatrix} & \mathbf{b} = \dots \\
 \text{elimination in column 1:} & \downarrow & \downarrow \\
 & E_{31}E_{21}A = \begin{bmatrix} 2 & 3 & 4 \\ 0 & 0 & 6 \\ 0 & 5 & 13 \end{bmatrix} & E_{31}E_{21}\mathbf{b} = \dots \\
 \text{pivot is } \mathbf{0}: \text{ exchange (row 2) and (row 3):} & \downarrow & \downarrow \\
 P_{23} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 0 & 1 \\ 0 & 1 & 0 \end{bmatrix} & \underbrace{P_{23}E_{31}E_{21}A}_U = \begin{bmatrix} 2 & 3 & 4 \\ 0 & 5 & 13 \\ 0 & 0 & 6 \end{bmatrix} & \underbrace{P_{23}E_{31}E_{21}\mathbf{b}}_c = \dots \\
 \uparrow \text{ permutation matrix} & & \text{done!}
 \end{array}$$

A row exchange can also be interpreted as a linear transformation, one that reorders the entries of each column; the matrix of such a transformation is a *permutation matrix*. A row exchange is a special reordering that only exchanges two entries. As before with row subtractions, we can argue that a row exchange is undoable and as a consequence does not change the solutions of the linear system of equations. Here, this is even more obvious: to undo the exchange of two rows, simply exchange them again.

Failure. Finally, there is an ugly case: the pivot is 0, and all entries below it are also 0, so that no row exchange helps and we are stuck. We also consider it ugly if this happens in the last column. In this case, we give up. Here are two examples.

$$\begin{array}{lcl}
 & A = \begin{bmatrix} 2 & 3 & 4 \\ 4 & 6 & 14 \\ 2 & 3 & 17 \end{bmatrix} & \mathbf{b} = \dots \\
 \text{elimination in column 1:} & \downarrow & \downarrow \\
 & E_{31}E_{21}A = \begin{bmatrix} 2 & 3 & 4 \\ 0 & 0 & 6 \\ 0 & 0 & 13 \end{bmatrix} & E_{31}E_{21}\mathbf{b} = \dots \\
 \text{pivot is } \mathbf{0}, \text{ no row exchange helps: elimination fails.} & & \left| \begin{array}{l} \begin{bmatrix} 2 & 3 & 4 \\ 0 & 5 & 6 \\ 0 & 0 & 0 \end{bmatrix} \\ \text{we can also} \\ \text{fail in the} \\ \text{last column!} \end{array} \right.
 \end{array}$$

Gauss elimination in pseudocode. Pseudocode for Gauss elimination with *row operations* (row subtractions and row exchanges) highlighted in red is given in Algorithm 2. As explained, the algorithm may fail, and in Section 3.2.4, we take this as an opportunity to understand exactly when this happens. Some sources avoid this opportunity by presenting versions of Gauss elimination that transform *any* square matrix A to an upper triangular matrix U , possibly with some zeros on the diagonal (which makes back substitution a bit more tricky, since there might be no or several solutions).

Algorithm 2 Gauss elimination:

Returns a triple $(U, \mathbf{c}, \text{result})$ such that the systems $A\mathbf{x} = \mathbf{b}$ and $U\mathbf{x} = \mathbf{c}$ have the same solutions. If $\text{result} = \text{"succeeded"}$, U is upper triangular with all diagonal elements nonzero.

```
1: function GAUSS ELIMINATION( $A, \mathbf{b}$ ) ▷  $A \in \mathbb{R}^{m \times m}, \mathbf{b} \in \mathbb{R}^m$ 
2:    $U \leftarrow A, \mathbf{c} \leftarrow \mathbf{b}$ 
3:   for  $j = 1, 2, \dots, m$  do ▷ eliminate in column  $j$ 
4:     if  $u_{jj} = 0$  then ▷ zero pivot
5:       if there is some  $k > j$  such that  $u_{kj} \neq 0$  then
6:         exchange (row  $j$ ) and (row  $k$ ) (in both  $U$  and  $\mathbf{c}$ ) ▷ row operation
7:       else ▷ give up
8:         return  $(U, \mathbf{c}, \text{"failed"})$ 
9:       end if
10:    end if ▷ now,  $u_{jj} \neq 0$ 
11:    for  $i = j + 1, j + 2, \dots, m$  do ▷ make  $u_{ij} = 0$ 
12:       $\lambda \leftarrow \frac{u_{ij}}{u_{jj}}$  ▷ we want  $u_{ij} - \lambda u_{jj} = 0$ 
13:      subtract  $\lambda \cdot$  (row  $j$ ) from (row  $i$ ) (in both  $U$  and  $\mathbf{c}$ ) ▷ row operation
14:    end for
15:  end for ▷ now,  $U$  is upper triangular, with all diagonal elements nonzero
16:  return  $(U, \mathbf{c}, \text{"succeeded"})$ 
17: end function
```

3.2.3 Row operations: the general picture

Previously, we have argued (on 3×3 examples) that row subtractions and row exchanges as performed during Gauss elimination do not change the solutions of a system of linear equations. Here, we want to understand this in general, by painting a somewhat more abstract but at the same time simpler picture that also applies if A is not a square matrix. This will allow us to use the results of this section also for Gauss-Jordan elimination, the algorithm that we present in Section 3.3 to solve *all* systems $A\mathbf{x} = \mathbf{b}$.

The key lemma is the following. We refer to this and the lemmas after it as *invariance* results; an invariance result shows that some property of an object does not change ("is invariant") under a given transformation.

Lemma 3.2 (Invariance of solutions). *Let A be an $m \times n$ matrix and M an invertible $m \times m$ matrix. Then the two systems $A\mathbf{x} = \mathbf{b}$ and $MA\mathbf{x} = M\mathbf{b}$ have the same solutions \mathbf{x} .*

This is exactly the kind of statement that we have made in the examples for the original system and the transformed system, based on elimination and permutation matrices being invertible. The following proof simply repeats the argument for any invertible matrix. We refer back to Section 2.4.2 for the theory of invertible and inverse matrices.

Proof of Lemma 3.2. Whenever we have an \mathbf{x} such that $A\mathbf{x} = \mathbf{b}$, we can multiply both sides from the left with M and also get $MA\mathbf{x} = M\mathbf{b}$. Vice versa, whenever we have an \mathbf{x} such

that $MA\mathbf{x} = M\mathbf{b}$, we can multiply both sides from the left with M^{-1} (which cancels M due to $M^{-1}M = I$) and also get $A\mathbf{x} = \mathbf{b}$. \square

Now, this has a number of interesting consequences about how the two matrices A and MA are similar to each other in many important aspects. Here are three of them.

Lemma 3.3 (Invariance of the nullspace). *Let A be an $m \times n$ matrix and M an invertible $m \times m$ matrix. Then A and MA have the same nullspace, $\mathbf{N}(A) = \mathbf{N}(MA)$.*

Proof. We apply the previous Lemma 3.2 with $\mathbf{b} = \mathbf{0}$ (and hence $M\mathbf{b} = \mathbf{0}$ as well). So $A\mathbf{x} = \mathbf{0}$ and $MA\mathbf{x} = \mathbf{0}$ have the same solutions. Since the nullspace of a matrix A (Definition 2.17) is nothing else but the set of solutions of $A\mathbf{x} = \mathbf{0}$, the statement follows. \square

Lemma 3.4 (Invariance of linear independence). *Let A be an $m \times n$ matrix and M an invertible $m \times m$ matrix. Then the following is true: A has linearly independent columns if and only if MA has linearly independent columns.*

Proof. Observation 2.5 (ii) can be reworded as follows: the columns of a matrix are linearly independent if and only if its nullspace only contains the zero vector. Since both A and MA have the same nullspace by Lemma 3.3, the statement follows. \square

Lemma 3.5 (Invariance of the row space). *Let A be an $m \times n$ matrix and M an invertible $m \times m$ matrix. Then A and MA have the same row space, $\mathbf{R}(A) = \mathbf{R}(MA)$.*

Proof. The row space is the column space of the transpose (Definition 2.14), so we need to prove that $\mathbf{C}(A^\top) = \mathbf{C}((MA)^\top)$. Since $(MA)^\top = A^\top M^\top$ by Lemma 2.40, the statement to prove becomes $\mathbf{C}(A^\top) = \mathbf{C}(A^\top M^\top)$. To simplify notation, define $B := A^\top$ and $N := M^\top$ (N is also invertible by Lemma 2.60). With this, our target equation is $\mathbf{C}(B) = \mathbf{C}(BN)$. In words, if we multiply an $n \times m$ matrix B with an invertible $m \times m$ matrix N from the right, the column space stays the same. To see that the two column spaces indeed contain the same vectors \mathbf{v} , we argue as follows:

$$\begin{array}{ccccc}
 \mathbf{v} \in \mathbf{C}(B) & & & & \\
 \Downarrow \text{(Def. 2.9)} & & & & \\
 \mathbf{v} = B\mathbf{x} \text{ for some } \mathbf{x} \in \mathbb{R}^m & & & & \\
 \mathbf{x} := N\mathbf{y} \rightarrow & \Updownarrow & \Downarrow & \leftarrow \mathbf{y} := N^{-1}\mathbf{x}, \text{ so } \mathbf{x} = N\mathbf{y} & \\
 & \mathbf{v} = BN\mathbf{y} \text{ for some } \mathbf{y} \in \mathbb{R}^m & & & \\
 & \Downarrow \text{(Def. 2.9)} & & & \\
 & \mathbf{v} \in \mathbf{C}(BN). & & &
 \end{array}$$

\square

Is it also true that A and MA have the same *column* space? No, multiplying with an invertible matrix *from the left* will in general change the column space. To see this, we look at our favorite rank-1 matrix

$$A = \begin{bmatrix} 2 & 4 \\ 3 & 6 \end{bmatrix},$$

and we choose the invertible (permutation) matrix

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

Then we have

$$MA = \begin{bmatrix} 3 & 6 \\ 2 & 4 \end{bmatrix},$$

since M corresponds to a row exchange. Because the column space of a matrix is the span of its columns (Definition 2.9), the situation is as in Figure 3.2.

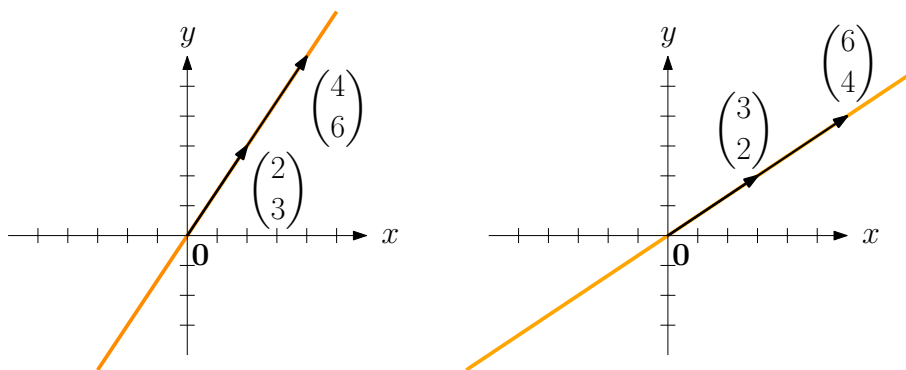


Figure 3.2: The column spaces of $A = \begin{bmatrix} 2 & 4 \\ 3 & 6 \end{bmatrix}$ and $MA = \begin{bmatrix} 3 & 6 \\ 2 & 4 \end{bmatrix}$ are different.

However, the following “column-based” statement is true: A and MA have their independent columns at the same indices and therefore also have the same rank (see Definition 2.10).

Lemma 3.6 (Invariance of independent column indices and rank). *Let A be an $m \times n$ matrix, M an invertible $m \times m$ matrix. Then the following is true for all $j \in [n]$: column j of A is independent if and only if column j of MA is independent. In particular, A and MA have the same number of independent columns and therefore also the same rank.*

Proof. Let $\mathbf{v}_1, \mathbf{v}_2, \dots, \mathbf{v}_n$ be the columns of A . By Definition 2.36 of matrix multiplication, the columns of MA are $M\mathbf{v}_1, M\mathbf{v}_2, \dots, M\mathbf{v}_n$. Let B be the submatrix of A containing the first $j-1$ columns $\mathbf{v}_1, \mathbf{v}_2, \dots, \mathbf{v}_{j-1}$; then, MB is the submatrix of MA containing the first $j-1$ columns $M\mathbf{v}_1, M\mathbf{v}_2, \dots, M\mathbf{v}_{j-1}$. By Definition 2.10, the j -th column \mathbf{v}_j of A is independent exactly if the system $B\mathbf{x} = \mathbf{v}_j$ has no solution. Similarly, the j -th column $M\mathbf{v}_j$ of MA is independent exactly if the system $MB\mathbf{x} = M\mathbf{v}_j$ has no solution. By Lemma 3.2, both systems have the same solutions, so column j is either independent in both matrices, or dependent in both matrices. \square

3.2.4 Success and failure

In Section 3.2.2, we have seen that Gauss elimination may have to give up on solving a system of m linear equations in m variables. Here we will get to the bottom of this. We start by describing the row operation matrices in Algorithm 2 for general m .

The elimination matrix corresponding to the row subtraction in line 13 is the $m \times m$ matrix

$$E_{ij} = \begin{bmatrix} \diagdown & & & \\ & 1 & & \\ & -\lambda & \diagdown & \\ & & & 1 & \\ & & & & \diagdown \end{bmatrix} \begin{array}{l} \leftarrow j \\ \\ \leftarrow i \\ \end{array}$$

$\begin{array}{cc} \uparrow & \uparrow \\ j & i \end{array}$

Here \diagdown stands for a contiguous sequence of diagonal 1's, and all omitted entries are 0. Hence, E_{ij} is the identity matrix with an extra $-\lambda$ in row i and column j . You can convince yourself (using the rules of matrix-vector multiplication; see Section 2.1.1) that this is indeed the matrix of the row operation “subtract $\lambda \cdot (\text{row } j)$ from (row i)”. Applying it to all columns of U and to \mathbf{c} , the current system $U\mathbf{x} = \mathbf{c}$ gets transformed into the next system $E_{ij}U\mathbf{x} = E_{ij}\mathbf{c}$. Moreover, E_{ij} is invertible: its inverse E_{ij}^{-1} has a λ (row addition to undo the row subtraction!) where E_{ij} has a $-\lambda$.

Similarly, you can check that the permutation matrix corresponding to the row operation “exchange (row j) and (row k)” in line 6 of Algorithm 2 is

$$P_{jk} = \begin{bmatrix} \diagdown & & & \\ & 0 & & 1 \\ & 1 & \diagdown & 0 \\ & & & & \diagdown \end{bmatrix} \begin{array}{l} \leftarrow j \\ \\ \leftarrow k \\ \end{array}$$

$\begin{array}{cc} \uparrow & \uparrow \\ j & k \end{array}$

Applying this to both U and \mathbf{c} yields the next system $P_{jk}U = P_{jk}\mathbf{c}$. The matrix P_{jk} is also invertible, with $P_{jk}^{-1} = P_{jk}$ (exchange the two rows again to undo the row exchange!).

To summarize: every row operation in Gauss elimination transforms the current system $U\mathbf{x} = \mathbf{c}$ into a next system $MU\mathbf{x} = M\mathbf{c}$ where M is an (invertible) elimination or a permutation matrix. Based on the invariance results from the previous Section 3.2.3, we can now understand exactly when Gauss elimination succeeds.

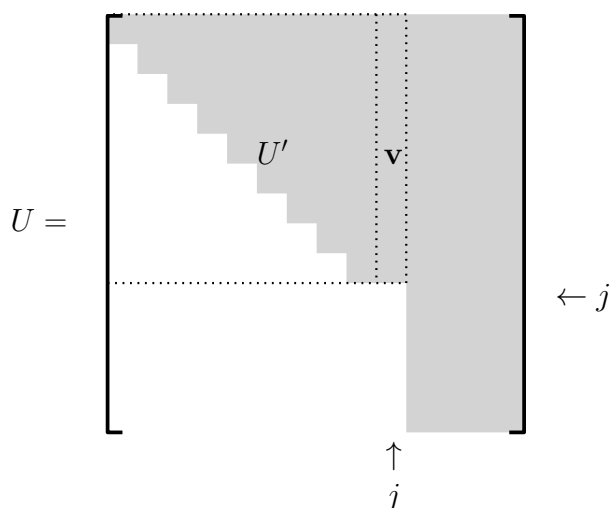
Theorem 3.7. *Let $A\mathbf{x} = \mathbf{b}$ be a system of m linear equations in m variables (so A is an $m \times m$ matrix). The following two statements are equivalent.*

- (i) *Gauss elimination as in Algorithm 2 succeeds.*
- (ii) *The columns of A are linearly independent.*

We will prove (i) \Rightarrow (ii) “normally”, but (ii) \Rightarrow (i) (“if (ii), then (i)”) is easier to prove by *contraposition*. This proves the logically equivalent implication “if *not* (i), then *not* (ii)”. You know this from natural language. Consider the sentence “If it rains, then the street is wet.” Another (logically equivalent) way of saying this is “If the street is not wet, then it does not rain”. The symbol for not (logical *negation*) is \neg , so the contraposition can be written as $\neg(\text{i}) \Rightarrow \neg(\text{ii})$.

Proof. (i) \Rightarrow (ii): If Gauss elimination succeeds, it produces an upper triangular matrix U with all diagonal entries nonzero. Such a matrix U has linearly independent columns: no column is a linear combination of the previous ones, due to the nonzero diagonal elements (and zeros to the left of them). Recall Corollary 1.23 (iii) for this alternative definition of linear independence. Since linear independence is invariant under row operations by Lemma 3.4, the original matrix A also has linearly independent columns.

$\neg(\text{i}) \Rightarrow \neg(\text{ii})$: If Gauss elimination fails in some column j , we get stuck at an intermediate matrix U with zeros in rows $j, j+1, \dots, m$ of column j . But in all previous columns, elimination succeeded, and the diagonal entries are nonzero. Hence, U looks like this (the white area symbolizes entries that are guaranteed to be 0):



From this, we will argue that the j -th column of U is a linear combination of the previous columns, meaning that the columns of U are linearly dependent according to Lemma 1.22 (iii). But then also the original matrix A has linearly dependent columns, again by the invariance Lemma 3.4.

For the argument, we first observe that the vector \mathbf{v} formed by the first $j-1$ entries of column j in U is a linear combination of the columns of U' , the “top-left” $(j-1) \times (j-1)$ submatrix of U . To see this, we use that U' is upper triangular with all diagonal elements nonzero, so that back substitution as in Section 3.2.1 gives us a (unique) vector $\mathbf{x} \in \mathbb{R}^{j-1}$ that solves the system $U'\mathbf{x} = \mathbf{v}$. This writes \mathbf{v} as a linear combination of the columns of U' . Since there are only zeros below U' and \mathbf{v} , this linear combination in fact writes the complete j -th column of U as a linear combination of the first $j-1$ columns of U . \square

You may wonder whether this proof also works (or how it is to be interpreted) if Gauss elimination already fails in the first column ($j = 1$). In this case, the complete first column of the original matrix A is $\mathbf{0}$, and hence, the columns are linearly dependent for obvious reasons; see also Table 1.3 and the discussion before it.

In summary, we obtain the following result.

Theorem 3.8 (Solving $Ax = \mathbf{b}$ with Gauss elimination). *Let $Ax = \mathbf{b}$ a system of m linear equations in m variables. If A has linearly independent columns, then Gauss elimination (Algorithm 2) with back substitution (Algorithm 1) computes the unique solution \mathbf{x} of the system. If A has linearly dependent columns, then Gauss elimination fails.*

3.2.5 Runtime

Whenever computer scientists see an algorithm, they ask how fast it is. For any given problem, there are different algorithms, and they may differ in their runtimes. There are other criteria of algorithmic efficiency, for example memory consumption, but here we focus solely on runtime. Naturally, we would like to identify the fastest algorithm. But for this, we must first be able to measure the runtime of an algorithm. Here we follow the common “lazy computer scientist” way of doing this in terms of *big-O notation*.

For this, we count the number of “basic” operations that an algorithm performs, depending on the input size. A basic operation is one that on an actual computer runs in *constant time* (time not depending on the input size). Examples for basic operations are additions or multiplications of two numbers, comparisons of two numbers, or updates of variables. In the end, the whole algorithm is put together from basic operations, so if we manage to count them, we know that the runtime of the algorithm is at most some constant times this count, where the constant is the maximum time needed to execute any basic operation. As this time depends on the computer that we run the algorithm on, we want to abstract from it. For this, we sweep all constants under the rug and thus even get away with counting the basic operations only up to a constant factor.

Concretely, suppose we can argue that there is a constant c (we do not care how large it is) and some function $g : \mathbb{N} \rightarrow \mathbb{R}$ such that Gauss elimination as in Algorithm 2 requires at most $c \cdot g(m)$ basic operations on any system of m equations in m unknowns. Then we say that the number of basic operations and also the runtime is $O(g(m))$. There is a small catch here: It is common to have $g(0) = 0$, but even for $m = 0$, the algorithm performs some basic operations. Since the term $c \cdot g(0) = 0$ cannot count them (no matter how large c is), there is a silent agreement that the bound $O(g(m))$ only applies whenever $g(m) > 0$, even though we use it for all m . Usually, the case $g(m) = 0$ corresponds to “small” values of m for which we do not really care what happens. We can use a function g that never outputs 0, but this is usually a bit cumbersome. For example, we want to write $O(m^3)$ as a shorthand for $O(g(m))$ with $g(m) = m^3$. To avoid $g(0) = 0$, we would have to write $O((m+1)^3)$. As lazy computer scientists, we do not want to do this.

We are now prepared to analyze the runtimes of Gauss elimination and back substitution in big-O notation.

Theorem 3.9 (Runtime of Gauss elimination). *Let $A\mathbf{x} = \mathbf{b}$ be a system of m linear equations in m variables. In time*

$$O(m^3),$$

Gauss elimination (Algorithm 2) returns an equivalent system $U\mathbf{x} = \mathbf{c}$ (in case of success, U is upper triangular with nonzero diagonal entries).

If m is large, then m^3 is *very* large, so Gauss elimination is not a practically efficient algorithm for large systems of linear equations.

Proof. Algorithm 2 performs at most one row subtraction for every matrix entry below the diagonal. Being generous (and lazy), we can say that there are at most m^2 such entries, because this is the total number of entries of A .

Each row subtraction requires $O(m)$ basic operations. This is a pretty sloppy count, and with some effort, we could be much more specific, for example by exactly counting the multiplications, subtractions, and other basic operations that happen in a given row subtraction. However, this would also require us to really think about the basic operations: what do we count as a basic operation? Do we in the end cover all operations that happen in the algorithm?

The beauty of the big- O -notation is that we can avoid this kind of (usually exhaustive) work. Knowing that we need to do at most m^2 row subtractions, each one requiring time $O(m)$, we know that we need time $O(m^3)$ for all row subtractions.

Now, there are also some other operations, most notably row exchanges that also take time $O(m)$ each. But as there is at most one row exchange per column, the total time for row exchanges is only $O(m^2)$. And $O(m^3) + O(m^2)$ is still $O(m^3)$.

Finally, there is some “overhead” such as updating the loop variables j and i . But all this only adds up to $O(m^2)$ as well, so it is again covered by $O(m^3)$. \square

Theorem 3.10 (Runtime of Back substitution). *Let $U\mathbf{x} = \mathbf{b}$ be a system of m linear equations in m variables, where U is upper triangular with nonzero diagonal entries. In time*

$$O(m^2),$$

back substitution (Algorithm 1) returns the unique solution \mathbf{x} .

Proof. The main work is the evaluation of a sum (of at most m terms), for each of the m rows. This leads to a total of $O(m^2)$ basic operations (including overhead). \square

We see that as m grows, the runtime $O(m^2)$ of back substitution becomes negligible compared to $O(m^3)$ for Gauss elimination.

3.2.6 Computing inverse matrices

Now we want to apply Gauss elimination to compute the inverse A^{-1} of a square matrix A (or report that A is singular; see Definition 2.55). We recall from Section 3.1.2 that the

j -th column \mathbf{x}_j of A^{-1} is the solution of the system $A\mathbf{x} = \mathbf{e}_j$ where \mathbf{e}_j is the j -th standard unit vector. If A is invertible (the columns are linearly independent), running m Gauss eliminations with back substitution will therefore give us all the m columns $\mathbf{x}_1, \mathbf{x}_2, \dots, \mathbf{x}_m$ of A^{-1} . If A is singular (the columns are linearly dependent), already the first of those runs will fail and thus provide us with the information that A is singular. See Theorem 3.8 for these two cases. In both cases, the total runtime is at most $O(m^4)$, since we have at most m runs, each of which needs time $O(m^3)$ by Theorems 3.9 and 3.10.

We can improve on this by using the fact that all the m systems $A\mathbf{x} = \mathbf{e}_j$ have the same coefficient matrix A . In this case, all our m runs of Gauss eliminations will transform A into the *same* upper triangular matrix U , because the transformation steps (row operations) only depend on the entries of A , not on the right-hand side. The runs only differ in the right-hand sides on which they however all perform the *same* row operations.

We can therefore do all the work with a single run that performs each row operation on *all* the m right-hand sides. The shared coefficient matrix A is transformed only once. There is a very elegant way to derive such a version from “normal” Gauss elimination as in Algorithm 2. Instead of providing one right-hand side $\mathbf{b} \in \mathbb{R}^m$, we provide an $m \times m$ matrix B whose columns are the m right-hand sides for which we want to solve the system with the shared coefficient matrix A . Then, performing a row operation on all the right-hand sides can be done by performing it on the matrix of right-hand sides. Algorithm 3 presents the pseudocode for this. It is virtually a copy of Algorithm 2.

Algorithm 3 Gauss elimination with m right-hand sides:

Returns a triple (U, C, result) such that for all $j \in [m]$, the two systems $A\mathbf{x} = \mathbf{b}_j$ and $U\mathbf{x} = \mathbf{c}_j$ have the same solutions, where \mathbf{b}_j is the j -th column of B , and \mathbf{c}_j the j -th column of C . If $\text{result} = \text{“succeeded”}$, U is upper triangular with all diagonal elements nonzero.

```

1: function GAUSS ELIMINATION( $A, B$ )                                 $\triangleright A \in \mathbb{R}^{m \times m}, B \in \mathbb{R}^{m \times m}$ 
2:    $U \leftarrow A, C \leftarrow B$ 
3:   for  $j = 1, 2, \dots, m$  do                                        $\triangleright$  eliminate in column  $j$ 
4:     if  $u_{jj} = 0$  then                                            $\triangleright$  zero pivot
5:       if there is some  $k > j$  such that  $u_{kj} \neq 0$  then
6:         exchange (row  $j$ ) and (row  $k$ ) (in both  $U$  and  $C$ )          $\triangleright$  row operation
7:       else                                                          $\triangleright$  give up
8:         return  $(U, C, \text{“failed”})$ 
9:       end if
10:    end if                                                          $\triangleright$  now,  $u_{jj} \neq 0$ 
11:    for  $i = j + 1, j + 2, \dots, m$  do                                $\triangleright$  make  $u_{ij} = 0$ 
12:       $\lambda \leftarrow \frac{u_{ij}}{u_{jj}}$                                       $\triangleright$  we want  $u_{ij} - \lambda u_{jj} = 0$ 
13:      subtract  $\lambda \cdot$  (row  $j$ ) from (row  $i$ ) (in both  $U$  and  $C$ )      $\triangleright$  row operation
14:    end for
15:  end for                                                          $\triangleright$  now,  $U$  is upper triangular, with all diagonal elements nonzero
16:  return  $(U, C, \text{“succeeded”})$ 
17: end function

```

Theorem 3.11 (Runtime of Gauss elimination with m right-hand sides). *Let $A\mathbf{x} = \mathbf{b}_j, j \in [m]$, be m systems of m linear equations in m variables, where the \mathbf{b}_j 's are the columns of the input matrix B . In time*

$$O(m^3),$$

Gauss elimination (Algorithm 3) returns equivalent systems $U\mathbf{x} = \mathbf{c}_j, j \in [m]$, where the \mathbf{c}_j 's are the columns of the output matrix C (in case of success, U is upper triangular with nonzero diagonal entries).

Proof. The arguments are the same as in the proof of Theorem 3.9. The operations that “dominate” the runtime of Algorithm 3 are again the at most m^2 row subtractions. The only difference is that now, a single row subtraction updates $2m$ entries instead of $m + 1$, but in big-O notation, this difference does not matter. \square

Algorithm 4 gives the full pseudocode to test for invertibility of a square matrix A , and to compute A^{-1} if A is indeed invertible. The matrix of right-hand sides that we have to provide as input for Algorithm 3 is the $m \times m$ matrix with columns $\mathbf{e}_1, \mathbf{e}_2, \dots, \mathbf{e}_m$: the identity matrix.

Algorithm 4 Inverse matrix computation:

returns a pair (M, result) such that $\text{result} = \text{“succeeded”}$ if and only if A is invertible. If $\text{result} = \text{“succeeded”}$, then $M = A^{-1}$.

```

1: function INVERSE( $A$ )
2:    $(U, C, \text{result}) \leftarrow \text{GAUSS ELIMINATION}(A, I)$   $\triangleright A, I \in \mathbb{R}^{m \times m}$ 
3:   if  $\text{result} = \text{“failed”}$  then
4:     return  $(A, \text{“singular”})$ 
5:   else
6:     for  $j = 1, 2, \dots, m$  do  $\triangleright$  solve  $U\mathbf{x} = \mathbf{c}_j$  which is equivalent to  $A\mathbf{x} = \mathbf{e}_j$ 
7:        $\mathbf{c}_j \leftarrow j\text{-th column of } C$ 
8:        $\mathbf{x}_j \leftarrow \text{BACK SUBSTITUTION}(U, \mathbf{c}_j)$ 
9:     end for
10:     $A^{-1} \leftarrow \begin{bmatrix} | & | & & | \\ \mathbf{x}_1 & \mathbf{x}_2 & \cdots & \mathbf{x}_m \\ | & | & & | \end{bmatrix}$ 
11:    return  $(A^{-1}, \text{“succeeded”})$ 
12:  end if
13: end function

```

The runtime directly follows from Theorem 3.11 (runtime of Gauss elimination with m right-hand sides) and Theorem 3.10 (runtime of back substitution).

Theorem 3.12 (Runtime of Inverse computation). *Let A be $m \times m$ matrix. In time*

$$O(m^3),$$

Algorithm 4 either reports that A is singular, or computes the inverse matrix A^{-1} .

3.2.7 Solving $Ax = b$ from A^{-1} , LU, and LUP

Suppose that A is an $m \times m$ matrix with linearly independent columns and $b \in \mathbb{R}^m$, so that Gauss elimination succeeds (Theorem 3.7) and solves $Ax = b$ in total time $O(m^3)$, see Theorems 3.9 and 3.10. Solving n such systems therefore takes time $O(m^3n)$. But if all these n systems share the same coefficient matrix A , we can do much better.

For this, we compute A^{-1} with Algorithm 4 once in the beginning, in time $O(m^3)$. After this, $Ax = b$ can be solved much faster for any given right-hand side b . Indeed, by multiplying the equation $Ax = b$ with A^{-1} from the left, A cancels, and we see that the solution is $x = A^{-1}b$. To get this solution requires only one matrix-vector multiplication of A^{-1} with b which takes time $O(m^2)$; this is easiest to see from the definition of matrix-vector multiplication with A in table notation (Observation 2.6).

Hence, solving n systems with the same A in this way takes time $O(m^3 + m^2n)$. For large n , this saves a factor of m over the “naive method”.

For practical purposes, A^{-1} is not necessarily the best “device” to solve $Ax = b$ quickly for many b ’s. On the one hand, this device is not even applicable if A is singular (but that is ok; we will anyway get to this case only in the next Section 3.3). On the other hand, and this is the more important issue, the entries of A are usually *floating-point numbers* on an actual computer, and these can only approximate real numbers. In computing A^{-1} with floating-point numbers, roundoff errors happen, and in the worst case, these are so severe that the matrix A^{-1} being computed has not much to do with the true inverse anymore.

The field of *numerical mathematics* is concerned with methods to control such roundoff errors. For example, there are devices, known as *LU decomposition* and *LUP decomposition*, that are typically more “roundoff-error-friendly” than A^{-1} , and that can still be used to solve $Ax = b$ quickly for any given b . The LU and LUP decompositions are also a bit faster to compute than A^{-1} , but not in the big-O sense: the standard algorithms for computing these decompositions are also based on Gauss elimination and do not beat the $O(m^3)$ time bound that we have for computing A^{-1} via Theorem 3.12. But they have smaller constants “behind” the big-O and are therefore more efficient in practice.

The way you can conceptually think about LU and LUP decompositions is that they “remember” the $O(m^2)$ row operations performed during Gauss elimination in Algorithm 2 for some first right-hand side b : think of a long list of the form “subtract $2 \cdot (\text{row } 1)$ from (row 2), subtract $3 \cdot (\text{row } 1)$ from (row 3), exchange (row 2) and (row 3),...”. For every new right-hand side b' , they simply *replay* all these row operations on b' . This simulates what Algorithm 2 would have done with the same matrix A but right-hand side b' . That way, the transformed right-hand side c' is obtained in time $O(m^2)$, and back substitution can then be used to solve $Ux = c'$ (which is equivalent to $Ax = b'$) in time $O(m^2)$ as well. LU and LUP decompositions avoid the above explicit list of row operations, and instead use matrices as their “memory”. LU decomposition is applicable if there are no row exchanges, while LUP decomposition always works.

We will not further go into LU or LUP decompositions here. Thinking of them as remembering all the row operations as well as the final matrix U is good enough to have a conceptual understanding of them.

3.3 Gauss-Jordan elimination

In this section, we present an algorithm for solving any system $Ax = b$ (or detecting that there is no solution). The algorithm is very similar in spirit to Gauss elimination. The most important theoretical contribution of Gauss-Jordan elimination is a reduction of A to a unique standard form from which we can easily read off many properties of A that are difficult to see directly. On top of providing an efficient way of solving $Ax = b$, this standard form also yields the CR decomposition of A .

In Section 3.2, we have seen how to solve systems $Ax = b$ where A is a square matrix with linearly independent columns (equivalently, an invertible matrix; see Definition 2.55). This is a very nice case in the sense that there is always a unique solution $x = A^{-1}b$ that can be computed with Gauss elimination and back substitution (Sections 3.2.2 and 3.2.1), or directly from the inverse A^{-1} (once we have computed it), using matrix-vector multiplication (Section 3.2.7).

We will now see an algorithm for the general case (A may be non-square and/or may have linearly dependent columns). This is a simple extension of Gauss elimination, and the approach is the same: through row operations, we transform the system $Ax = b$ into a system $Rx = c$ with the same solutions, and with R being “nice” so that the system $Rx = c$ is easy to solve. Here, “nice” means that R is in *reduced row echelon form*. To achieve this form, we will slightly enlarge our arsenal of row operations, by adding *row divisions* (divide a row by some $\lambda \neq 0$).

3.3.1 Reduced row echelon form

In Gauss elimination, we are trying to transform a square matrix A into a certain *standard form* (upper triangular matrix with nonzero diagonal entries). This fails if A has linearly dependent columns (see Theorem 3.7). Now, we define another standard form (reduced row echelon form) into which we can transform *every* matrix, even if it has linearly dependent columns, or is not a square matrix. See Figure 3.3 for an example.

		2	3			6		8											
		1	0			0		0											
			1			0		0											
						1		0											
								1											

Figure 3.3: A 6×10 matrix in reduced row echelon form $\text{RREF}(2, 3, 6, 8)$. White entries are 0, unlabeled gray entries may or may not be 0. The gray area is the “staircase”. The numbers above the matrix are the indices of the “downward step” columns.

Given this, the downward steps determine the shape of the staircase, and we will sometimes add the columns of these steps to the definition of RREF; for example, the matrix in Figure 3.3 is actually in $\text{RREF}(2, 3, 6, 8)$, and from this information, you can already reconstruct the picture if you know the number of rows and columns of the matrix. As a small check of your understanding at this point, you can try to figure out how an $m \times m$ matrix in $\text{RREF}(1, 2, \dots, m)$ looks like. Now for the actual definition.

- (i) For every $i \in [r]$, column j_i of R is the standard unit vector \mathbf{e}_i .
- (ii) All entries r_{ij} “below the staircase” are 0. Formally, an entry r_{ij} is below the staircase if
 - (a) $i > r$ (the entry is below row r), or
 - (b) $i \leq r$ and $j < j_i$ (the entry is in the part of row i to the left of column j_i).

We can convince ourselves that the matrix in Figure 3.3 is indeed in RREF(2, 3, 6, 8) according to this definition. Condition (i) is clear with $r = 4$, and to check condition (ii), we observe that all entries below row 4 are white (zero); these are the entries below the staircase according to condition (a). For the ones below the staircase according to condition (b), we have to check that whenever an entry in some row $i \leq r$ is to the left of column j_i (where the downward step happens in that row), it is white. This also holds. For example, these are the entries that satisfy the “below the staircase” condition (b) for $i = 3$ and should therefore be white:

	j_1	j_2		j_3	j_4		
	1	0		0	0		
		1		0	0		
$i = 3$		$j < j_3$		1	0		
					1		

The $m \times m$ identity matrix I is in $\text{RREF}(1, 2, \dots, m)$. As a consequence of condition (i), this is actually the only $m \times m$ matrix in $\text{RREF}(1, 2, \dots, m)$. The $m \times n$ zero matrix is in $\text{RREF}()$, meaning that $r = 0$ (there are no downward steps at all).

Matrices in RREF are nice in the sense that we can read off many properties easily. Here is a first example.

Lemma 3.14. *A matrix R in $\text{RREF}(j_1, j_2, \dots, j_r)$ has independent columns j_1, j_2, \dots, j_r and therefore rank r .*

Proof. Recall that the rank of a matrix is the number of independent columns, the ones that are not linear combinations of previous columns (Definition 2.10). Now, if a matrix R is in $\text{RREF}(j_1, j_2, \dots, j_r)$, the independent columns are precisely the ones with indices j_1, j_2, \dots, j_r . Indeed, in each of these columns, R makes a downward step and has a 1 in a row where all previous columns have zeros. Such a downward step column is therefore not a linear combination of the previous columns. But any other column \mathbf{v} is a linear combination of the previous (downward step) columns. There is no harm in skipping the easy but a bit technical formal proof in exchange for this “proof by example”:

$$\begin{array}{cccccccc}
 & 2 & 3 & & 6 & 8 & & \\
 \begin{array}{|c|c|c|c|c|c|c|c|}
 \hline
 & 1 & 0 & & 0 & v_1 & 0 & \\
 \hline
 & & 1 & & 0 & v_2 & 0 & \\
 \hline
 & & & & 1 & v_3 & 0 & \\
 \hline
 & & & & & & 1 & \\
 \hline
 & & & & & & & \\
 \hline
 & & & & & & & \\
 \hline
 & & & & & & & \\
 \hline
 \end{array}
 & \mathbf{v} = \begin{pmatrix} v_1 \\ v_2 \\ v_3 \\ 0 \\ 0 \\ 0 \\ 0 \end{pmatrix} = v_1 \mathbf{e}_1 + v_2 \mathbf{e}_2 + v_3 \mathbf{e}_3
 \end{array}$$

$\mathbf{e}_1 \mathbf{e}_2 \quad \mathbf{e}_3 \mathbf{v} \mathbf{e}_4$

However, in order to understand how a formal proof uses Definition 3.13, let us do it anyway. Suppose $\mathbf{v} \in \mathbb{R}^m$ is the j -th column of R (where j is not one of j_1, j_2, \dots, j_r). There are two cases.

The first case is $j > j_r$, so \mathbf{v} is to the right of all downward step columns. For all $i > r$, entry v_i is below the staircase by Definition 3.13 (a) and hence equal to 0. In this case, \mathbf{v} is an “obvious” linear combination of the previous (meaning all) downward step columns, since those are the first r standard unit vectors by Definition 3.13 (i):

$$\mathbf{v} = \sum_{i=1}^r v_i \mathbf{e}_i \quad (\text{using } v_i = 0 \text{ for } i > r).$$

The second case is $j < j_r$; let s be the smallest index such that $j < j_s$ (in the picture above, $s = 4$ and $j_s = 8$). Thus, $j < j_s, j_{s+1}, \dots, j_r$. By Definition 3.13 (b), entries $v_i, i \geq s$, are below the staircase and thus 0. Then \mathbf{v} is an obvious linear combination of the previous downward step columns $j_1, j_2, \dots, j_{s-1} < j$, the first $s - 1$ standard unit vectors:

$$\mathbf{v} = \sum_{i=1}^{s-1} v_i \mathbf{e}_i \quad (\text{using } v_i = 0 \text{ for } i \geq s).$$

□

3.3.2 Direct solution

It is easy to solve a system $R\mathbf{x} = \mathbf{c}$ (or decide that there is no solution) whenever R is in RREF. In fact, this is even easier than doing back substitution for a system $U\mathbf{x} = \mathbf{c}$ with U an upper triangular matrix (see Section 3.2.1). This is why we call the method “direct solution”. The key are the standard unit vectors in the downward step columns.

Suppose that R is an $m \times n$ matrix in $\text{RREF}(j_1, j_2, \dots, j_r)$. There are two cases: if $c_i \neq 0$ for some $i > r$, then $R\mathbf{x} = \mathbf{c}$ has no solution, because already the i -th equation $\mathbf{r}_i^\top \mathbf{x} = c_i$ has no solution. Here, \mathbf{r}_i^\top denotes the i -th row of R which is below the staircase and hence equal to $\mathbf{0}^\top$ by Definition 3.13 (a). Hence, $\mathbf{r}_i^\top \mathbf{x} = 0 \neq c_i$, no matter what \mathbf{x} is. For this argument, we made use of the rowwise interpretation of a system of linear equations, see the discussion after Definition 3.1.

The other case is that $c_i = 0$ for all $i > r$. In this case, there is a *canonical* (standard) solution: we define $\mathbf{x} \in \mathbb{R}^n$ by

$$x_j = \begin{cases} c_i, & \text{if } j = j_i \text{ for some } i \\ 0, & \text{otherwise} \end{cases} \quad (3.3)$$

This has the effect that in the linear combination $R\mathbf{x}$ (see Definition 2.4 of matrix-vector multiplication), only the r downward step columns $\mathbf{e}_1, \mathbf{e}_2, \dots, \mathbf{e}_r$ have (potentially) non-zero scalars c_1, c_2, \dots, c_r , and these are exactly the right ones such that $R\mathbf{x} = \mathbf{c}$. Here is an illustration.

$$\begin{array}{cccc}
 & j_1 & j_2 & & j_3 & j_4 \\
 \begin{array}{|c|c|c|c|c|c|c|c|}
 \hline
 & 1 & 0 & & 0 & 0 & & \\
 \hline
 & & 1 & & 0 & 0 & & \\
 \hline
 & & & & 1 & 0 & & \\
 \hline
 & & & & & 1 & & \\
 \hline
 & & & & & & & \\
 \hline
 & & & & & & & \\
 \hline
 \end{array}
 & = &
 \begin{array}{|c|}
 \hline
 0 \\
 \hline
 c_1 \\
 \hline
 c_2 \\
 \hline
 0 \\
 \hline
 c_3 \\
 \hline
 0 \\
 \hline
 c_4 \\
 \hline
 0 \\
 \hline
 0 \\
 \hline
 \end{array}
 & = &
 \begin{array}{|c|}
 \hline
 c_1 \\
 \hline
 c_2 \\
 \hline
 c_3 \\
 \hline
 c_4 \\
 \hline
 0 \\
 \hline
 0 \\
 \hline
 0 \\
 \hline
 0 \\
 \hline
 \end{array}
 & \leftarrow \text{if } \neq 0 \text{ here, no solution} \\
 & & \mathbf{c} \\
 & & \mathbf{x}
 \end{array}$$

Formally, since column j_i of R is \mathbf{e}_i by Definition 3.13 (i), the vector \mathbf{x} as in (3.3) indeed gives

$$R\mathbf{x} = \sum_{i=1}^r c_i \mathbf{e}_i = \mathbf{c} \quad (\text{using } c_i = 0 \text{ for } i > r).$$

We note that the vector \mathbf{x} in (3.3) is typically not the only solution of $R\mathbf{x} = \mathbf{c}$. Our running example is a 6×10 matrix whose first column is $\mathbf{0}$. Hence, we could for example change the canonical solution by setting x_1 to any value (instead of 0), and we would still have a solution. Section 4.4 discusses how to find all solutions.

Algorithm 5 gives the pseudocode for the direct solution method. Its operations do not depend on R , only on where the downward step columns are. The runtime bound immediately follows from inspecting the pseudocode.

Algorithm 5 Direct solution:

Returns a pair $(\mathbf{x}, \text{result})$ such that $R\mathbf{x} = \mathbf{c}$ if $\text{result} = \text{"solution"}$. If $\text{result} = \text{"no solution"}$, there is no solution. The matrix R must be in $\text{RREF}(j_1, j_2, \dots, j_r)$.

```

1: function DIRECT SOLUTION( $R, j_1, j_2, \dots, j_r, \mathbf{c}$ )  $\triangleright R \in \mathbb{R}^{m \times n}, \mathbf{c} \in \mathbb{R}^m$ 
2:    $\mathbf{x} \leftarrow \mathbf{0} \in \mathbb{R}^n$ 
3:   if  $c_i \neq 0$  for some  $i > r$  then
4:     return  $(\mathbf{x}, \text{"no solution"})$ 
5:   end if
6:   for  $i = 1, 2, \dots, r$  do
7:      $x_{j_i} \leftarrow c_i$ 
8:   end for
9:   return  $(\mathbf{x}, \text{"solution"})$ 
10: end function

```

Theorem 3.15 (Runtime of Direct solution). *Let $R\mathbf{x} = \mathbf{c}$ be a system of m linear equations in n variables, where R is in $\text{RREF}(j_1, j_2, \dots, j_r)$. In time*

$$O(m + n),$$

direct solution (Algorithm 5) returns a solution \mathbf{x} or reports that there is no solution.

3.3.3 Elimination

Here, we proceed as in Gauss elimination (Section 3.2.2) and use row operations to transform any system $A\mathbf{x} = \mathbf{b}$ into a system $R\mathbf{x} = \mathbf{c}$ with the same solutions, where R is in RREF. After this, we can use direct solution (see previous section) to either report that the system is unsolvable, or to compute the canonical solution.

In the following example, we only show how elimination transforms A into R . The pseudocode in Algorithm 6 below will also take care of the right-hand side(s).

As in Gauss elimination, we proceed column by column. The nonzero pivots that we find will determine the “downward step” columns j_1, j_2, \dots of the resulting matrix in RREF. Our example matrix is the 3×5 matrix

$$A = \begin{bmatrix} 2 & 4 & 2 & 2 & -2 \\ 6 & 12 & 6 & 7 & 1 \\ 4 & 8 & 2 & 2 & 6 \end{bmatrix}. \quad (3.4)$$

Already in row 1 of column 1, we have a nonzero pivot that we can use for a downward step in this column:

$$\begin{bmatrix} \mathbf{2} & 4 & 2 & 2 & -2 \\ 6 & 12 & 6 & 7 & 1 \\ 4 & 8 & 2 & 2 & 6 \end{bmatrix}$$

In contrast to Gauss elimination, we first apply a *row division* in order to make the pivot equal to 1. This is because the first downward step column needs to be transformed into the first standard unit vector e_1 , according to the requirements of RREF (Definition 3.13). Only after that, we perform row subtractions to eliminate the nonzero entries in column 1. As a bonus, this requires no further divisions:

$$\begin{array}{l}
 \begin{bmatrix} \mathbf{2} & 4 & 2 & 2 & -2 \\ 6 & 12 & 6 & 7 & 1 \\ 4 & 8 & 2 & 2 & 6 \end{bmatrix} \\
 \text{divide (row 1) by 2:} \quad \downarrow \\
 \begin{bmatrix} \mathbf{1} & 2 & 1 & 1 & -1 \\ 6 & 12 & 6 & 7 & 1 \\ 4 & 8 & 2 & 2 & 6 \end{bmatrix} \\
 \text{subtract } 6 \cdot (\text{row 1}) \text{ from (row 2):} \quad \downarrow \\
 \begin{bmatrix} \mathbf{1} & 2 & 1 & 1 & -1 \\ 0 & 0 & 0 & 1 & 7 \\ 4 & 8 & 2 & 2 & 6 \end{bmatrix} \\
 \text{subtract } 4 \cdot (\text{row 1}) \text{ from (row 3):} \quad \downarrow \\
 \begin{bmatrix} \mathbf{1} & 2 & 1 & 1 & -1 \\ 0 & 0 & 0 & 1 & 7 \\ 0 & 0 & -2 & -2 & 10 \end{bmatrix}
 \end{array}$$

We have made our first downward step and move on to column 2 where we attempt to make the next downward step in row 2:

$$\begin{bmatrix} 1 & 2 & 1 & 1 & -1 \\ 0 & \mathbf{0} & 0 & 1 & 7 \\ 0 & \mathbf{0} & -2 & -2 & 10 \end{bmatrix}$$

But we have a zero pivot, and no row exchange can bring a nonzero entry from further down into the pivot position. In Gauss elimination, we called this “the ugly case” and simply gave up. But here, this is an unusually good case. There is simply no downward step to be made in column 2, and we can directly move on to column 3. In this column, it is possible to make a row exchange to get a nonzero pivot in row 2:

$$\begin{array}{l}
 \begin{bmatrix} 1 & 2 & 1 & 1 & -1 \\ 0 & 0 & \mathbf{0} & 1 & 7 \\ 0 & 0 & -2 & -2 & 10 \end{bmatrix} \\
 \text{exchange (row 2) and (row 3):} \quad \downarrow \\
 \begin{bmatrix} 1 & 2 & 1 & 1 & -1 \\ 0 & 0 & -\mathbf{2} & -2 & 10 \\ 0 & 0 & 0 & 1 & 7 \end{bmatrix} \\
 \text{divide (row 2) by } -2: \quad \downarrow \\
 \begin{bmatrix} 1 & 2 & 1 & 1 & -1 \\ 0 & 0 & \mathbf{1} & 1 & -5 \\ 0 & 0 & 0 & 1 & 7 \end{bmatrix}
 \end{array}$$

So we make the second downward step in column 3, but in order to transform this column into the second standard unit vector e_2 , we also need to eliminate *above* the pivot:

$$\begin{array}{l} \text{subtract } 1 \cdot (\text{row } 2) \text{ from (row } 1): \\ \begin{bmatrix} 1 & 2 & 1 & 1 & -1 \\ 0 & 0 & 1 & 1 & -5 \\ 0 & 0 & 0 & 1 & 7 \end{bmatrix} \end{array} \quad \downarrow$$

$$\begin{bmatrix} 1 & 2 & 0 & 0 & 4 \\ 0 & 0 & 1 & 1 & -5 \\ 0 & 0 & 0 & 1 & 7 \end{bmatrix}$$

No elimination below the pivot is necessary here, so our downward step is done, and we move to column 4. We already have a pivot of 1 in the third row, so we need no row exchange and no row division. All that is left to do for this third downward step is one row subtraction above the pivot to produce e_3 :

$$\begin{array}{l} \text{subtract } 1 \cdot (\text{row } 3) \text{ from (row } 2): \\ \begin{bmatrix} 1 & 2 & 0 & 0 & 4 \\ 0 & 0 & 1 & 1 & -5 \\ 0 & 0 & 0 & 1 & 7 \end{bmatrix} \end{array} \quad \downarrow$$

$$R = \begin{bmatrix} 1 & 2 & 0 & 0 & 4 \\ 0 & 0 & 1 & 0 & -12 \\ 0 & 0 & 0 & 1 & 7 \end{bmatrix}$$

Further downward steps are neither possible nor necessary, so we can stop even before having looked at the last column. Indeed, the matrix R that we have now is in RREF(1, 3, 4), without any zero rows at the end:

$$\begin{bmatrix} 1 & 2 & 0 & 0 & 4 \\ 0 & 0 & 1 & 0 & -12 \\ 0 & 0 & 0 & 1 & 7 \end{bmatrix} \quad (3.5)$$

But in general, the result of Gauss-Jordan elimination can be a matrix R with trailing zero rows. As a simple example, think of the same starting matrix A as in (3.4), with a zero row appended in the end. This zero row will then survive until the final matrix R .

Algorithm 6 presents the full Gauss Jordan elimination algorithm. As in Algorithm 3 for Gauss elimination, we write down a variant that transforms m right-hand sides at the same time (collected in a matrix B). There is a variable r whose value at any time corresponds to the number of downward steps already made.

There are also two special commands, **break** and **continue**. The **break** command leaves its surrounding loop and jumps to the command directly after the loop. We use this in line 5 when we have already made the maximum of m downward steps and are done. The **continue** command skips everything that happens after it in the surrounding loop, but does not leave the loop; so we immediately jump to the beginning of the next loop

repetition. We use this in line 12 to handle the (formerly ugly, but now unusually good) case of a zero pivot that cannot be fixed by row exchanges.

Algorithm 6 Gauss Jordan elimination with m right-hand sides:

Returns a sequence $(R, j_1, j_2, \dots, j_r, C)$ such that R is in $\text{RREF}(j_1, j_2, \dots, j_r)$, and for all $j \in [m]$, the two systems $Ax = \mathbf{b}_j$ and $Rx = \mathbf{c}_j$ have the same solutions, where \mathbf{b}_j is the j -th column of B , and \mathbf{c}_j the j -th column of C .

```

1: function GAUSS-JORDAN ELIMINATION( $A, B$ )                                 $\triangleright A \in \mathbb{R}^{m \times n}, B \in \mathbb{R}^{m \times m}$ 
2:    $R \leftarrow A, C \leftarrow B, r \leftarrow 0$                              $\triangleright r$ : number of downward steps so far
3:   for  $j = 1, 2, \dots, n$  do                                             $\triangleright$  eliminate in column  $j$ 
4:     if  $r = m$  then                                                     $\triangleright$  no further downward steps possible, done!
5:       break                                                             $\triangleright \dots$  loop execution and go to line 21
6:     end if
7:      $s \leftarrow r + 1$                                                    $\triangleright$  row of (potential) next downward step
8:     if  $u_{sj} = 0$  then                                                   $\triangleright$  zero pivot
9:       if there is some  $k > s$  such that  $u_{kj} \neq 0$  then
10:        exchange (row  $s$ ) and (row  $k$ ) (in both  $R$  and  $C$ )                 $\triangleright$  row operation
11:      else                                                             $\triangleright$  no downward step in column  $j$ 
12:        continue                                                        $\triangleright \dots$  in line 3 with the next column
13:      end if
14:    end if                                                             $\triangleright$  now  $u_{sj} \neq 0$ 
15:    divide (row  $s$ ) by  $u_{sj}$  (in both  $R$  and  $C$ )                             $\triangleright$  row operation
16:    for  $i = 1, 2, \dots, s - 1$  and  $i = s + 1, s + 2, \dots, m$  do         $\triangleright$  make  $u_{ij} = 0$ 
17:      subtract  $u_{ij} \cdot$  (row  $s$ ) from (row  $i$ ) (in both  $R$  and  $C$ )           $\triangleright$  row operation
18:    end for                                                             $\triangleright$  now, the  $j$ -th column of  $R$  equals  $\mathbf{e}_s$ 
19:     $r \leftarrow r + 1, j_r \leftarrow j$                                  $\triangleright$  next downward step was made in column  $j$ 
20:  end for
21:  return  $(R, j_1, j_2, \dots, j_r, C)$                                  $\triangleright R$  is in  $\text{RREF}(j_1, j_2, \dots, j_r)$ 
22: end function

```

Theorem 3.16 (Runtime of Gauss-Jordan elimination with m right-hand sides). *Let $Ax = \mathbf{b}_j, j \in [m]$, be m systems of m linear equations in n variables, where the \mathbf{b}_j 's are the columns of the input matrix B . In time*

$$O(m^2(m + n)),$$

Gauss-Jordan elimination (Algorithm 6) returns equivalent systems $Rx = \mathbf{c}_j, j \in [m]$, where the \mathbf{c}_j 's are the columns of the output matrix C , and R is in $\text{RREF}(j_1, j_2, \dots, j_r)$.

Proof. As in Gauss elimination, the runtime is dominated by the row operations. A row operation now takes time $O(m + n)$, since a row has n entries from R and m entries from C . The number of row operations is $O(m^2)$, because there are at most m downward steps, and in each of them, all m rows are being updated through row operations. The resulting bound of $O(m^2(m + n))$ also covers all other operations. \square

3.3.4 Standard form and CR decomposition

The theoretical implications of Algorithm 6 are far-reaching. As an immediate result, we get that every matrix A can be transformed into reduced row echelon form, our standard form according to Definition 3.13. The following theorem says what this precisely means.

Theorem 3.17 (Output of Gauss-Jordan elimination). *Let A be an $m \times n$ matrix, and let $(R, j_1, j_2, \dots, j_r, M)$ be the output of Algorithm 6 with input (A, I) , where I is the $m \times m$ identity matrix. Then M is invertible, $R = MA$, and R is in $\text{RREF}(j_1, j_2, \dots, j_r)$.*

Proof. Starting from the two matrices A and I , Algorithm 6 is repeatedly applying row operations to both. These operations are row subtractions and row exchanges as also used by Gauss elimination, as well as row divisions. In Section 3.2.3, we have seen that such row operations correspond to multiplication with an invertible matrix from the left. This also applies to row divisions as the new kind of row operations that Algorithm 6 is using (you are invited to write down the matrix for dividing row s by scalar $\lambda \neq 0$).

Therefore, the final right-hand side matrix $C = M$ (obtained after some number ℓ of row operations, starting from $B = I$) is $M = M_\ell M_{\ell-1} \dots M_1 I = M_\ell M_{\ell-1} \dots M_1$, where M_i is the (invertible) matrix of the i -th row operation. Since the product of invertible matrices is invertible (see Lemma 2.59 and the discussion after it), M is invertible. The final matrix R in $\text{RREF}(j_1, j_2, \dots, j_r)$ is obtained from A via the same row operations, so we have $R = M_\ell M_{\ell-1} \dots M_1 A = MA$. \square

In this proof, we have silently assumed that Algorithm 6 is correct in the sense that it indeed produces a final matrix R in RREF. A formal proof of this is possible (by induction on n , the number of columns of A), but this would not provide greater insights than our informal explanation of Gauss-Jordan elimination starting on page 118.

In Algorithm 6, we have some choices: in a row exchange, there can be several candidate rows k , and we are free to pick any of them for the exchange. Depending on these choices, the same input (A, I) could lead to different outputs. But maybe surprisingly, the resulting matrix R will always be the same.

Theorem 3.18 (Uniqueness of RREF, and relation to the CR decomposition). *Let A be an $m \times n$ matrix. There is a unique $m \times n$ matrix R (the one resulting from Gauss-Jordan elimination on A according to Theorem 3.17), with the following two properties.*

- (i) $R = MA$ for some invertible $m \times m$ matrix M .
- (ii) R is in RREF.

More precisely, R is in $\text{RREF}(j_1, j_2, \dots, j_r)$, where j_1, j_2, \dots, j_r are the indices of the independent columns in A , and

$$R = \left[\begin{array}{c} \underbrace{R'}_{r \times n} \\ \underbrace{0}_{(m-r) \times n} \end{array} \right],$$

with R' the unique matrix such that $A = CR'$ in Theorem 2.46 (CR decomposition).

It is therefore justified to call the matrix R in the theorem *the standard form* of A . Moreover, from R we get the rank r of A as well as the CR decomposition $A = CR'$, where C is the submatrix of A containing columns j_1, j_2, \dots, j_r , and R' is the submatrix of R containing the first r rows.

Let us illustrate this with an example. In Section 2.3.5, we have considered the 3×4 matrix

$$A = \begin{bmatrix} 1 & 2 & 0 & 3 \\ 2 & 4 & 1 & 4 \\ 3 & 6 & 2 & 5 \end{bmatrix}$$

and manually computed its CR decomposition

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

To get this via Gauss-Jordan elimination, we first transform A into RREF. This is a particularly simple case, since no row exchanges and row divisions are necessary, and also no eliminations above the pivot. All it takes are two row subtractions in the first column, and one in the third column:

$$\begin{array}{l} A = \begin{bmatrix} 1 & 2 & 0 & 3 \\ 2 & 4 & 1 & 4 \\ 3 & 6 & 2 & 5 \end{bmatrix} \\ \text{elimination in column 1:} \quad \begin{bmatrix} 1 & 2 & 0 & 3 \\ 0 & 0 & 1 & -2 \\ 0 & 0 & 2 & -4 \end{bmatrix} \downarrow \\ \text{elimination in column 3:} \quad \begin{bmatrix} 1 & 2 & 0 & 3 \\ 0 & 0 & 1 & -2 \\ 0 & 0 & 0 & 0 \end{bmatrix} \downarrow \\ R = \begin{bmatrix} 1 & 2 & 0 & 3 \\ 0 & 0 & 1 & -2 \\ 0 & 0 & 0 & 0 \end{bmatrix} \end{array}$$

The resulting matrix R is in RREF(1, 3), so $r = 2$ and Theorem 3.18 tells us that columns 1 and 3 are the independent ones in A , which then leads to submatrices

$$C = \begin{bmatrix} 1 & 0 \\ 2 & 1 \\ 3 & 2 \end{bmatrix} \text{ (of } A) \text{ and } R' = \begin{bmatrix} 1 & 2 & 0 & 3 \\ 0 & 0 & 1 & -2 \end{bmatrix} \text{ (of } R).$$

These matrices are precisely the ones that we also found back in Section 2.3.5.

Proof of Theorem 3.18. We know from Theorem 3.17 that Gauss-Jordan elimination on A produces *some* matrix R satisfying properties (i) and (ii). Now we show that *every* matrix R satisfying these two properties is the one given under “More precisely...” which also shows uniqueness. So let R be such a matrix.

Because of $R = MA$ with M invertible, invariance Lemma 3.6 applies and shows that the independent columns j_1, j_2, \dots, j_r of A are also the independent columns of R . So we have R in $\text{RREF}(j_1, j_2, \dots, j_r)$, see Lemma 3.14. Hence, R is of the form

$$R = \left[\begin{array}{c} \underbrace{R'}_{r \times n} \\ \hline \underbrace{0}_{(m-r) \times n} \end{array} \right]$$

for *some* matrix R' , using Definition 3.13 (a). It remains to show that $A = CR'$, where C is the submatrix containing the independent columns of A , the ones at indices j_1, j_2, \dots, j_r .

We actually show

$$MA = MCR', \quad (3.6)$$

and multiplying this from the left with M^{-1} cancels M , resulting in $A = CR'$. For (3.6), the key is the matrix MC which is the submatrix of MA with columns j_1, j_2, \dots, j_r . This may be clear to you, but if not, here is the argument: if A has columns $\mathbf{v}_1, \mathbf{v}_2, \dots, \mathbf{v}_n$, then C has columns $\mathbf{v}_{j_1}, \mathbf{v}_{j_2}, \dots, \mathbf{v}_{j_r}$. By Definition 2.36 of matrix multiplication, MA has columns $M\mathbf{v}_1, M\mathbf{v}_2, \dots, M\mathbf{v}_n$ and MC has columns $M\mathbf{v}_{j_1}, M\mathbf{v}_{j_2}, \dots, M\mathbf{v}_{j_r}$. Hence, MC is indeed the submatrix of MA with columns j_1, j_2, \dots, j_r .

Since $MA = R$ is in $\text{RREF}(j_1, j_2, \dots, j_r)$, we know the submatrix MC : it has the standard unit vectors $\mathbf{e}_1, \mathbf{e}_2, \dots, \mathbf{e}_r \in \mathbb{R}^m$ as columns, see Definition 3.13 (i). This statement can alternatively be written in the form

$$MC = \underbrace{\left[\begin{array}{c} \underbrace{I}_{r \times r} \\ \hline \underbrace{0}_{(m-r) \times r} \end{array} \right]}_{m \times r},$$

and hence (3.6) follows via

$$MCR' = \left[\begin{array}{c} \underbrace{I}_{r \times r} \\ \hline \underbrace{0}_{(m-r) \times r} \end{array} \right] R' = \underbrace{\left[\begin{array}{c} \underbrace{R'}_{r \times n} \\ \hline \underbrace{0}_{(m-r) \times n} \end{array} \right]}_R = MA.$$

Here, we use that matrix multiplication is “rows of the left matrix times the right matrix”, see Observation 2.45; so we can put the product together from the partial products $IR' = R'$ (the first r rows) and $0R' = 0$ (the last $m - r$ rows). \square

3.3.5 Computing inverse matrices, and solving $A\mathbf{x} = \mathbf{b}$ from R and M

Gauss-Jordan elimination provides a direct way of computing inverse matrices. This is even simpler than doing it via Gauss elimination as in Section 3.2.6.

Theorem 3.19 (Computing inverses with Gauss-Jordan elimination). *Let A be an $m \times m$ matrix, and let $(R, j_1, j_2, \dots, j_r, M)$ be the output of running Algorithm 6 with input (A, I) . Then A is invertible if and only if $R = I$, and in this case, $A^{-1} = M$.*

Proof. Since $R = MA$ by Theorem 3.17, the two matrices A and R have their independent columns at the same positions (see invariance Lemma 3.6).

A is invertible if and only if A has linearly independent columns (see Definition 2.55), and this is equivalent to all columns of A being independent, see Definition 2.10 and Corollary 1.23 (iii). Hence, A is invertible if and only if its independent columns are at indices $1, 2, \dots, m$ (all indices). For R , this says that R is in $\text{RREF}(1, 2, \dots, m)$ (see Lemma 3.14), and this is equivalent to $R = I$ by Definition 3.13 (i).

Therefore, if A is invertible, the equation $R = MA$ reads as $I = MA$, so $M = A^{-1}$ by Definition 2.57 of the inverse matrix. \square

Finally, we come back to the ultimate goal of this chapter: solve an *arbitrary* system $Ax = b$ of linear equations (or report that there is no solution). With Gauss-Jordan elimination, we can now do this and even be more efficient if several systems share the same coefficient matrix.

Theorem 3.20 (Solving $Ax = b$ with Gauss-Jordan elimination). *Let A be an $m \times n$ matrix, and let $(R, j_1, j_2, \dots, j_r, M)$ be the output of Algorithm 6 with input (A, I) . According to Theorem 3.16, this output is produced in time*

$$O(m^2(m+n)).$$

Given any right-hand side $b \in \mathbb{R}^m$, we can compute a solution x of the system $Ax = b$ (or report that there is no solution) in time

$$O(m^2 + n).$$

Proof. By Theorem 3.17, M is invertible, $R = MA$ holds, and R is in $\text{RREF}(j_1, j_2, \dots, j_r)$.

Invariance Lemma 3.2 shows that the two systems $Ax = b$ and $MAx = Mb$ have the same solutions, using that M is invertible. Using $R = MA$ and $c := Mb$, we arrive at the equivalent system $Rx = c$. The vector c can be computed in time $O(m^2)$ using matrix-vector multiplication (in table notation according to Observation 2.6, it is easiest to the runtime). Using that R is in RREF , we can apply direct solution and solve $Rx = c$ (or report that there is no solution) in time $O(m+n)$, see Theorem 3.15. In total, we need time $O(m^2 + m + n) = O(m^2 + n)$. \square

If $m = n$ (the square case), we see a behavior as in Section 3.2.7 for solving $Ax = b$ from A^{-1} (if the inverse exists): After spending $O(m^3)$ preprocessing time for A , we can solve $Ax = b$ in time $O(m^2)$ per right-hand side. This is by a factor of m faster than a fresh elimination for every right-hand side.

If $m < n$ (the quite typical wide case; see Figure 2.1), preprocessing takes time $O(m^2n)$, after which we only need time $(m^2 + n)$ per right-hand side. For $n \geq m^2$ (the *very* wide case), this is $O(n)$ and therefore even by a factor of m^2 faster than a fresh Gauss-Jordan elimination for every right-hand side.

Chapter 4

The Three Fundamental Subspaces

So far, we have said that vectors are elements of some space \mathbb{R}^m , and for $m = 2, 3$, we have drawn them as arrows in the 2-dimensional plane or in 3-dimensional space. But this is by far not the full picture. In fact, we have already seen other spaces: the column space, the row space, and the nullspace of an $m \times n$ matrix A . These *three fundamental subspaces*¹ (of a matrix) are the main characters of this chapter. The column space is a subspace of \mathbb{R}^m , while row space and nullspace are subspaces of \mathbb{R}^n . Here, “subspace” is still an informal notion, used for a space that occupies some part of another space.

But what is a “space”, actually? Officially, \mathbb{R}^m is just a set, and the column space of an $m \times n$ matrix is just a subset of it. We call a set a space when we think of it as a “natural habitat” for vectors. Then the question is: what is it that makes a set a natural habitat for vectors? We know that vectors have two natural behaviors: they add up ($\mathbf{v} + \mathbf{w}$), and they scale ($\lambda \mathbf{v}$). In a natural habitat, they should be able to do that, but not every set is a natural habitat in this sense. For example, the surface of the earth is a natural habitat for humans, but not for vectors (think of vectors “living on earth” as arrows from the center of the earth to the surface). Adding up such vectors or scaling them unfortunately leads to vectors hanging in the air, or being buried underground, so natural vector behavior is not possible on the surface of the earth.



In mathematics, a natural habitat for vectors is formalized as a *vector space*. This is an abstract notion that covers all \mathbb{R}^m 's, the three fundamental subspaces of a matrix, and many other spaces.

A vector space is the right level of abstraction if we want to develop some theory that does not only work in \mathbb{R}^m , but also in other spaces such as the three fundamental subspaces. The main theoretical terms that we introduce in this chapter are *subspace*, *basis*, *dimension*, and *isomorphism*.

In the end, we will apply them to the three fundamental subspaces and also answer a question that we have left open in Chapter 3: how can we compute *all* solutions of a system of linear equations $A\mathbf{x} = \mathbf{b}$?

¹Strang [Str23] has four fundamental subspaces, but we omit the left nullspace here, as we think it is less relevant.

4.1 Vector spaces

In this section, we introduce the abstract concept of a vector space as a “natural habitat” for vectors. We will see that the \mathbb{R}^m ’s are (important) examples of vector spaces, but we also get to know other examples. The vector space abstraction allows us to view subspaces such as the column space of a matrix as vector spaces themselves.

4.1.1 Definition and examples

On a high-level, a vector space is a set whose elements (which we call vectors) can add up ($\mathbf{v} + \mathbf{w}$) and scale ($\lambda \mathbf{v}$), without the results leaving the set. If the set in question is some \mathbb{R}^m , we have used this property without giving it any thought. For example, if you add up two vectors in \mathbb{R}^m , you obviously get another vector in \mathbb{R}^m (see Definition 1.2). If the set in question is not some \mathbb{R}^m , we cannot take this property for granted (see the “surface of the earth” example in the chapter introduction). In more exotic spaces, it might not even be clear what it means to add up two elements.

The following definition of a vector space takes care of this. It says precisely under which conditions a set can be called a natural habitat for vectors. These conditions are completely abstract, and on this abstract level, there is only one sensible answer to the question “What is a vector?” This answer is that a vector is an element of a vector space.

To make things a bit less abstract, we only talk about *real vector spaces* here. The word *real* does not stand for *true* or *proper*, but indicates that the scalars are real numbers. Each \mathbb{R}^m is a real vector space, and immediately after the definition, we will see another example of a real vector space. There are also vector spaces where the scalars are other kinds of numbers (*complex numbers* are an important case, and so are *bits*, the elements of the set $\{0, 1\}$), but we will not discuss them here. So we omit the word *real*, simply say vector space, but mean *real* vector space.

Do not be scared by the length of the following definition; in particular, there is no need to learn the axioms by heart. You can look them up whenever needed.

Definition 4.1 (Vector space). A vector space is a triple $(V, +, \cdot)$ where V is a set (the vectors), and

$$\begin{aligned} + & : V \times V \rightarrow V \text{ is a function (vector addition),} \\ \cdot & : \mathbb{R} \times V \rightarrow V \text{ is a function (scalar multiplication),} \end{aligned}$$

satisfying the following axioms of a vector space for all $\mathbf{u}, \mathbf{v}, \mathbf{w} \in V$ and all $\lambda, \mu \in \mathbb{R}$.

- | | |
|---|--|
| 1. $\mathbf{v} + \mathbf{w} = \mathbf{w} + \mathbf{v}$ | commutativity |
| 2. $\mathbf{u} + (\mathbf{v} + \mathbf{w}) = (\mathbf{u} + \mathbf{v}) + \mathbf{w}$ | associativity |
| 3. There is a vector $\mathbf{0}$ such that $\mathbf{v} + \mathbf{0} = \mathbf{v}$ for all \mathbf{v} | zero vector |
| 4. There is a vector $-\mathbf{v}$ such that $\mathbf{v} + (-\mathbf{v}) = \mathbf{0}$ | negative vector |
| 5. $1 \cdot \mathbf{v} = \mathbf{v}$ | identity element |
| 6. $(\lambda \cdot \mu) \mathbf{v} = \lambda \cdot (\mu \cdot \mathbf{v})$ | compatibility of \cdot and \cdot in \mathbb{R} |
| 7. $\lambda(\mathbf{v} + \mathbf{w}) = \lambda \mathbf{v} + \lambda \mathbf{w}$ | distributivity over $+$ |
| 8. $(\lambda + \mu) \mathbf{v} = \lambda \mathbf{v} + \mu \mathbf{v}$ | distributivity over $+$ in \mathbb{R} |

Here, we use red color to indicate that there are two different “+”, and also two different “·”. The red ones stand for the normal addition and multiplication of real numbers. The black ones are for vector addition and scalar multiplication. We will omit the coloring (and even the “·”) in the following, but there is (hopefully) only limited potential for confusion. After all, if you want to know which “+” or “·” is meant in a given context, you simply need to check what is on the left side and the right side. We have done this kind of *overloading* before when we used the normal addition symbol “+” both for real numbers and for vectors in \mathbb{R}^m .

Now for the actual axioms: with \mathbb{R}^m in mind, they seem obvious, given how we have defined vector addition and scalar multiplication. We therefore get

Observation 4.2. $(\mathbb{R}^m, +, \cdot)$, with “+” as in Definition 1.2 and “·” as in Definition 1.3, is a vector space.

Previously, we have called this vector space \mathbb{R}^m which—in hindsight—is an abuse of notation. But this is acceptable, since our “+” and “·” are what mathematicians call the *canonical* (standard) choices for vector addition and scalar multiplication in \mathbb{R}^m , so there is no strict need to mention them.

But in general, V could be any set, with $+$ and \cdot defined in non-canonical ways, so we explicitly need to include these functions and also make sure that they behave as expected, where the expectations come from what happens in \mathbb{R}^m . This is what the axioms are about.

To illustrate the concept, we present a new vector space, the vector space of *real polynomials* in one variable. We could also consider polynomials in several variables, but for the vector space example that we want to give here, one variable is enough. As for vector spaces, we omit the word *real*, since we do not consider any other polynomials here. You know polynomials from high school, an example is $2x^2 + x + 1$. What may be new to you is that polynomials can be considered as vectors in a vector space.

Definition 4.3 (Polynomial). A polynomial \mathbf{p} is a formal sum of the form

$$\mathbf{p} = \sum_{i=0}^m p_i x^i,$$

for some $m \in \mathbb{N}$. Here x is a variable, and the numbers $p_0, p_1, \dots, p_m \in \mathbb{R}$ are the coefficients of \mathbf{p} . The largest i such that $p_i \neq 0$ is the degree of \mathbf{p} . If all p_i are 0, we have the zero polynomial $\mathbf{0} = 0$ whose degree we define to be -1 .

We note that m does not have to be the same for all polynomials, but for every polynomial, we have some m . For example, $\mathbf{p} = 2x^2 + x + 1$ is a polynomial of degree 2, and $\mathbf{q} = 5x - 2$ is a polynomial of degree 1.

A formal sum is one that we cannot simplify (to one number, for example), because the individual summands are of different types (likes apples and oranges) that cannot be added up. Here, the different types are the different powers of x : $x^0 = 1, x, x^2, \dots$

To turn the set of polynomials into a vector space, we need to define addition of two polynomials, and multiplication of a polynomial with a scalar. There are canonical ways of doing this. In the example, we would define

$$(2x^2 + x + 1) + (5x - 2) = 2x^2 + 6x - 1,$$

i.e. we add corresponding powers of x . And scalar multiplication simply scales all coefficients, as in

$$5(2x^2 + x + 1) = 10x^2 + 5x + 5.$$

Theorem 4.4. Let $\mathbb{R}[x]$ denote the set of polynomials in one variable x . Given two polynomials $\mathbf{p} = \sum_{i=0}^m p_i x^i$ and $\mathbf{q} = \sum_{i=0}^n q_i x^i$, we define $\mathbf{p} + \mathbf{q}$ to be the polynomial

$$\mathbf{p} + \mathbf{q} = \sum_{i=0}^{\max(m,n)} (p_i + q_i) x^i,$$

where we set $p_i = 0$ for $i > m$ and $q_i = 0$ for $i > n$. For a scalar $\lambda \in \mathbb{R}$, we further define $\lambda \mathbf{p}$ as the polynomial

$$\lambda \mathbf{p} = \sum_{i=0}^m (\lambda p_i) x^i.$$

Then $(\mathbb{R}[x], +, \cdot)$ is a vector space.

We omit the proof, since it is quite boring; it boils down to checking the obvious. Here is a second example: the vector space of $m \times n$ matrices. Again, we omit the easy but boring proof.

Theorem 4.5. Let $\mathbb{R}^{m \times n}$ be the set of $m \times n$ matrices, with addition $A + B$ and scalar multiplication λA defined in the usual way, see Definition 2.2. Then $(\mathbb{R}^{m \times n}, +, \cdot)$ is a vector space.

Proving the obvious. As boring as this may be, it is still surprising that we only need to check 8 “obvious” axioms to guarantee proper behavior of a vector space. Indeed, there are many other things that we expect from knowing how things work in \mathbb{R}^m . For example, we expect that there is only one zero vector in a vector space $(V, +, \cdot)$, but this does not appear among the axioms. So we need to prove that it follows from the axioms.

Fact 4.6. Let $(V, +, \cdot)$ be a vector space. V contains exactly one zero vector (a vector satisfying axiom 3 of Definition 4.1: $\mathbf{v} + \mathbf{0} = \mathbf{v}$ for all \mathbf{v}).

Proof. Take two zero vectors $\mathbf{0}$ and $\mathbf{0}'$. Then

$$\begin{aligned} \mathbf{0}' &= \mathbf{0}' + \mathbf{0} \quad (\text{by axiom 3, since } \mathbf{0} \text{ is a zero vector}) \\ &= \mathbf{0} + \mathbf{0}' \quad (\text{by axiom 1, commutativity}) \\ &= \mathbf{0} \quad (\text{by axiom 3, since } \mathbf{0}' \text{ is a zero vector}). \end{aligned}$$

So $\mathbf{0}$ and $\mathbf{0}'$ are equal. □

Doing this may feel a little bit like learning to walk again after a serious leg injury: hard work for something that we have previously taken for granted. Here is another such “relearning step.”

Fact 4.7. *Let $(V, +, \cdot)$ be a vector space. For every $\mathbf{v} \in V$, there is exactly one negative vector $-\mathbf{v}$ (a vector satisfying axiom 4 of Definition 4.1: $\mathbf{v} + (-\mathbf{v}) = \mathbf{0}$).*

Proof. First of all, we need to realize that there is no way of simply computing $-\mathbf{v}$ as we do it in \mathbb{R}^m , by negating all entries. A vector \mathbf{v} might not have any such “entries”, and there is no “ $-$ ” operator in $(V, +, \cdot)$ that we could apply. In the proof, we can only use the 8 axioms (and everything we have already derived from them). We could attempt to compute $-\mathbf{v}$ as $(-1)\mathbf{v}$ using scalar multiplication; this can indeed be shown to produce a negative vector but does not rule out the existence of another negative vector. Here is how we go about that:

Take two negative vectors \mathbf{u} and \mathbf{u}' of \mathbf{v} . Then

$$\begin{aligned} \mathbf{u}' &= \mathbf{u}' + \mathbf{0} && \text{(by axiom 3, zero vector)} \\ &= \mathbf{u}' + (\mathbf{v} + \mathbf{u}) && \text{(by axiom 4, since } \mathbf{u} \text{ is a negative of } \mathbf{v}) \\ &= (\mathbf{u}' + \mathbf{v}) + \mathbf{u} && \text{(by axiom 2, associativity)} \\ &= (\mathbf{v} + \mathbf{u}') + \mathbf{u} && \text{(by axiom 1, commutativity)} \\ &= \mathbf{0} + \mathbf{u} && \text{(by axiom 4, since } \mathbf{u}' \text{ is a negative of } \mathbf{v}) \\ &= \mathbf{u} + \mathbf{0} && \text{(by axiom 1, commutativity)} \\ &= \mathbf{u} && \text{(by axiom 3, zero vector).} \end{aligned}$$

So \mathbf{u} and \mathbf{u}' are equal. □

Having understood why a vector space is a triple $(V, +, \cdot)$ and not simply a set V of vectors, we will continue our (now more educated) abuse of notation and still write V for the vector space, with the understanding that vector addition and scalar multiplication are clear from the context. We also still write $\mathbf{0}$ for the zero vector when V is clear from the context.

4.1.2 Subspaces

Definition 4.8 (Subspace). *Let V be a vector space. A nonempty subset $U \subseteq V$ is called a subspace of V if the following two axioms of a subspace are true for all $\mathbf{v}, \mathbf{w} \in U$ and all $\lambda \in \mathbb{R}$.*

(i) $\mathbf{v} + \mathbf{w} \in U$;

(ii) $\lambda \mathbf{v} \in U$.

These axioms guarantee that vector addition and scalar multiplication cannot take us out of the subspace. In this sense, a subspace is a natural “sub-habitat” within a bigger habitat V (see Figure 4.1) for some examples). As a consequence of the axioms, a subspace of V always contains at least the zero vector.

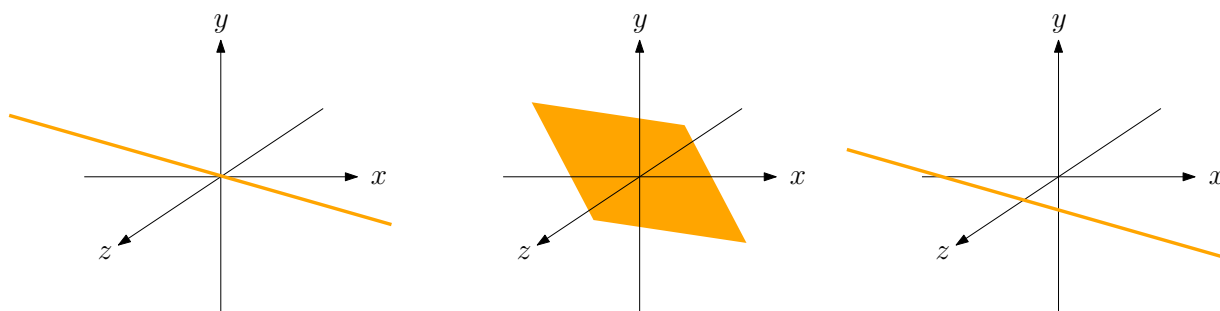


Figure 4.1: Subspaces of \mathbb{R}^3 : a line through $\mathbf{0}$ (all scalar multiples of one vector); a plane through $\mathbf{0}$ (all linear combinations of two linearly independent vectors); the right subset is not a subspace, since it misses $\mathbf{0}$.

Lemma 4.9. *Let $U \subseteq V$ be a subspace of a vector space V . Then $\mathbf{0} \in U$.*

Proof. Take any $\mathbf{u} \in U$ (U is nonempty). By subspace axiom (ii), we have $0\mathbf{u} = \mathbf{0} \in U$. \square

This proof seems quite clear, but it is actually incomplete. While the equation $0\mathbf{u} = \mathbf{0}$ certainly holds in any \mathbb{R}^m , this does not automatically mean that it holds in all vector spaces. We need to prove it—another case of learning to walk again. We promise, it is the last one! In fact, the axioms of vector spaces have been carefully designed by mathematicians before us, with the goal of ensuring that everything that seems obvious is actually true. So we will relax and rely on this in the future.

Fact 4.10. *Let V be a vector space, $\mathbf{v} \in V$. Then $0\mathbf{v} = \mathbf{0}$.*

Proof. This may not be the shortest proof (can you find a shorter one?), but it works:

$$\begin{aligned}
 & 0\mathbf{v} \\
 = & 0\mathbf{v} + \mathbf{0} && \text{(by axiom 3 of Definition 4.1, zero vector)} \\
 = & 0\mathbf{v} + (0\mathbf{v} + (-0\mathbf{v})) && \text{(by axiom 4, negative vector)} \\
 = & (0\mathbf{v} + 0\mathbf{v}) + (-0\mathbf{v}) && \text{(by axiom 2, associativity)} \\
 = & (0+0)\mathbf{v} + (-0\mathbf{v}) && \text{(by axiom 8, distributivity over } + \text{ in } \mathbb{R}) \\
 = & 0\mathbf{v} + (-0\mathbf{v}) && \text{(by the rules of } \mathbb{R}) \\
 = & \mathbf{0} && \text{(by axiom 4, negative vector)}
 \end{aligned}$$

\square

Figure 4.1 above gives two examples and one counterexample of subspaces of \mathbb{R}^3 . Next we see three subspaces that we have already encountered, without thinking about them as subspaces of a vector space: the column space, the row space, and the nullspace of a matrix. These are the three fundamental subspaces in the chapter title.

Lemma 4.11 (The column space is a subspace). *Let A be an $m \times n$ matrix. Then the column space $C(A) = \{Ax : x \in \mathbb{R}^n\}$ is a subspace of \mathbb{R}^m .*

Proof. Let v, w be in $C(A)$. Then there exist vectors $x, y \in \mathbb{R}^n$ such that $v = Ax, w = Ay$. Hence,

$$A(\underbrace{x+y}_{\in \mathbb{R}^n}) = Ax + Ay = v + w \Rightarrow v + w \in C(A).$$

This was subspace axiom (i). For axiom (ii), let $\lambda \in \mathbb{R}$. Then

$$A(\underbrace{\lambda x}_{\in \mathbb{R}^n}) = \lambda Ax = \lambda v \Rightarrow \lambda v \in C(A).$$

In both chain of equalities, the first equality comes from linearity of matrix transformations, see Lemma 2.19 and its proof. \square

Since the row space of a matrix is the column space of the transpose (Definition 2.14), we immediately get

Corollary 4.12 (The row space is a subspace). *Let A be an $m \times n$ matrix. Then the row space $R(A) = C(A^T)$ is a subspace of \mathbb{R}^n .*

Finally, the nullspace of a matrix (Definition 2.17) is also a subspace. We leave this as an exercise.

Exercise 4.13 (The nullspace is a subspace). *Let A be an $m \times n$ matrix. Then the nullspace $N(A) = \{x \in \mathbb{R}^n : Ax = 0\}$ is a subspace of \mathbb{R}^n .*

If A is an $m \times n$ matrix and $b \in \mathbb{R}^m$, the set of solutions of the system of linear equations $Ax = b$ is a subset of \mathbb{R}^n , but not a subspace if $b \neq 0$. Indeed, in this case, 0 is not a solution, so the set of solutions cannot be a subspace by Lemma 4.9. If $b = 0$, the set of solutions is the nullspace of A and therefore a subspace by Exercise 4.13.

Knowing that a subspace always contains 0 , we can actually say more.

Lemma 4.14 (Subspaces are vector spaces). *Let V be a vector space, and let U be a subspace of V . Then U is also a vector space (with the same “+” and “.” as V).*

Proof. Formally, to turn “+” and “.” into functions that work for U , we have to restrict their domains to $U \times U$ and $\mathbb{R} \times U$, respectively. The subspace axioms (i) and (ii) make sure that we can then also restrict their codomains to U without losing anything.

Next, we need to check the 8 axioms. All but axiom 4 are true for all vectors in V , since V is a vector space; in particular, they hold for all vectors in U , so there is nothing to check. In Case of axiom 3, we are also using that $0 \in U$ (Lemma 4.9). What remains is axiom 4: we need to make sure that for all $u \in U$, $-u$ is actually in U ; so far we only know that it is in V . But this holds, since $(-1)u \in U$ by subspace axiom (ii), and “obviously” $(-1)u = -u$. If you are up for it, you can prove the obvious, otherwise, you can safely believe it. \square

Subspaces of $\mathbb{R}[x]$. Let us look at some subspaces of $\mathbb{R}[x]$, the vector space of polynomials (Theorem 4.4). A *polynomial without constant term* is a polynomial of the form

$$\mathbf{p} = \sum_{i=1}^m p_i x^i.$$

An example is $x^2 + 3x$. It is clear that these polynomials form a subspace of $\mathbb{R}[x]$, since the sum of two polynomials without constant term is again a polynomial without constant term, and so is each scalar multiple of a polynomial without constant term.

A *quadratic polynomial* is a polynomial \mathbf{p} of the form

$$\mathbf{p} = p_0 + p_1 x + p_2 x^2.$$

We do not require $p_2 \neq 0$, so $3x$ is also a quadratic polynomial. Again, it is easy to see that the quadratic polynomials form a subspace of $\mathbb{R}[x]$. In fact, this subspace looks a lot like \mathbb{R}^3 : each quadratic polynomial \mathbf{p} is determined by three real numbers p_0, p_1, p_2 , so we can also describe it by a vector

$$\mathbf{v}_{\mathbf{p}} = \begin{pmatrix} p_0 \\ p_1 \\ p_2 \end{pmatrix} \in \mathbb{R}^3.$$

Moreover, “+” and “.” on quadratic polynomials translate to “+” and “.” on the corresponding vectors:

$$\mathbf{v}_{\mathbf{p}+\mathbf{q}} = \mathbf{v}_{\mathbf{p}} + \mathbf{v}_{\mathbf{q}}, \quad \mathbf{v}_{\lambda \mathbf{p}} = \lambda \mathbf{v}_{\mathbf{p}}. \quad (4.1)$$

Therefore, the subspace of quadratic polynomials is just \mathbb{R}^3 in disguise. The mathematical term is that the two spaces are *isomorphic*. We will formalize this in Section 4.2.5 below.

Subspaces of $\mathbb{R}^{m \times n}$. Let us turn to $\mathbb{R}^{m \times n}$, the vector space of $m \times n$ matrices (Theorem 4.5). Here, we first observe that this is not really a new vector space, as $\mathbb{R}^{m \times n}$ is isomorphic to \mathbb{R}^{mn} : an $m \times n$ matrix is one way of grouping mn numbers, an mn -dimensional vector is another way. In both cases, vector addition and scalar-multiplication are defined entry-wise, so it does not really matter how we group the numbers.

The difference is that we think of a vector in \mathbb{R}^{mn} as one column with mn entries (Definition 1.1), while a matrix in $\mathbb{R}^{m \times n}$ is a table with its mn entries arranged in m rows and n columns (Definition 2.1).

The table view leads to subspaces of $\mathbb{R}^{m \times n}$ that would not make intuitive sense in \mathbb{R}^{mn} . For the examples, we consider $\mathbb{R}^{2 \times 2}$.

Our first subspace is the set of symmetric matrices according to Definition 2.3 (v), the ones of the form

$$\begin{bmatrix} a & b \\ b & d \end{bmatrix}.$$

To see that this is a subspace, it suffices to observe that the sum of two symmetric matrices is symmetric, and that scaling a symmetric matrix keeps it symmetric. The second and

slightly more creative subspace consists of the matrices of *trace* 0, the ones of the form

$$\begin{bmatrix} a & b \\ c & d \end{bmatrix}, \text{ where } a + d = 0.$$

Generally, the trace of a square matrix is the sum of the diagonal entries. Again, it is easy to see that the subspace axioms are satisfied. However, matrices of trace 1 do not form a subspace, and there are many other non-subspaces, for example the invertible matrices, the ones of rank 1, etc. For the three sets of matrices just mentioned, the “non-subspace” property is easy to see from Lemma 4.9: the zero matrix neither has trace 1, nor is it invertible, nor has it rank 1. A non-subspace containing the zero matrix is the set of all matrices with only nonnegative entries. Find a violation of the subspace axioms for this!

4.2 Bases and dimension

The dimension of a vector space is an important measure of its complexity. So far, we only have an intuitive understanding of dimension, according to which \mathbb{R}^m has dimension m . In this section, we define bases of vector spaces and prove via the Steinitz exchange lemma that every vector space has a basis, and that all bases have the same size. This allows us to define the dimension of a vector space as the size of an arbitrary basis of it. We introduce linear transformations between vector spaces and show how they can be used to formally define when two vector spaces are isomorphic. Maybe surprisingly, all (real) vector spaces of the same dimension are isomorphic. A basis is also useful as a minimal description of the vector space; computing a vector space means to compute a basis of it.

According to our intuition, \mathbb{R}^m should have dimension m . But if V is some other vector space, we may not have such an intuition, so we need to *define* the dimension of a vector space. We expect this definition to tell us that \mathbb{R}^m indeed has dimension m .

What, for example, is the dimension of the vector space of polynomials introduced in Section 4.1.1)? As it “contains” \mathbb{R}^3 (in form of the quadratic polynomials, see page 133), as well as \mathbb{R}^4 (in form of the cubic polynomials), \mathbb{R}^5 , and so on, we expect the dimension to be infinite.

4.2.1 Linear combinations and related concepts in vector spaces

The crucial concept that we develop below is that of a *basis*. A basis of a vector space V consists of linearly independent vectors whose span is V . For that, we need the concept of linear independence and span in an abstract vector space.

In \mathbb{R}^m , we have first defined linear combinations of vectors (Definition 1.4). Based on this fundamental concept, we have introduced linear (in)dependence and the span, proving various results around them (Section 1.3). Formally, we would have to redo

all this for an abstract vector space V —after convincing ourselves that it can be done in the same way—in order to make use of linear combinations and related notions also in other vector spaces. However, we promised that we will stop proving the “obvious” after Fact 4.10; mechanically redoing \mathbb{R}^m theory for abstract vector spaces is part of the “obvious”, so we skip it.

But there *is* something new for abstract vector spaces. Previously, we have worked with *sequences* of vectors in \mathbb{R}^m . This was important in order to handle for example the sequence of columns of a matrix that may contain duplicates. For an abstract vector space V , we can mechanically redo this, but in addition, we will also work with *sets* of vectors. The reason is that we may have to argue with infinite sets of vectors once we move away from \mathbb{R}^m , and such sets are cumbersome (or even impossible) to describe with sequences. Here is an example where infinite sets of vectors are needed.

It is easy to find a basis of \mathbb{R}^m : simply take the m standard unit vectors $\mathbf{e}_1, \mathbf{e}_2, \dots, \mathbf{e}_m$. But in $\mathbb{R}[x]$, the vector space of polynomials, finitely many polynomials $\mathbf{p}_1, \mathbf{p}_2, \dots, \mathbf{p}_n$ cannot form a basis: if d_i is the degree of \mathbf{p}_i , then x^d with $d = \max(d_1, d_2, \dots, d_n)$ is the highest power of x occurring in $\mathbf{p}_1, \mathbf{p}_2, \dots, \mathbf{p}_n$. But then the polynomial $x^{d+1} \in \mathbb{R}[x]$ is not a linear combination of $\mathbf{p}_1, \mathbf{p}_2, \dots, \mathbf{p}_n$, so these polynomials do not even span $\mathbb{R}[x]$.

In order to deal with such situations, we define linear combination, linear (in)dependence and span for a (possibly infinite) set of vectors.

Definition 4.15 (Linear combination of a set of vectors). *Let V be a vector space, $G \subseteq V$ a (possibly infinite) subset of vectors. A linear combination of G is a sum of the form*

$$\sum_{j=1}^n \lambda_j \mathbf{v}_j,$$

where $F = \{\mathbf{v}_1, \mathbf{v}_2, \dots, \mathbf{v}_n\}$ is a finite subset of G .

Hence, a linear combination of G is obtained by selecting *finitely many* elements of G and taking their “normal” linear combination according to Definition 1.4. If G itself is finite, then this is a normal linear combination of the elements of G .

There are two things to note here. First, there is no defined order of elements in a set, so the sequence $\mathbf{v}_1, \mathbf{v}_2, \dots, \mathbf{v}_n$ lists the elements of F in some arbitrary order. We have to make sure that this order does not matter. This is the case: since vector addition in a vector space is commutative by Definition 4.1 (i), the result of the linear combination does not depend on how we order the vectors.

Second, the sequences $\mathbf{v}_1, \mathbf{v}_2, \dots, \mathbf{v}_n$ arising from Definition 4.15 do not contain any duplicates, since a set contains each of its elements only once.

Here is an important fact. In \mathbb{R}^m , we never even thought about it. You could again call it “obvious”, but since it is about the essence of a vector space (being a natural habitat for vectors), we still want to prove it.

Lemma 4.16 (A vector space is closed under linear combinations). *Let V be a vector space. Every linear combination of V is again in V .*

This is quite intuitive: the codomains of the functions $+$ and \cdot in Definition 4.1 are V , meaning that the two “natural vector behaviors”, namely vector addition ($\mathbf{v} + \mathbf{w}$) and scalar multiplication ($\lambda \mathbf{v}$), do not take us out of the vector space. And linear combinations simply combine these two natural behaviors.

Proof. Let $\sum_{j=1}^n \lambda_j \mathbf{v}_j$ be a linear combination of V . Using that V is a vector space (Definition 4.1), we get $\mathbf{w}_j := \lambda_j \mathbf{v}_j \in V$ for all j , by definition of the scalar multiplication ($\cdot : \mathbb{R} \times V \rightarrow V$). By definition of vector addition ($+$: $V \times V \rightarrow V$), we also have $\mathbf{w}_1 + \mathbf{w}_2 \in V$. Applying this again yields $(\mathbf{w}_1 + \mathbf{w}_2) + \mathbf{w}_3 \in V$, and so on, until we get the desired conclusion $\mathbf{w}_1 + \mathbf{w}_2 + \cdots + \mathbf{w}_n \in V$. (Under the hood, this is a proof by induction, and it uses the “obvious” fact that brackets can be omitted in writing down a sum of vectors). \square

You may wonder why we do not allow infinite linear combinations. Infinite sums in themselves can be defined, you may for example know the formula

$$\sum_{j=0}^{\infty} x^j = \frac{1}{1-x} \quad \text{for } x \in \mathbb{R}, |x| < 1.$$

The problem is that Lemma 4.16 may fail for infinite linear combinations, and we do not want this. Consider the vector space of polynomials $\mathbb{R}[x]$, and the infinite “linear combination”

$$\sum_{j=0}^{\infty} x^j$$

of the *unit monomials* $1, x, x^2, \dots$. This is *not* a polynomial: according to Definition 4.3, a polynomial contains only finitely many different monomials (powers of x).

Therefore, the vector space axioms only imply that *finite* linear combinations of vectors are again vectors, but infinite ones may be undefined or take us out of the vector space, as we have just seen.

We conclude with the definitions of linear (in)dependence and span of a set of vectors, based on linear combinations. This is analogous to what we have done in Definitions 1.21 and 1.25 for sequences of vectors.

Definition 4.17 (Linear independence and span of a set of vectors). *Let V be a vector space, $G \subseteq V$ a (possibly infinite) subset of vectors.*

The set G is called linearly dependent if there is an element $\mathbf{v} \in G$ such that \mathbf{v} is a linear combination of $G \setminus \{\mathbf{v}\}$. Otherwise, G is called linearly independent.

The span of G , written as $\text{Span}(G)$, is the set of all linear combinations of G .

4.2.2 Bases

With linear independence and span as in Definition 4.17, we can now formally define a basis of a vector space.

Definition 4.18 (Basis). Let V be a vector space. A subset $B \subseteq V$ is called a basis of V if B is linearly independent and $\text{Span}(B) = V$.

Examples. The set $\{\mathbf{e}_1, \mathbf{e}_2, \dots, \mathbf{e}_m\}$ of standard unit vectors (Section 1.2.2) is the *standard basis* of \mathbb{R}^m . For example, if $m = 2$, then

$$\mathbf{e}_1 = \begin{pmatrix} 1 \\ 0 \end{pmatrix}, \mathbf{e}_2 = \begin{pmatrix} 0 \\ 1 \end{pmatrix}.$$

These two vectors are linearly independent and span \mathbb{R}^2 : for every vector

$$\mathbf{v} = \begin{pmatrix} v_1 \\ v_2 \end{pmatrix} \in \mathbb{R}^2,$$

we have $\mathbf{v} = v_1\mathbf{e}_1 + v_2\mathbf{e}_2$. For general m , the standard unit vectors are seen to be linearly independent by the *private nonzero* argument: every standard unit vector has a nonzero entry (a 1-entry, actually) at a coordinate where all other standard unit vectors have 0-entries. We call such an entry a *private nonzero*. A vector with a private nonzero cannot be a linear combination of the other vectors, and if every vector has a private nonzero, the vectors are linearly independent.

Lemma 4.19. Let A be an $m \times n$ matrix. The set of independent columns of A (Definition 2.10) is a basis of the column space $\mathbf{C}(A)$.

Proof. $\mathbf{C}(A)$ is a subspace by Lemma 4.11 and thus also a vector space by Lemma 4.14. The independent columns are in the column space and linearly independent: by definition, no independent column is a linear combination of the previous columns, and this means that the independent columns are in fact linearly independent; see Corollary 1.23 (iii). Furthermore, the independent columns span the column space, as we have shown in Lemma 2.11. \square

For the subspace of symmetric 2×2 matrices

$$\begin{bmatrix} a & b \\ b & d \end{bmatrix},$$

the following set of three symmetric matrices is a basis:

$$\left\{ \begin{bmatrix} 1 & 0 \\ 0 & 0 \end{bmatrix}, \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix}, \begin{bmatrix} 0 & 0 \\ 0 & 1 \end{bmatrix} \right\}.$$

They are linearly independent: every matrix has at least one private nonzero, a 1-entry where all other matrices have 0-entries. It remains to observe that every symmetric matrix is a linear combination of these three matrices. Indeed, we have

$$\begin{bmatrix} a & b \\ b & d \end{bmatrix} = a \begin{bmatrix} 1 & 0 \\ 0 & 0 \end{bmatrix} + b \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix} + d \begin{bmatrix} 0 & 0 \\ 0 & 1 \end{bmatrix}.$$

For the trace-0 matrices

$$\begin{bmatrix} a & b \\ c & d \end{bmatrix}, \text{ where } a + d = 0,$$

a basis is

$$\left\{ \begin{bmatrix} 1 & 0 \\ 0 & -1 \end{bmatrix}, \begin{bmatrix} 0 & 1 \\ 0 & 0 \end{bmatrix}, \begin{bmatrix} 0 & 0 \\ 1 & 0 \end{bmatrix} \right\}.$$

Linear independence is again easy due to private nonzeros, and as $d = -a$ in a trace-0 matrix, we can obtain every trace-0 matrix as a linear combination:

$$\begin{bmatrix} a & b \\ c & -a \end{bmatrix} = a \begin{bmatrix} 1 & 0 \\ 0 & -1 \end{bmatrix} + b \begin{bmatrix} 0 & 1 \\ 0 & 0 \end{bmatrix} + c \begin{bmatrix} 0 & 0 \\ 1 & 0 \end{bmatrix}.$$

For the vector space $\mathbb{R}[x]$ of polynomials, the infinite set of unit monomials

$$\{x^i : i \in \mathbb{N}\}$$

is a basis. By Definitions 4.3 (of a polynomial) and 4.15 (of a linear combination), every polynomial is a linear combination of unit monomials (recall that $x^0 = 1$). It remains to argue that the unit monomials are linearly independent. Indeed, every unit monomial x^i has its “private nonzero”, the i -th power of x and can therefore not be obtained as a linear combination of other monomials. Here, we really use Definition 4.17 of linear (in)dependence and span for an infinite set.

The subspace of polynomials without a constant term has

$$\{x^i : i \in \mathbb{N}, i > 0\}$$

as a basis, and for the subspace of quadratic polynomials, a basis is

$$\{1, x, x^2\}.$$

Finally, what is the basis of $\{0\}$, the smallest possible subspace of a given vector space? It is the empty set. Indeed, this is linearly independent by Definition 4.17: in the empty set, no vector is a linear combination of the others. And $\text{Span}(\emptyset) = \{0\}$, since an empty sum yields 0; see the discussion in Section 1.1.5.

There are typically many bases. The above examples should not trick us into believing that there is always only one basis of a vector space. For example, $\{e_1, e_2, \dots, e_m\}$ is the *canonical basis* of \mathbb{R}^m , but there are many other choices.

Observation 4.20. Every set $B = \{v_1, v_2, \dots, v_m\} \subseteq \mathbb{R}^m$ of m linearly independent vectors is a basis of \mathbb{R}^m .

Proof. B is linearly independent, and then $\text{Span}(B) = \mathbb{R}^m$ is true by Lemma 1.28. \square

As a second example, let us consider the column space $C(A)$ of a matrix A . In Section 2.3.5, we have asked you to believe that the independent columns of

$$A = \begin{bmatrix} 1 & 2 & 0 & 3 \\ 2 & 4 & 1 & 4 \\ 3 & 6 & 2 & 5 \end{bmatrix}$$

are the first and the third column, so they form a basis of $C(A)$ by Lemma 4.19. But we could also have defined the “backwards independent” columns, by going through the columns of A from *right to left*, and selecting a column if it is not a linear combination of the ones *succeeding* it. This also results in a basis of $C(A)$, by the same arguments as for the “forward independent” columns. If we do this in our example, we end up with the fourth and the third column as a basis.

More generally, we could go through the columns in *any* order and pick up the ones that are not linear combinations of the ones previously considered. This potentially gives us many different bases of $C(A)$.

However, what we find in both \mathbb{R}^m and $C(A)$ is that the alternative bases still have the *same number* of vectors: m in case of \mathbb{R}^m and 2 in case of the column space example. In Section 4.2.3, we prove that this is not a coincidence but true for every vector space.

Existence of a basis. In all examples so far, we have been able to find bases, but is it actually true that every vector space has a basis? The answer is yes, but we will only prove this for *finitely generated* vector spaces, see the next Definition 4.21. The general proof involves some machinery that is standard but beyond the scope of these notes. The Wikipedia article about bases of vector spaces is a good entry point for further reading.²

Definition 4.21 (Finitely generated vector space). *A vector space V is called finitely generated if there exists a finite subset $G \subseteq V$ with $\text{Span}(G) = V$.*

For example, \mathbb{R}^m is finitely generated (by $G = \{\mathbf{e}_1, \mathbf{e}_2, \dots, \mathbf{e}_m\}$) but $\mathbb{R}[x]$, the vector space of polynomials, is not, as we have argued on page 135.

Theorem 4.22. *Let V be a finitely generated vector space, and let $G \subseteq V$ be a finite subset with $\text{Span}(G) = V$. Then V has a basis $B \subseteq G$.*

Proof. This is what we call an “algorithmic proof”. It constructs B by an algorithm. Here is how it goes.

If G is linearly independent, then $B := G$ is a basis by Definition 4.18, so we can stop. “line 1” Otherwise, some vector $\mathbf{v} \in G$ is a linear combination of the other ones (Definition 4.17), so Corollary 1.27 gives $\text{Span}(G \setminus \{\mathbf{v}\}) = \text{Span}(G) = V$. The corollary is about sequences, not sets, but since G is a finite set, we can apply the corollary to any ordering of G . Now we replace G with $G \setminus \{\mathbf{v}\}$ (which still spans V) and go back to line 1. Because the set G gets smaller in every step, the algorithm must eventually stop with a basis in line 1. \square

²[https://en.wikipedia.org/wiki/Basis_\(linear_algebra\)](https://en.wikipedia.org/wiki/Basis_(linear_algebra)), accessed September 2, 2025

A formal correctness proof would go via a precise formulation of the algorithm, either as a loop (correctness proof with *loop invariants*), or a *recursive algorithm* (correctness proof by induction). The latter approach is very close to directly proving the theorem by induction on $g = |G|$ (this is a good exercise). Our proof above, while hopefully clear and easy to understand, is of a somewhat informal “and-so-on” nature, but knowing how we could make it formal if necessary, this is acceptable.

We also see why we need “finitely generated” here. Starting the algorithmic proof above from some infinite set G , we can still remove one element from G in each step, as long as G is linearly dependent, but we cannot argue that this ever stops: after removing an element from an infinite set, we still have an infinite set, so we are not making progress.

4.2.3 The Steinitz exchange lemma

The *Steinitz exchange lemma* is a cornerstone result in the theory of vector spaces. Let V be a finitely generated vector space, let $F \subseteq V$ be a finite set of linearly independent vectors, and $G \subseteq V$ a finite set of vectors with $\text{Span}(G) = V$. The lemma makes two statements. The first one is $|F| \leq |G|$. This makes sense: In \mathbb{R}^2 , for example, we can have *at most* 2 linearly independent vectors, and it takes *at least* 2 vectors to span \mathbb{R}^2 .

The second statement sounds a bit technical at first: we can enlarge F by some elements from G such that the enlarged set has at most the size of G and also spans V . But despite the technical language, this statement is immensely useful.

The name of the lemma comes from the fact that we can think of it as “exchanging elements between G and F ”.

Lemma 4.23 (Steinitz exchange lemma). *Let V be a finitely generated vector space, $F \subseteq V$ a finite set of linearly independent vectors, and $G \subseteq V$ a finite set of vectors with $\text{Span}(G) = V$. Then the following two statements hold.*

(i) $|F| \leq |G|$.

(ii) *There exists a subset $E \subseteq G$ of size $|G| - |F|$ such that $\text{Span}(F \cup E) = V$.*

Note that the set E in (ii) may contain elements of F . In this case, $|F \cup E| < |G|$.

We have in fact already seen a special case of the Steinitz exchange lemma in the form of Lemma 1.28 (m linearly independent vectors $\mathbf{v}_1, \mathbf{v}_2, \dots, \mathbf{v}_m \in \mathbb{R}^m$ span \mathbb{R}^m). To derive this from Lemma 4.23, we use $V = \mathbb{R}^m$, $F = \{\mathbf{v}_1, \mathbf{v}_2, \dots, \mathbf{v}_m\}$ and $G = \{\mathbf{e}_1, \mathbf{e}_2, \dots, \mathbf{e}_m\}$ (the set of m standard unit vectors spans \mathbb{R}^m). Since we already have $|F| = |G|$, we get $|E| = 0$, so E must be the empty set, and $\text{Span}(F) = \text{Span}(F \cup E) = \mathbb{R}^m$.

Proof. Although the Steinitz exchange lemma is more general than Lemma 1.28, we can use (almost) the same proof. We start with the set G and in each step replace one of its elements with an element from F , without changing the span. Once all elements of F have been “swapped in”, we are done (with E being the elements of G that remain). We actually copy the proof of Lemma 1.28 to the extent possible. Here it comes:

Suppose that $F = \{\mathbf{v}_1, \mathbf{v}_2, \dots, \mathbf{v}_m\}$ and $G = \{\mathbf{w}_1, \mathbf{w}_2, \dots, \mathbf{w}_n\}$, so we list the elements of F and G in some arbitrary order. Now we consider the sequence

$$\mathbf{v}_1, \mathbf{w}_1, \mathbf{w}_2, \dots, \mathbf{w}_n,$$

obtained by adding \mathbf{v}_1 before the vectors from G . Since $\text{Span}(\mathbf{w}_1, \mathbf{w}_2, \dots, \mathbf{w}_n) = V$, we know that $\mathbf{v}_1 \in V$ is a linear combination of $\mathbf{w}_1, \mathbf{w}_2, \dots, \mathbf{w}_n$, and hence the $n + 1$ vectors are linearly dependent by Definition 1.21. Then we also know that one of the vectors is a linear combination of the *previous* ones in the sequence, see Lemma 1.22 (iii). This vector cannot be \mathbf{v}_1 because \mathbf{v}_1 also starts the linearly independent sequence $\mathbf{v}_1, \mathbf{v}_2, \dots, \mathbf{v}_m$ in which no vector is a linear combination of the previous ones by Corollary 1.23 (iii). Therefore, one of the \mathbf{w}_i 's is a linear combination of the previous vectors. Removing that vector does not change the span (see Corollary 1.27), so we get a sequence $\mathbf{v}_1, \mathbf{u}_2, \mathbf{u}_3, \dots, \mathbf{u}_n$ (the \mathbf{u}_i 's name the $n - 1$ remaining vectors from G) with

$$\text{Span}(\mathbf{v}_1, \mathbf{u}_2, \mathbf{u}_3, \dots, \mathbf{u}_n) = V.$$

So we have successfully replaced one of the vectors of G with \mathbf{v}_1 , without changing the span. Now we do the same with \mathbf{v}_2 : we add \mathbf{v}_2 directly after \mathbf{v}_1 and argue as before that the $n + 1$ vectors

$$\mathbf{v}_1, \mathbf{v}_2, \mathbf{u}_2, \mathbf{u}_3, \dots, \mathbf{u}_n$$

must be linearly dependent; therefore one of them is a linear combination of the previous ones. This can neither be \mathbf{v}_1 nor \mathbf{v}_2 , because $\mathbf{v}_1, \mathbf{v}_2$ also start a sequence of linearly independent vectors. Hence, some \mathbf{u}_i is a linear combination of the previous vectors and can be removed without changing the span. We call the $n - 2$ remaining vectors from G $\mathbf{u}_3, \mathbf{u}_4, \dots, \mathbf{u}_n$ (a slight abuse of notation, as this redefines the \mathbf{u}_i 's) and get

$$\text{Span}(\mathbf{v}_1, \mathbf{v}_2, \mathbf{u}_3, \mathbf{u}_4, \dots, \mathbf{u}_n) = V.$$

By now, the pattern should be clear. After m replacement steps, we have $n - m$ remaining vectors $\mathbf{u}_{m+1}, \mathbf{u}_{m+2}, \dots, \mathbf{u}_n$ from G and get our desired statement (ii):

$$\text{Span}\left(\underbrace{\mathbf{v}_1, \mathbf{v}_2, \dots, \mathbf{v}_m}_F, \underbrace{\mathbf{u}_{m+1}, \mathbf{u}_{m+2}, \dots, \mathbf{u}_n}_{E, \text{ with } |E|=n-m=|G|-|F|}\right) = V.$$

There is one caveat: this argument only works if $m \leq n$ which is what we still need to prove as statement (i) of the lemma. But the argument itself proves it: whenever we add the next element of F to our current sequence, there must still be some element of G left, because we are guaranteed to find an element of G that we can remove. Therefore, $|F| \leq |G|$, indeed. \square

The Steinitz exchange lemma has an important corollary. Because of its importance, we also call it a theorem. It confirms what we have observed in examples before: even if a vector space has different bases, all of them have the same number of vectors.

Theorem 4.24 (All bases have the same size). *Let V be a finitely generated vector space and let $B, B' \subseteq V$ be two bases of V . Then $|B| = |B'|$.*

Proof. B and B' are linearly independent, and $\text{Span}(B) = \text{Span}(B') = V$ (Definition 4.18). Applying statement (i) of the Steinitz exchange Lemma 4.23 with $F = B, G = B'$ yields $|B| \leq |B'|$; with $F = B', G = B$, we get $|B'| \leq |B|$. \square

There are vector spaces that are not finitely generated, such as the vector space $\mathbb{R}[x]$ of polynomials defined in Section 4.1.1. While Theorem 4.24 does not apply to such vector spaces, it can be generalized to the infinite case where $|B| = |B'|$ then means “the same kind of infinity.”

4.2.4 Dimension

Now we can define the dimension of a vector space, at least if it is finitely generated.

Definition 4.25 (Dimension). *Let V be a finitely generated vector space. Then $\dim(V)$, the dimension of V , is the size of an arbitrary basis B of V .*

This definition uses that every finitely generated vector space has a basis to begin with (Theorem 4.22), and that all bases have the same size (Theorem 4.24).

From the examples of bases in Section 4.2.2, you can therefore immediately deduce the dimensions of the corresponding vector spaces. As expected, $\dim(\mathbb{R}^m) = m$.

Figure 4.2 shows three subspaces of \mathbb{R}^3 , of dimensions 0 (point), 1 (line), and 2 (plane).

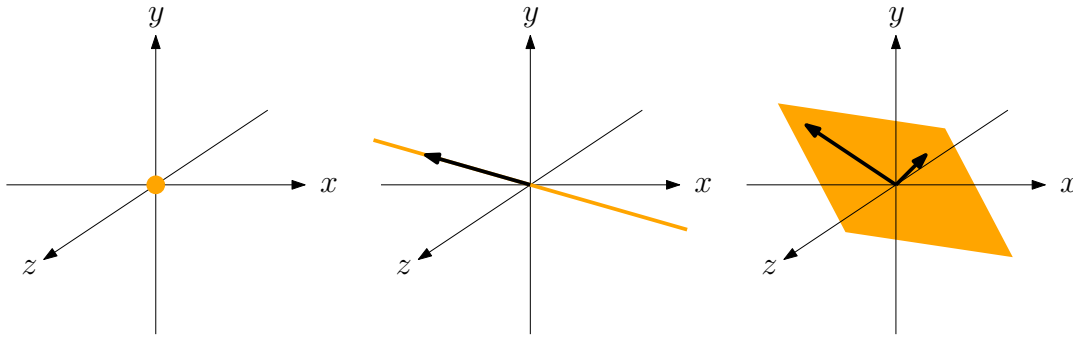


Figure 4.2: Three subspaces of \mathbb{R}^3 : The unique subspace of dimension 0, the point at the origin with an empty basis (left); a subspace of dimension 1, a line through the origin with a basis of size 1 (middle); a subspace of dimension 2, a plane through the origin with a basis of size 2. (right)

In many cases, the dimension of a vector space directly corresponds to the *degrees of freedom* in its definition, informally defined as the number of values that can vary independently. For example, \mathbb{R}^m has m degrees of freedom, since all m coordinates of a vector

can vary independently within \mathbb{R}^m . The vector space of symmetric 2×2 matrices

$$\begin{bmatrix} a & b \\ b & d \end{bmatrix}$$

has 3 degrees of freedom, the values a, b, d . Consequently, its dimension is 3. For the 2×2 matrices of trace 0,

$$\begin{bmatrix} a & b \\ c & d \end{bmatrix}, \text{ where } a + d = 0,$$

we have 4 defining values a, b, c, d , but writing a trace-0 matrix as

$$\begin{bmatrix} a & b \\ c & -a \end{bmatrix},$$

we saw that there are also only 3 degrees of freedom, the values a, b, c . So the dimension of this vector space is 3 as well.

But there are vector spaces without such obvious ways of identifying degrees of freedom. For example, the column space $C(A)$ of a matrix A has dimension $r = \text{rank}(A)$ (because the r independent columns form a basis; see Lemma 4.19). But it is not clear from the definition of $C(A)$ whether there are any r values that can vary independently. Theorem 4.29 and the discussion after this show the existence of such degrees of freedom for every vector space, even if they are initially hidden.

4.2.5 Linear transformations between vector spaces

In Section 2.2.2, we have defined linear transformations as functions $T : \mathbb{R}^n \rightarrow \mathbb{R}^m$ satisfying linearity: $T(\lambda_1 \mathbf{x}_1 + \lambda_2 \mathbf{x}_2) = \lambda_1 T(\mathbf{x}_1) + \lambda_2 T(\mathbf{x}_2)$ for all $\mathbf{x}_1, \mathbf{x}_2 \in \mathbb{R}^n$ and all $\lambda_1, \lambda_2 \in \mathbb{R}$.

However, the right level of abstraction is to define linear transformations as functions between two vector spaces V and W . Previously, we have only used $V = \mathbb{R}^n, W = \mathbb{R}^m$.

Bijjective (undoable) linear transformations between vector spaces are particularly interesting. A bijective linear transformation between V and W can be used to compute a basis of one of the spaces from a basis of the other one (see Lemma 4.27 below). We will make use of this in computing a basis of the nullspace of a matrix in Section 4.3.3.

Definition 4.26 (Linear transformation between vector spaces). *Let V, W be two vector spaces. A function $T : V \rightarrow W$ is called a linear transformation between vector spaces if the following linearity axiom holds for all $\mathbf{x}_1, \mathbf{x}_2 \in V$ and all $\lambda_1, \lambda_2 \in \mathbb{R}$.*

$$T(\lambda_1 \mathbf{x}_1 + \lambda_2 \mathbf{x}_2) = \lambda_1 T(\mathbf{x}_1) + \lambda_2 T(\mathbf{x}_2).$$

In order for linearity to even make sense, we need $\lambda_1 \mathbf{x}_1 + \lambda_2 \mathbf{x}_2 \in V$ (so that this vector is a legal input for T), and $\lambda_1 T(\mathbf{x}_1) + \lambda_2 T(\mathbf{x}_2) \in W$ (so that this vector is a legal output of T). For $V = \mathbb{R}^n$ and $W = \mathbb{R}^m$, we did not have to think about this at all, and here, we have to think about it only for a second: by Definition 4.1, adding up and scaling vectors

does not take us out of the vector space, so everything is fine. Or, how Lemma 4.16 puts it: a vector space is closed under linear combinations.

Linear transformations and linear functionals that we have introduced separately in Definition 2.21 are now both special cases of linear transformations between vector spaces. For a linear functional, we use $W = \mathbb{R}$. Indeed, with its normal “+” and “·”, the set of real numbers is “obviously” a vector space according to Definition 4.1.

Here, we are particularly interested in bijective (undoable) linear transformations between vector spaces. The key lemma is the following (we refer to Definition 2.48 for the term *bijective*, but we also explain it again during the proof).

Lemma 4.27 (Bijective linear transformations preserve bases). *Let $T : V \rightarrow W$ be a bijective linear transformation between vector spaces V and W . Let $B = \{\mathbf{v}_1, \mathbf{v}_2, \dots, \mathbf{v}_\ell\} \subseteq V$ be a finite set of some size ℓ , and $T(B) = \{T(\mathbf{v}_1), T(\mathbf{v}_2), \dots, T(\mathbf{v}_\ell)\} \subseteq W$ the transformed set. Then $|T(B)| = |B|$. Moreover, B is a basis of V if and only if $T(B)$ is a basis of W . We therefore also have $\dim(V) = \dim(W)$.*

In words, bijective linear transformations between vector spaces “preserve” bases and dimension (but the lemma only proves this in the finitely generated case). If there is a bijective linear transformation between V and W , the two spaces are *isomorphic* (“of the same form”), see Definition 4.28 below.

Proof of Lemma 4.27. The equation $|T(B)| = |B|$ might seem obvious, but the set notation is a bit deceptive. If different inputs $\mathbf{v}_i, \mathbf{v}_j$ would lead to the same output $T(\mathbf{v}_i) = T(\mathbf{v}_j)$, we would get $|T(B)| < |B|$, because $T(B)$ as a set contains this output only once. However, this situation cannot occur: T being bijective means that for every output, exactly one input leads to it. Said in another way: different inputs lead to different outputs, so the ℓ inputs in B also lead to ℓ outputs in $T(B)$.

Next we prove the “if” direction: if B is a basis of V , then $T(B)$ is a basis of W . The other (“only if”) direction follows by applying the “if” direction to the inverse T^{-1} which is a bijective linear transformation $T^{-1} : W \rightarrow V$ (we use the “obvious” generalization of Lemma 2.52 to $T : V \rightarrow W$). For T^{-1} , the “if” direction says that if $T(B)$ is a basis of W , then $T^{-1}(T(B)) = B$ is a basis of V . This is our desired “only if” direction. Here we use that $T^{-1} \circ T = \text{id}$ (T^{-1} is undoing T ; see Fact 2.49).

It remains to prove the “if” direction: if B is a basis of V , then $T(B)$ is a basis of W . For this, we use the generalization of linearity that we have proved in Lemma 2.25 for the special cases $V = \mathbb{R}^n$ and $W = \mathbb{R}^m$ or $W = \mathbb{R}$, but it “obviously” holds for abstract vector spaces V, W as well: for all $\mathbf{x}_1, \mathbf{x}_2, \dots, \mathbf{x}_\ell \in V$ and all $\lambda_1, \lambda_2, \dots, \lambda_\ell \in \mathbb{R}$, we have

$$T\left(\sum_{j=1}^{\ell} \lambda_j \mathbf{x}_j\right) = \sum_{j=1}^{\ell} \lambda_j T(\mathbf{x}_j). \quad (4.2)$$

Now for the actual proof. If B is a basis, we have to show according to Definition 4.18 that $T(B)$ is (a) linearly independent, and (b) $\text{Span}(T(B)) = W$.

(a): We show linear independence of $T(B)$ according to Corollary 1.23 (ii): the zero vector can be written as a linear combination

$$\sum_{j=1}^{\ell} \lambda_j T(\mathbf{v}_j) = \mathbf{0}$$

of $T(B)$ only in the trivial way, meaning $\lambda_j = 0$ for all j . To see this, we apply (4.2) to such a linear combination and get

$$\mathbf{0} = \sum_{j=1}^{\ell} \lambda_j T(\mathbf{v}_j) = T\left(\sum_{j=1}^{\ell} \lambda_j \mathbf{v}_j\right).$$

T is bijective and therefore injective (for every possible output, at most one input leads to it). Output $\mathbf{0}$ can therefore only come from input $\mathbf{0}$ (use the “obvious” generalization of Lemma 2.24 to linear transformations between vector spaces); hence

$$\sum_{j=1}^{\ell} \lambda_j \mathbf{v}_j = \mathbf{0}. \quad (4.3)$$

Since B is a basis of V and therefore linearly independent, (4.3) implies $\lambda_j = 0$ for all j , see Corollary 1.23 (ii).

(b): We show $\text{Span}(T(B)) = W$ by writing every vector $\mathbf{w} \in W$ as a linear combination of $T(B)$. Since T is bijective and therefore surjective (for every possible output, at least one input leads to it), there is a vector $\mathbf{v} \in V$ such that $T(\mathbf{v}) = \mathbf{w}$. Because B is a basis of V , we have $\text{Span}(B) = V$, so \mathbf{v} is a linear combination of B ,

$$\mathbf{v} = \sum_{j=1}^{\ell} \lambda_j \mathbf{v}_j.$$

Applying T to both sides gives

$$\mathbf{w} = T(\mathbf{v}) = T\left(\sum_{j=1}^{\ell} \lambda_j \mathbf{v}_j\right) \stackrel{(4.2)}{=} \sum_{j=1}^{\ell} \lambda_j T(\mathbf{v}_j),$$

writing \mathbf{w} as a linear combination of $T(B)$. □

Having a bijective transformation $T : V \rightarrow W$, means that from “knowing” V (having a basis of it), we also “know” W , and vice versa (we talk about this kind of “knowledge” in more detail in Section 4.2.6 below). Moreover, the two spaces have the same dimensions. In this case, we consider V and W as “essentially the same” (on the very abstract level of *category theory* that we will not further go into here). The mathematical term for this is that V and W are isomorphic.

Definition 4.28 (Isomorphic vector spaces, isomorphism). *Let V, W be two vector spaces. If there is a bijective linear transformation $T : V \rightarrow W$ (Definition 4.26), then V and W are called isomorphic, and T is called an isomorphism between V and W .*

We have already seen an example of two isomorphic vector spaces on page 133. Let V be the vector space of degree-2 polynomials, and $W = \mathbb{R}^3$. We have observed before that both a degree-2 polynomial as well as a vector in \mathbb{R}^3 are determined by a triple of real numbers, and in both cases, the vector operations (vector addition, scalar multiplications) are the same on the level of these triples. For example, adding the two degree-2 polynomials $3x^2 + 1$ and $2x - 5$ results in the degree-2 polynomial $3x^2 + 2x - 4$. On “triple level”, this is $(1, 0, 3) + (-5, 2, 0) = (-4, 2, 3)$. The corresponding vector addition

$$\begin{pmatrix} 1 \\ 0 \\ 3 \end{pmatrix} + \begin{pmatrix} -5 \\ 2 \\ 0 \end{pmatrix} = \begin{pmatrix} -4 \\ 2 \\ 3 \end{pmatrix}$$

does the same on triple level. Therefore, although defined differently, degree-2 polynomials and vectors in \mathbb{R}^3 are conceptually the same. The notion of an isomorphism captures this. Indeed, here is the canonical isomorphism between degree-2 polynomials (vector space V) and $W = \mathbb{R}^3$:

$$T : \mathbf{p} \mapsto \mathbf{v}_{\mathbf{p}},$$

with \mathbf{v}_p as defined on page 133. In (4.1), we have in fact already said what it takes to check linearity of T according to Definition 4.26.

Another case of isomorphism is $V = \mathbb{R}^{mn}$ and $W = \mathbb{R}^{m \times n}$, see page 133.

All m -dimensional vector spaces are isomorphic. If we consider isomorphic vector spaces as “the same”, then the diversity among vector spaces is much smaller than you might have expected. In fact, any two (real) vector spaces of the same dimension m are isomorphic! We will not give the full proof but the main ingredients, from which it is not hard to work out the full proof. No deep mathematics is needed here.

The main ingredient is the following nice property of a basis. It does not only span the vector space, it even writes every vector as a linear combination in a *unique* way. This is true for all vector spaces, but for simplicity, we only cover the finitely generated case.

Theorem 4.29 (A basis writes each vector as a unique linear combination). *Let V be a finitely generated vector space of dimension m and $B = \{\mathbf{v}_1, \mathbf{v}_2, \dots, \mathbf{v}_m\} \subseteq V$ a basis of V . For every $\mathbf{v} \in V$, there are unique scalars $\lambda_1, \lambda_2, \dots, \lambda_m$ such that*

$$\mathbf{v} = \sum_{j=1}^m \lambda_j \mathbf{v}_j.$$

Proof. Since $\text{Span}(B) = V$ by Definition 4.18 of a basis, we are sure to find *some* such scalars. Now we proceed as in the proof of Lemma 1.24: given two linear combinations

$$\mathbf{v} = \sum_{j=1}^m \lambda_j \mathbf{v}_j = \sum_{j=1}^m \mu_j \mathbf{v}_j,$$

we subtract them, after which $\lambda_j = \mu_j$ for all j follows from linear independence of B according to Corollary 1.23 (ii). \square

From this, we conclude that any two (real) vector spaces of the same dimension m are isomorphic. The reason is the following: take any vector space V of dimension m and fix a basis $B = \{\mathbf{v}_1, \mathbf{v}_2, \dots, \mathbf{v}_m\} \subseteq V$ with an arbitrary ordering of its vectors. Given this, any vector $\mathbf{v} \in V$ can be described by its unique *coordinate vector* $(\lambda_1, \lambda_2, \dots, \lambda_m) \in \mathbb{R}^m$, where the λ_j are the ones guaranteed by Theorem 4.29. One can check that the function

$$T : V \rightarrow \mathbb{R}^m, \mathbf{v} \mapsto \mathbf{v}'\text{'s coordinate vector}$$

is an isomorphism between V and \mathbb{R}^m according to Definition 4.28.

Coming back to the *degrees of freedom* mentioned in Section 4.2.4, we can now declare the m entries of the coordinate vector as the degrees of freedom of V . Indeed, they can vary independently, and each choice uniquely leads to an element of V .

4.2.6 Computing a vector space

All vector spaces and subspaces (except the trivial one, $\{0\}$) that we have seen in this chapter have infinitely many elements. Computing a vector space can therefore not mean to output all its elements. Instead, we want to output something that *describes* all the elements, and this output should ideally be as small as possible.

This is precisely what a basis of a vector space (Definition 4.18) does. Indeed, if V is a vector space with a basis $B \subseteq V$, we know that $V = \text{Span}(B)$, the set of all linear combinations of B . This is the description of V . Moreover, in the finitely generated case (Definition 4.21), B is a smallest description in the sense that no smaller set of vectors can span V . This may be plausible, but we have not proved it, so let us do it now.

Lemma 4.30 (Less than $\dim(V)$ vectors do not span V). *Let V be a finitely generated vector space. Let $G \subseteq V$ be a finite subset of size $|G| < \dim(V)$ (Definition 4.25). Then $\text{Span}(G) \neq V$.*

Proof. We want to prove the implication $|G| < \dim(V) \Rightarrow \text{Span}(G) \neq V$. This is logically equivalent to the *contraposition* $\text{Span}(G) = V \Rightarrow |G| \geq \dim(V)$, so we can instead prove the second implication (recall Theorem 3.7 (ii) \Rightarrow (i) which we have also proved by contraposition). This is now easy: If $\text{Span}(G) = V$, Theorem 4.22 tells us that V has a basis $B \subseteq G$. Since $|B| = \dim(V)$ by Definition 4.25, $|G| \geq \dim(V)$ follows. \square

Therefore, *computing* a finitely generated vector space for us means to find a basis of it. Figure 4.2 visualizes the representation of a space by a basis. Knowing the basis vectors, the space in question is uniquely determined: it consists of all linear combinations of the basis vectors. Vector spaces that are not finitely generated cannot be computed in this way, since we cannot even output the (infinite) basis. But the three fundamental subspaces that we will compute below are all finitely generated, and we will show how to find bases of them and therefore also their dimensions.

4.3 Computing the three fundamental subspaces

An $m \times n$ matrix A defines three fundamental subspaces: Column space, row space, and nullspace. We show how they can be computed by which we mean to find bases for them. Once we have such bases, we also know the dimensions of the fundamental subspaces, and how they relate to each other: If $\text{rank}(A) = r$, then both row and column space have dimension r , while the nullspace has dimension $n - r$. We finally apply these results to compute all solutions of a system of linear equations $A\mathbf{x} = \mathbf{b}$.

For a given $m \times n$ matrix A , we now want to compute the three *fundamental subspaces* $\mathbf{C}(A)$ (column space), $\mathbf{R}(A)$ (row space), and $\mathbf{N}(A)$ (nullspace). If we have bases for them, we also know the dimensions of these spaces, by Definition 4.25.

The key to understanding these subspaces is Gauss-Jordan elimination. In Section 3.3.3, we have shown how this algorithm can transform every matrix A into a unique standard form, a matrix R in reduced row echelon form (RREF). As our running example, we again use the matrix that we have previously considered in Sections 2.3.5 and 3.3.4. Here, Gauss-Jordan elimination leads to the following standard form R (see Page 123):

$$A = \begin{bmatrix} 1 & 2 & 0 & 3 \\ 2 & 4 & 1 & 4 \\ 3 & 6 & 2 & 5 \end{bmatrix} \rightarrow R = \begin{bmatrix} 1 & 2 & 0 & 3 \\ 0 & 0 & 1 & -2 \\ 0 & 0 & 0 & 0 \end{bmatrix}. \quad (4.4)$$

From this, we have already been able to read off the CR decomposition of A , see Theorem 3.18. Now we will see how to read off (bases of) the three fundamental subspaces.

4.3.1 Column space

We recall Definition 2.9. The column space of an $m \times n$ matrix A is the set of all linear combinations of the columns of A ,

$$\mathbf{C}(A) = \{A\mathbf{x} : \mathbf{x} \in \mathbb{R}^n\} \subseteq \mathbb{R}^m.$$

We know from Lemma 4.11 that $\mathbf{C}(A)$ is a subspace of \mathbb{R}^m and therefore also a vector space by Lemma 4.14. For computing it, we have already done all the work. We summarize the situation in the following theorem.

Theorem 4.31. *Let A be an $m \times n$ matrix, and let R in $\text{RREF}(j_1, j_2, \dots, j_r)$ be the result of Gauss-Jordan elimination on A according to Theorem 3.17. Then A has its independent columns at indices j_1, j_2, \dots, j_r , and these columns form a basis of the column space $\mathbf{C}(A)$. In particular,*

$$\dim(\mathbf{C}(A)) = \text{rank}(A) = r.$$

Proof. The independent columns of A form a basis of $\mathbf{C}(A)$ by Lemma 4.19, and R reveals their indices j_1, j_2, \dots, j_r ; see Theorem 3.18. We thus have $\dim(\mathbf{C}(A)) = r$ by Definition 4.25 and $\text{rank}(A) = r$ by Definition 2.10. \square

In the running example (4.4), R is in $\text{RREF}(1, 3)$, so the basis B of $\mathbf{C}(A)$ resulting from the theorem is given by the two independent columns 1 and 3,

$$B = \left\{ \begin{pmatrix} 1 \\ 2 \\ 3 \end{pmatrix}, \begin{pmatrix} 0 \\ 1 \\ 2 \end{pmatrix} \right\}.$$

4.3.2 Row space

We recall Definition 2.14. The row space of an $m \times n$ matrix A is the set of all linear combinations of the rows of A , officially defined as the column space of the transpose:

$$\mathbf{R}(A) = \mathbf{C}(A^\top) \subseteq \mathbb{R}^n.$$

The row space is a subspace of \mathbb{R}^n by Corollary 4.12 and therefore also a vector space by Lemma 4.14.

We could compute $\mathbf{R}(A) = \mathbf{C}(A^\top)$ via Gauss-Jordan elimination on A^\top , according to the previous Theorem 4.31. But Gauss-Jordan elimination on A already provides us with all we need. The upshot is that the first r rows of R (also forming the matrix R' in the CR decomposition $A = CR'$ according to Theorem 3.18) are a basis of the row space.

Theorem 4.32. *Let A be an $m \times n$ matrix, and let R in $\text{RREF}(j_1, j_2, \dots, j_r)$ be the result of Gauss-Jordan elimination on A , according to Theorem 3.17. Then the first r columns of R^\top form a basis of the row space $\mathbf{R}(A)$. In particular,*

$$\dim(\mathbf{R}(A)) = r.$$

Proof. We have $R = MA$ with M invertible (Theorem 3.17), hence R and A have the same row space by invariance Lemma 3.5. From R , a basis of the row space can be read off immediately (for simplicity, we argue with the rows of R although we should officially argue with the columns of R^\top): by RREF (Definition 3.13), R starts with r nonzero rows and ends with $m-r$ zero rows. Hence, the r nonzero rows already span the row space, and they are also linearly independent, as each of them has a private nonzero (in the column where the downward step happens). So the first r rows of R (officially, the first r columns of R^\top) are a basis of $\mathbf{R}(R) = \mathbf{R}(A)$ by Definition 4.18. We then have $\dim(\mathbf{C}(A)) = r$ by Definition 4.25 \square

In our running example (4.4), the basis B of $\mathbf{R}(A)$ resulting from the theorem is

$$B = \left\{ \begin{pmatrix} 1 \\ 2 \\ 0 \\ 3 \end{pmatrix}, \begin{pmatrix} 0 \\ 0 \\ 1 \\ -2 \end{pmatrix} \right\}.$$

The previous two Theorems 4.31 and 4.32 have a very interesting consequence that also deserves to be called a theorem, although it is now simply a corollary. This is the

following result that is sometimes summarized as *row rank equals columns rank*. We have previously proved this in Section 2.1.4 for matrices of rank 1, but now we can show that it holds for all matrices.

Theorem 4.33. *Let A be an $m \times n$ matrix. Then*

$$\text{rank}(A) = \text{rank}(A^\top).$$

Proof. By Theorem 4.31,

$$\text{rank}(A) = \dim(\mathbf{C}(A)). \quad (4.5)$$

In words, the rank of a matrix is the dimension of its column space. Applying this to the transpose A^\top gives

$$\text{rank}(A^\top) = \dim(\mathbf{C}(A^\top)) = \dim(\mathbf{R}(A)), \quad (4.6)$$

using Definition 2.14 of the row space. It remains to observe that both $\dim(\mathbf{C}(A))$ in (4.5) and $\dim(\mathbf{R}(A))$ in (4.6) evaluate to the same number r from Theorems 4.31 and 4.32, the number of downward steps in the reduced row echelon form R of A . \square

Corollary 4.34 (The rank is at most the smaller of the two matrix dimensions). *Let A be an $m \times n$ matrix of rank r . Then $r \leq \min(n, m)$.*

Proof. The rank of a matrix is the number of independent columns. Theorem 4.33 says that $\text{rank}(A) = \text{rank}(A^\top)$. Therefore we have $r \leq n$ (the number of columns of A) and $r \leq m$ (the number of columns of A^\top). \square

4.3.3 Nullspace

We recall Definition 2.17. The nullspace of an $m \times n$ matrix A is the set of all solutions of the system $A\mathbf{x} = \mathbf{0}$,

$$\mathbf{N}(A) = \{\mathbf{x} \in \mathbb{R}^n : A\mathbf{x} = \mathbf{0}\} \subseteq \mathbb{R}^n.$$

By invariance Lemma 3.3, the matrix R resulting from A via Gauss-Jordan elimination has the same nullspace as A , and as for column and row space before, a basis of the nullspace is easy to read off from R . The technical realization of this is a bit more involved than for column and row space, so we look at the running example (4.4) first. Here,

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

with $\mathbf{N}(A) = \mathbf{N}(R)$. We further observe that $\mathbf{N}(R) = \mathbf{N}(R')$ where

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

is obtained from R by removing the zero row at the end. Indeed, the system of equations $R\mathbf{x} = \mathbf{0}$ defining $\mathbf{N}(R)$ is equivalent to $R'\mathbf{x} = \mathbf{0}$ defining $\mathbf{N}(R')$, since the extra equation in $R\mathbf{x} = \mathbf{0}$ is $0^\top \mathbf{x} = 0$. This equation always holds and can therefore be ignored.

Splitting the sum in the matrix-vector product $R'\mathbf{x}$ (Definition 2.4) into two sums, one for $j = 1, 3$, and one for $j = 2, 4$, we can write $R'\mathbf{x} = \mathbf{0}$ as

$$\begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \begin{pmatrix} x_1 \\ x_3 \end{pmatrix} + \begin{bmatrix} 2 & 3 \\ 0 & -2 \end{bmatrix} \begin{pmatrix} x_2 \\ x_4 \end{pmatrix} = \mathbf{0}.$$

The submatrix of R' with the independent columns 1, 3 is the identity matrix. We can therefore directly solve for x_1, x_3 and get

$$\begin{pmatrix} x_1 \\ x_3 \end{pmatrix} = - \begin{bmatrix} 2 & 3 \\ 0 & -2 \end{bmatrix} \begin{pmatrix} x_2 \\ x_4 \end{pmatrix}. \quad (4.7)$$

Now we have a full understanding of all $\mathbf{x} \in \mathbf{N}(R')$: for the values of x_2, x_4 , we can choose arbitrary real numbers, and the values of x_1, x_3 are then determined via (4.7). We also expect $\mathbf{N}(R')$ to have dimension 2, because the nullspace—the solutions of (4.7)—can be described by vectors in \mathbb{R}^2 , concretely the x_2x_4 -plane. In technical terms, $\mathbf{N}(R')$ is isomorphic to \mathbb{R}^2 (Definition 4.28).

Indeed, a basis of $\mathbf{N}(R')$ is obtained in the form of the two *special solutions*, the ones coming from the standard unit vectors $\mathbf{e}_1, \mathbf{e}_2$ in the x_2x_4 -plane:

$\begin{pmatrix} x_2 \\ x_4 \end{pmatrix}$	special solutions
$\begin{pmatrix} x_2 \\ x_4 \end{pmatrix}$	$\begin{pmatrix} 1 \\ 0 \end{pmatrix} \quad \begin{pmatrix} 0 \\ 1 \end{pmatrix}$
$-\begin{bmatrix} 2 & 3 \\ 0 & -2 \end{bmatrix} \begin{pmatrix} x_2 \\ x_4 \end{pmatrix} = \begin{pmatrix} x_1 \\ x_3 \end{pmatrix}$	$\begin{pmatrix} -2 \\ 0 \end{pmatrix} \quad \begin{pmatrix} -3 \\ 2 \end{pmatrix}$

Let us double check on the running example (4.4) that these two special solutions are indeed in the original nullspace $\mathbf{N}(A)$ that we want to compute. For this, we verify that

$$\begin{bmatrix} 1 & 2 & 0 & 3 \\ 2 & 4 & 1 & 4 \\ 3 & 6 & 2 & 5 \end{bmatrix} \begin{pmatrix} -2 \\ 1 \\ 0 \\ 0 \end{pmatrix} = \mathbf{0}, \quad \text{and} \quad \begin{bmatrix} 1 & 2 & 0 & 3 \\ 2 & 4 & 1 & 4 \\ 3 & 6 & 2 & 5 \end{bmatrix} \begin{pmatrix} -3 \\ 0 \\ 2 \\ 1 \end{pmatrix} = \mathbf{0}.$$

But why do these solutions form a basis of $\mathbf{N}(A)$ according to Definition 4.18? This follows from the general construction of a basis of $\mathbf{N}(A)$ that we present next.

We proceed in two steps. First, we prove that $\mathbf{N}(R) = \mathbf{N}(A)$ is isomorphic to \mathbb{R}^{n-r} . Knowing this, we can apply the “basis preservation” Lemma 4.27 to construct a basis of $\mathbf{N}(R)$ from a basis of \mathbb{R}^{n-r} . We conveniently use the standard basis $\{\mathbf{e}_1, \mathbf{e}_2, \dots, \mathbf{e}_{n-r}\}$ of \mathbb{R}^{n-r} . This second step works exactly like in the example above, but now we can use Lemma 4.27 to conclude that the result is indeed a basis of $\mathbf{N}(R) = \mathbf{N}(A)$.

Lemma 4.35 (Nullspace isomorphism). *Let R be an $m \times n$ matrix in $\text{RREF}(j_1, j_2, \dots, j_r)$, and let k_1, k_2, \dots, k_{n-r} be the column indices not in $\{j_1, j_2, \dots, j_r\}$. Then $T : \mathbf{N}(R) \rightarrow \mathbb{R}^{n-r}$,*

$$T : \mathbf{x} \mapsto \begin{pmatrix} x_{k_1} \\ x_{k_2} \\ \vdots \\ x_{k_{n-r}} \end{pmatrix}$$

is an isomorphism (bijective linear transformation) between $\mathbf{N}(R)$ and \mathbb{R}^{n-r} . In particular, we get $\dim(\mathbf{N}(R)) = n - r$ from Lemma 4.27.

In the example on the previous page, we have $k_1 = 2, k_2 = 4$, so in this case T is

$$T : \mathbf{x} \mapsto \begin{pmatrix} x_2 \\ x_4 \end{pmatrix}.$$

We have seen in the example why this is bijective (for every possible output, exactly one input from the nullspace leads to it): given arbitrary output values x_2, x_4 , we get unique values x_1, x_3 through (4.7) and therefore a unique input vector $\mathbf{x} \in \mathbf{N}(R)$ leading to the given values of x_2 and x_4 . It is also easy to check that T is linear (Definition 4.26). T is in fact a *parallel projection* (we saw such a projection in Figure 2.8), but one of the simplest kind: the output results from just omitting some coordinates (x_1 and x_3) of the input. In this case, linearity is an immediate consequence of vector addition (Definition 1.2) and scalar multiplication (Definition 1.3).

Proof of Lemma 4.35. Let R' be the submatrix of R containing the first r rows (the nonzero ones). We have $\mathbf{N}(R) = \mathbf{N}(R')$, because all $m - r$ extra equations in $R\mathbf{x} = \mathbf{0}$ are $\mathbf{0}^\top \mathbf{x} = 0$ and can therefore be ignored, since they always hold.

Because R' is in $\text{RREF}(j_1, j_2, \dots, j_r)$ as well, simply without zero rows at the end, column j_i of R is \mathbf{e}_i , the i -th standard unit vector in \mathbb{R}^r ; see Definition 3.13 (i). The submatrix of R' with columns j_1, j_2, \dots, j_r is therefore the identity matrix, and in analogy to (4.7), we can rewrite $R'\mathbf{x} = \mathbf{0}$ as

$$\begin{pmatrix} x_{j_1} \\ x_{j_2} \\ \vdots \\ x_{j_r} \end{pmatrix} = -Q \begin{pmatrix} x_{k_1} \\ x_{k_2} \\ \vdots \\ x_{k_{n-r}} \end{pmatrix}, \quad (4.8)$$

where Q is the $r \times (n - r)$ submatrix of R' with columns k_1, k_2, \dots, k_{n-r} . This matrix Q is in general not a square matrix, unlike in (4.7).

Now we can show as in the example that T is bijective: for every possible output (already giving some coordinates of the input $\mathbf{x} \in \mathbf{N}(R)$), the remaining coordinates of \mathbf{x} are uniquely determined via (4.8), so there is exactly one input leading to the given output. We omit the easy calculations that prove linearity of T according to Definition 4.26. \square

Finding them is easy: For all $i \in [n - r]$, the condition is $T(\mathbf{v}_i) = \mathbf{e}_i$, so \mathbf{v}_i is the unique input for T leading to output \mathbf{e}_i . We have previously seen how to construct this via (4.8). We therefore have the following result.

A basis of the nullspace $\mathbf{N}(A)$ is given by the $n - r$ vectors $\mathbf{v}_1, \mathbf{v}_2, \dots, \mathbf{v}_{n-r}$, where \mathbf{v}_i is the vector $\mathbf{x} \in \mathbf{N}(R)$ with

$$\begin{pmatrix} x_{k_1} \\ x_{k_2} \\ \vdots \\ x_{k_{n-r}} \end{pmatrix} = \mathbf{e}_i, \quad \begin{pmatrix} x_{j_1} \\ x_{j_2} \\ \vdots \\ x_{j_r} \end{pmatrix} = -Q\mathbf{e}_i.$$

$$\dim(\mathbf{N}(A)) = n - r.$$
$$\begin{array}{ccccccc}
 & & \mathbf{C}(A) & & \mathbf{N}(A) & & \mathbf{R}(A) \\
 & & & & \begin{array}{cc} \overbrace{\begin{pmatrix} 1 \\ 0 \end{pmatrix}}^{x_2, x_4} & \overbrace{\begin{pmatrix} 0 \\ 1 \end{pmatrix}}^{x_2, x_4} \\
 & & & & \begin{array}{cc} \overbrace{\begin{pmatrix} -2 \\ 0 \end{pmatrix}}^{x_1, x_3} & \overbrace{\begin{pmatrix} -3 \\ 2 \end{pmatrix}}^{x_2, x_3} \\
 \text{Bases:} & & \begin{pmatrix} 1 \\ 2 \\ 3 \end{pmatrix} & \begin{pmatrix} 0 \\ 1 \\ 2 \end{pmatrix} & & & \begin{pmatrix} 1 \\ 2 \\ 0 \\ 3 \end{pmatrix} & \begin{pmatrix} 0 \\ 0 \\ 1 \\ -2 \end{pmatrix} \\
 & & \uparrow & \uparrow & & & \uparrow & \uparrow \\
 \underbrace{\begin{bmatrix} 1 & 0 \\ 2 & 1 \\ 3 & 2 \end{bmatrix}}_C & \leftarrow & \underbrace{\begin{bmatrix} 1 & 2 & 0 & 3 \\ 2 & 4 & 1 & 4 \\ 3 & 6 & 2 & 5 \end{bmatrix}}_{A=CR'} & \rightarrow & \underbrace{\begin{bmatrix} 1 & -2 & 0 & 3 \\ 0 & 0 & 1 & -2 \\ 0 & 0 & 0 & 0 \end{bmatrix}}_{R \text{ in RREF}(1,3), Q} & \rightarrow & \underbrace{\begin{bmatrix} 1 & 2 & 0 & 3 \\ 0 & 0 & 1 & -2 \end{bmatrix}}_{R' \text{ in RREF}(1,3)} \\
 & & \text{Theorem 4.31} & & \text{Theorem 4.36} & & \text{Theorem 4.32}
 \end{array}$$

4.4 All solutions of $Ax = b$

Finally, we have come full circle: having motivated linear algebra from its two historical roots (analytic geometry and linear equations) in Section 0.3, we come back to linear equations. We are now in a position to precisely understand (and compute) the set of all solutions of a system of linear equations $Ax = b$. If there is some solution, all solutions can be obtained by suitably shifting the nullspace of A away from the origin. We also understand the kinds of systems that (typically) have a solution, and the ones that do not.

Using Gauss elimination and back substitution (Sections 3.2.2 and 3.2.1), we have seen how to compute the unique solution x of $Ax = b$ when A is an invertible square matrix.

In the general case, we can apply Gauss-Jordan elimination and direct solution (Sections 3.3.3 and 3.3.2) to either find the canonical solution, or to conclude that there is no solution. Now, we want to understand and compute the space of *all* solutions, so let us define this first.

Definition 4.37 (Solution space of a system of linear equations). *Let A be an $m \times n$ matrix and $b \in \mathbb{R}^m$. The set*

$$\text{Sol}(A, b) := \{x \in \mathbb{R}^n : Ax = b\} \subseteq \mathbb{R}^n$$

is the solution space of $Ax = b$.

If $b \neq 0$, $\text{Sol}(A, b)$ is unfortunately not a subspace of \mathbb{R}^n , simply because it does not contain the zero vector (see Lemma 4.9). Luckily, the vector space theory we have developed in this chapter still applies, but with a little twist.

Let us look at a concrete example, a system with one equation in two variables:

$$2x + 3y = 6.$$

The solutions, all pairs (x, y) satisfying $2x + 3y = 6$, form a line in \mathbb{R}^2 , not containing the origin; see the top line in Figure 4.3.

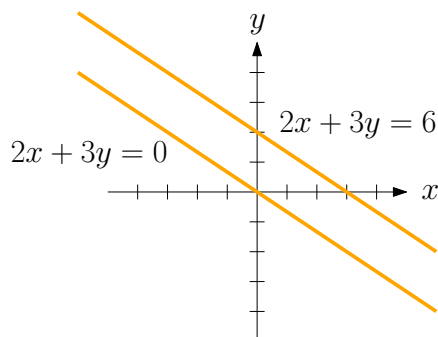


Figure 4.3: $\text{Sol}(A, b)$ for $A = \begin{bmatrix} 2 & 3 \end{bmatrix}$, $b = (6)$ (upper line) and $b = (0)$ (lower line).

Replacing the right-hand side 6 by 0 results in another line which contains the origin and is actually a subspace: the nullspace of the 1×2 matrix $\begin{bmatrix} 2 & 3 \end{bmatrix}$. We know how to compute nullspaces, see Section 4.3.3. In this example, we see that $\text{Sol}(\begin{bmatrix} 2 & 3 \end{bmatrix}, (6))$ and $\text{Sol}(\begin{bmatrix} 2 & 3 \end{bmatrix}, (0)) = \mathbf{N}(\begin{bmatrix} 2 & 3 \end{bmatrix})$ are parallel lines. To get $\text{Sol}(\begin{bmatrix} 2 & 3 \end{bmatrix}, (6))$ from $\mathbf{N}(\begin{bmatrix} 2 & 3 \end{bmatrix})$, we simply have to “shift” the nullspace “away from the origin”.

This picture generalizes: For $\mathbf{b} \neq \mathbf{0}$, and if there is a solution at all, $\text{Sol}(A, \mathbf{b})$ can be obtained by shifting the nullspace of A by an arbitrary solution of $A\mathbf{x} = \mathbf{b}$.

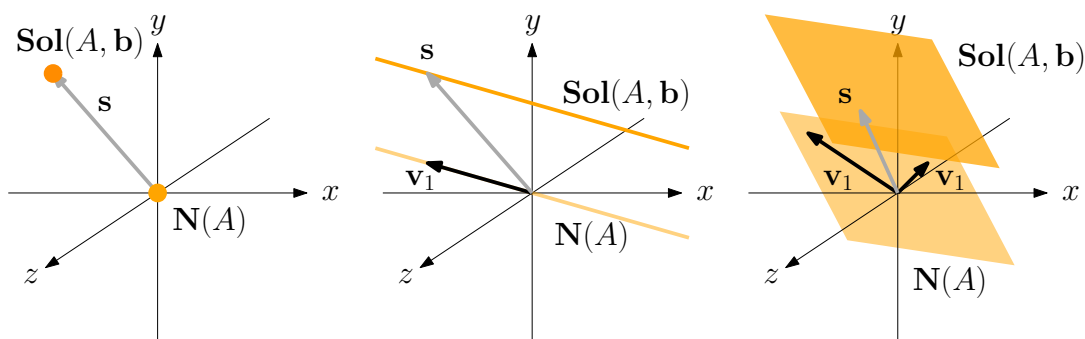
Theorem 4.38 (Solution space from shifting the nullspace). *Let A be an $m \times n$ matrix, $\mathbf{b} \in \mathbb{R}^m$. Let \mathbf{s} be some solution of $A\mathbf{x} = \mathbf{b}$. Then*

$$\text{Sol}(A, \mathbf{b}) = \{\mathbf{s} + \mathbf{x} : \mathbf{x} \in \mathbf{N}(A)\}.$$

Hence, we can also compute $\text{Sol}(A, \mathbf{b})$, despite the fact that it is not a subspace. To describe all solutions, we just need *some* solution \mathbf{s} (for example the canonical one from Section 3.3.2) and a basis $B = \{\mathbf{v}_1, \mathbf{v}_2, \dots, \mathbf{v}_{n-r}\}$ of $\mathbf{N}(A)$ (see Theorem 4.36). Then

$$\text{Sol}(A, \mathbf{b}) = \left\{ \mathbf{s} + \sum_{i=1}^{n-r} \lambda_i \mathbf{v}_i : \lambda_i \in \mathbb{R} \text{ for } i \in [n-r] \right\}.$$

$\text{Sol}(A, \mathbf{b})$ can be a point, a line, a plane, ...:



From these pictures, you can also try to get an intuition why the result of shifting $\mathbf{N}(A)$ does not depend on the particular solution \mathbf{s} that we choose.

Proof of Theorem 4.38. We first show that every solution $\mathbf{y} \in \text{Sol}(A, \mathbf{b})$ is of the form $\mathbf{s} + \mathbf{x}$ with $\mathbf{x} \in \mathbf{N}(A)$. For this, we write \mathbf{y} as

$$\mathbf{y} = \mathbf{s} + \underbrace{(\mathbf{y} - \mathbf{s})}_{\mathbf{x}},$$

and due to $A\mathbf{x} = A(\mathbf{y} - \mathbf{s}) = A\mathbf{y} - A\mathbf{s} = \mathbf{b} - \mathbf{b} = \mathbf{0}$ we have $\mathbf{x} \in \mathbf{N}(A)$. The second equality in this chain uses linearity of matrix transformations, see Lemma 2.19. For the other direction, we show that every vector \mathbf{y} of the form $\mathbf{y} = \mathbf{s} + \mathbf{x}$ with $\mathbf{x} \in \mathbf{N}(A)$ is in $\text{Sol}(A, \mathbf{b})$. For this, we compute

$$A\mathbf{y} = A(\mathbf{s} + \mathbf{x}) = A\mathbf{s} + A\mathbf{x} = \mathbf{b} + \mathbf{0} = \mathbf{b}.$$

□

4.4.1 Affine subspaces

Generally, a shifted copy of a subspace in a vector space is called an *affine subspace*. In the context of affine subspaces, a “normal” subspace is also sometimes called *linear subspace*. There is a full theory of *affine spaces* which are essentially vector spaces without an origin. The elements of an affine space are called *points*. You can think of them as locations in space whose coordinates are “absolute” and not defined relative to a distinguished origin $\mathbf{0}$. Natural point behavior is to move $(\mathbf{p} + \mathbf{v})$, where \mathbf{v} is a vector from the vector space “behind” the affine space. This also allows to subtract points $(\mathbf{p} - \mathbf{q})$, resulting in a *difference vector*. But unlike vectors, points do not add up or scale. In summary, affine spaces are natural habitats for points.³

4.4.2 The number of solutions

A system of linear equations has two characteristic numbers: m , the number of equations, and n , the number of variables. But if we want to understand the solution space, there is a third important characteristic number: r , the rank of the coefficient matrix A . This is revealed by Gauss-Jordan elimination on A : r is the number of “downward steps” in the reduced row echelon form of the resulting matrix R ; see Theorem 4.31.

The number of solutions of $A\mathbf{x} = \mathbf{b}$ can be 0 (the system is unsolvable), 1 (there is a unique solution), or infinite. In the latter case, we would like to be more precise and know the dimension of the solution space. We have already done all the work for this and summarize it in the following result.

Theorem 4.39 (Dimension of the solution space). *Let A be an $m \times n$ matrix of rank r . If $A\mathbf{x} = \mathbf{b}$ has a solution, then $\text{Sol}(A, \mathbf{b})$ has dimension $n - r$, where $\dim(\text{Sol}(A, \mathbf{b})) := \dim(\mathbf{N}(A))$.*

If $\dim(\text{Sol}(A, \mathbf{b})) = 0, 1, 2, \dots$, the solution space is a point, a line, a plane,...

Proof. If there is a solution at all, the solution space is obtained by shifting the nullspace $\mathbf{N}(A)$ away from the origin by some solution \mathbf{s} (Theorem 4.38). We declare $\text{Sol}(A, \mathbf{b})$ to be of the same dimension as the nullspace, and this dimension is $n - r$ by Theorem 4.36. \square

Theorem 4.39 covers all cases in which there is a solution. Next, we want to understand in more detail in which cases this happens. As we will see, this typically depends on whether $r = m$ or $r < m$. The case $r > m$ cannot happen, see Corollary 4.34.

Theorem 4.40 (Systems of rank m are solvable). *Let A be an $m \times n$ matrix of rank $r = m$. Then $A\mathbf{x} = \mathbf{b}$ has a solution for every $\mathbf{b} \in \mathbb{R}^m$.*

Proof. From Theorem 4.31, we know that $\dim(\mathbf{C}(A)) = r = m$, so $\mathbf{C}(A)$ is a subspace of \mathbb{R}^m of the same dimension m . Then we must have $\mathbf{C}(A) = \mathbb{R}^m$. This seems obvious, but actually needs an argument. The argument is that every basis B of $\mathbf{C}(A)$ contains m linearly independent vectors in \mathbb{R}^m , and these already span \mathbb{R}^m by Lemma 1.28.

³https://en.wikipedia.org/wiki/Affine_space, accessed September 2, 2025

It follows that every $\mathbf{b} \in \mathbb{R}^m$ is in $C(A)$, and this is the same as saying that the system $A\mathbf{x} = \mathbf{b}$ has a solution for every $\mathbf{b} \in \mathbb{R}^m$. \square

If $r < m$, the column space $C(A)$ has dimension smaller than m . Then $A\mathbf{x} = \mathbf{b}$ is “typically” unsolvable, because “almost all” \mathbf{b} are not in $C(A)$. For an illustration, consider Figure 2.2 (right) where the column space is a line in \mathbb{R}^2 . If we pick a “typical” \mathbf{b} from \mathbb{R}^2 , we expect \mathbf{b} to be outside of that line. This can properly be formalized: a subspace of \mathbb{R}^m of dimension smaller than m is a *set of measure 0*. A “typical” vector (a vector randomly chosen from a *continuous probability distribution* over \mathbb{R}^m) has probability 0 to land in a set of measure 0. We do not go the formal route here and summarize the discussion as follows.

Informal Theorem 4.41 (Systems of rank less than m are typically unsolvable). *Let A be an $m \times n$ matrix of rank $r < m$. For a “typical” \mathbf{b} , the system $A\mathbf{x} = \mathbf{b}$ has no solution.*

It may happen that in a given application, we have an “untypical” right-hand side that leads to a solvable system, despite $r < m$. The previous informal theorem is a rule of thumb about what to expect without knowing anything specific about the system.

Depending on the shape of the coefficient matrix A (see Figure 2.1), systems of linear equations come in three categories.

Definition 4.42 (Square, underdetermined, overdetermined system). *Let A be an $m \times n$ matrix, $\mathbf{b} \in \mathbb{R}^m$.*

- (i) *If $m = n$ (A is a square matrix), the system $A\mathbf{x} = \mathbf{b}$ is called square.*
- (ii) *If $m < n$ (A is a wide matrix), the system $A\mathbf{x} = \mathbf{b}$ is called underdetermined.*
- (iii) *If $m > n$ (A is a tall matrix), the system $A\mathbf{x} = \mathbf{b}$ is called overdetermined.*

The notion *underdetermined* comes from the fact that there are more variables than equations, in which case it is not possible to uniquely “nail down” the values of the variables from the information given (the equations). Indeed, in the underdetermined case, the nullspace $N(A)$ has dimension $n - r > m - r \geq 0$ (see Theorem 4.36 for the dimension and Corollary 4.34 for $m - r \geq 0$), so it is at least a line. Hence, if there is a solution at all, there are infinitely many (see Theorem 4.39).

In contrast, an *overdetermined* system has more equations than variables. The notion indicates that such a system contains more information about the variables than necessary. Indeed, we know that having as many equations as variables (a square system) is in principle sufficient to determine the values of the variables uniquely: all we need is that $r = m = n$ (see Theorem 4.40 together with Theorem 4.39).

We conclude with two more informal theorems.

Informal Theorem 4.43 (Overdetermined systems are typically unsolvable). *Let A be a tall matrix. For a “typical” \mathbf{b} , the overdetermined system $A\mathbf{x} = \mathbf{b}$ has no solution.*

Proof. We have $r \leq n$ (Corollary 4.34) and $n < m$ since A is tall. Hence, $r < m$, so there is “typically” no solution by Informal Theorem 4.41. \square

For a square or an underdetermined system, we want to say that it is “typically” solvable. But for this, we have to consider a “typical” matrix A instead of a “typical” right-hand side \mathbf{b} . For example, if $A = 0$, the system is unsolvable for “typical” \mathbf{b} (the only solvable case is $\mathbf{b} = \mathbf{0}$), but if A is “typical”, the system turns out to be solvable for all \mathbf{b} . So here, solvability strongly depends on the “typicality” of A .

Informal Theorem 4.44 (Square and Underdetermined systems are typically solvable). *Let $\mathbf{b} \in \mathbb{R}^m$, $m \leq n$. For a “typical” matrix $A \in \mathbb{R}^{m \times n}$, the square or underdetermined system $A\mathbf{x} = \mathbf{b}$ has a solution.*

For this, we use that “typical” matrices with $m \leq n$ have rank $r = m$, so Theorem 4.40 applies. The formal argument is that matrices with $r < m$ are a set of measure 0 (in the space of $m \times n$ matrices with $m \leq n$). To make this argument, we need *determinants* that will only be introduced in the second part of the course. But we can get a glimpse of this by looking at the 2×2 case. We can argue that a “typical” 2×2 matrix

$$A = \begin{bmatrix} a & b \\ c & d \end{bmatrix}$$

has rank 2. To (informally) see this, recall that the rank is the number of independent columns; see Definition 2.10. The columns are two “typical” vectors

$$\begin{pmatrix} a \\ c \end{pmatrix}, \begin{pmatrix} b \\ d \end{pmatrix}$$

in \mathbb{R}^2 , and we get rank smaller than 2 only when they are linearly dependent, i.e. point into the same (or opposite) direction. But this is “untypical”.

Bibliography

- [BP98] Sergey Brin and Lawrence Page. The anatomy of a large-scale hypertextual web search engine. *Computer Networks and ISDN Systems*, 30(1):107–117, 1998. <https://www.sciencedirect.com/science/article/pii/S016975529800110X>.
- [CLRS22] Cormen, Thomas H., Leiserson, Charles E., Rivest, Ronald L., and Stein, Clifford. *Introduction to Algorithms, Fourth Edition*. The MIT Press, 2022. <https://mitpress.mit.edu/9780262046305/introduction-to-algorithms/>.
- [CSB06] V. Claus, A. Schwill, and R. Böving. *Duden Informatik A - Z: Fachlexikon für Studium, Ausbildung und Beruf, 4. Auflage*. Dudenverlag, 2006.
- [Fel93] Michael R. Fellows. Computer science and mathematics in the elementary schools. In Harvey B. Keynes Naomi Fisher and Philip D. Wagreich, editors, *Mathematicians and Education Reform 1990–1991*, volume 3 of *CBMS Issues in Mathematics Education*. American Mathematical Society, 1993. <https://ianparberry.com/research/cseducation/fellows1991.pdf>.
- [Str23] Gilbert Strang. *Introduction to Linear Algebra (Sixth Edition)*. Wellesley-Cambridge Press, 2023.

Index

- \emptyset , 37
- ∞ -norm, 28
- $\mathbf{0}$, 13
- 1, 30
- m -tuples, 12
- 1-norm, 28

- absolute value, 25
- abuse of notation, 13
- affine combination, 19
- affine space, 156
- affine subspace, 156
- AI chips, 45
- Al-Khwarizmi, 7
- algebra, 7, 93
- algorithm, 7
- analytic geometry, 8
- applicable, 49
- apply, 35
- arrows, 11
- assignment symbol, 100
- associative, 15
- associativity, 75
- axiom, 63
- axioms of a subspace, 130
- axioms of a vector space, 127

- back substitution, 99
- base case, 66
- basis, 126, 134, 137
- below the staircase, 115
- big-O notation, 109
- bijective, 86
- bits, 127

- canonical, 117, 128
- canonical basis, 138
- Carl Friedrich Gauss, 66
- Cartesian coordinate system, 12
- Cartesian coordinates, 9
- category theory, 145
- citations, 96
- codomain, 35
- coefficient matrix, 95
- coefficients, 128
- collinear, 36, 40
- column notation, 46
- column rank, 52
- column space, 51
- column vector, 12
- commutative, 15
- commutative diagram, 59
- complex numbers, 127
- composition, 71
- compression, 84
- computer science, 7
- computing, 147
- condition, 36
- cone, 20
- conic combination, 19
- constant time, 109
- continuous probability distribution, 157
- contraposition, 108, 147
- convex combination, 19, 60
- coordinate vector, 12, 147
- coordinate-wise, 14
- coordinates, 13
- coplanar, 41

- corollary, 39
- covector, 34, 35
- covector-matrix multiplication, 76
- Cramer's rule, 91
- damping factor, 97
- defining property, 63
- definition, 13
- degree, 128
- degrees of freedom, 142, 147
- dependent, 52
- determinants, 91, 158
- diagonal, 48, 54
- difference vector, 156
- dimension, 126
- distributivity, 24, 75
- domain, 35
- domino effect, 66
- dual space, 35
- dynamic programming, 80
- elimination, 100
- elimination matrix, 101
- empty set, 37
- equivalent, 33, 38
- Euclidean norm, 25
- finitely generated, 139
- floating-point numbers, 113
- follows, 39
- function, 35
- function value, 35
- fundamental subspaces, 148
- Gauss elimination, 98
- generalized associativity, 76
- Hadamard product, 23
- hyperplane, 32
- identity, 48, 64
- identity function, 86
- if, 38
- if and only if, 38
- ill-formed, 81

- image, 68
- implies, 38
- independent, 52, 55
- independent column, 51
- index, 12
- induction hypothesis, 66
- induction step, 66
- Informatik, 7
- information, 7
- injective, 86
- interval, 31
- invariance, 104
- inverse, 86, 90
- invertible, 90
- isomorphic, 133, 144, 146
- isomorphism, 126, 146
- kernel, 68
- Kronecker delta, 48
- lemma, 29
- limited induction, 67
- line segment, 20
- linear combination, 16, 135
- linear equations, 8
- linear form, 35
- linear functional, 63
- linear subspace, 156
- linear transformation, 63
- linear transformation between vector spaces, 143
- linear transformations, 45
- linearity, 58
- linearity in both arguments, 24
- linearly dependent, 36, 136
- linearly independent, 36, 136
- link graph, 96
- loop invariants, 140
- lower triangular, 49
- LU decomposition, 113
- LUP decomposition, 113
- maps, 35
- matrix, 46

matrix transformation, 58
 matrix transposition, 54
 Matrix-vector multiplication, 49
 matrix-vector product, 49

 natural numbers, 12
 negation, 108
 negations, 39
 non-invertible, 90
 non-singular, 90
 nontrivial linear combination, 38
 nullspace, 57
 numerical mathematics, 113

 observation, 24
 only if, 38
 origin, 12
 orthogonal, 32
 outer product, 76, 78
 overdetermined, 157
 overloading, 128

 parallel, 63
 parallel projection, 61, 152
 parallelogram, 14
 permutation matrix, 103
 perspective projection, 61
 pivot, 102
 points, 156
 polygon, 59
 polynomial without constant term, 133
 positive-definiteness, 24
 predatory publishers, 96
 preprocessing, 125
 private nonzero, 137
 product, 49, 72, 78
 projection, 61
 proof by contradiction, 87
 proof by induction, 66
 pseudocode, 100
 Pythagorean theorem, 25

 quadratic equations, 93
 quadratic polynomial, 133

 rank, 52
 real numbers, 12
 real polynomials, 128
 real vector space, 127
 recursive algorithm, 140
 reduced row echelon form, 114, 115
 replay, 113
 right-hand side, 95
 rotation, 60
 row addition, 102
 row division, 119
 row divisions, 114
 row exchange, 102
 row notation, 46
 row operations, 99, 103
 row rank, 52, 55
 row rank equals columns rank, 150
 row space, 55
 row subtraction, 101
 row vector, 34, 35

 scalar, 15
 scalar division, 27
 scalar multiple, 15, 47
 scalar multiplication, 15
 scalar product, 24
 scaling, 60
 school book addition, 7
 sequences, 12
 set, 22
 set builder notation, 20, 22
 set of measure 0, 157
 shear, 60
 size of a set, 25
 solution space, 154
 span, 40, 136
 special solutions, 151, 153
 square, 157
 square matrix, 47
 standard basis, 137, 153
 standard form, 84, 114, 123
 standard unit vectors, 27
 Steinitz exchange lemma, 44, 140

- stretching, 60
- stretching factors, 60
- submatrix, 53
- subset, 41
- subspace, 126, 130
- sum, 14, 47
- summation index, 21
- surjective, 86
- symmetric, 49
- symmetry, 24
- system of linear equations, 94

- taking out scalars, 24
- tall matrices, 48
- three fundamental subspaces, 126
- trace, 134
- transpose, 34, 35, 54
- trivial linear combination, 39
- trivial solution, 97

- underdetermined, 157
- unit circle, 26
- unit cube, 61
- unit monomials, 136
- unit vector, 26
- upper triangular, 48, 99

- variables in a programming language, 100
- vector, 12
- vector addition, 14
- vector of variables, 95
- vector space, 126, 127

- weighted, 97
- wide matrices, 48

- zero matrix, 47
- zero polynomial, 128
- zero vector, 13