Grundlehren der mathematischen Wissenschaften 349
A Series of Comprehensive Studies in Mathematics

Peter Bürgisser
Felipe Cucker

Condition

The Geometry of Numerical Algorithms

Springer

Grundlehren der mathematischen Wissenschaften 349

A Series of Comprehensive Studies in Mathematics

Peter Bürgisser • Felipe Cucker

Condition

The Geometry of Numerical Algorithms

Springer

Peter Bürgisser
Institut für Mathematik
Technische Universität Berlin
Berlin, Germany

Felipe Cucker
Department of Mathematics
City University of Hong Kong
Hong Kong, Hong Kong SAR

ISSN 0072-7830 Grundlehren der mathematischen Wissenschaften
ISBN 978-3-642-38895-8 ISBN 978-3-642-38896-5 (eBook)
DOI 10.1007/978-3-642-38896-5
Springer Heidelberg New York Dordrecht London

Library of Congress Control Number: 2013946090

Mathematics Subject Classification (2010): 15A12, 52A22, 60D05, 65-02, 65F22, 65F35, 65G50, 65H04, 65H10, 65H20, 90-02, 90C05, 90C31, 90C51, 90C60, 68Q25, 68W40, 68Q87

Printed on acid-free paper

Springer is part of Springer Science+Business Media (www.springer.com)

Dedicated to the memory of
Walter Bürgisser and Gritta Bürgisser-Glogau
and of
Federico Cucker and Rosemary Farkas
in love and gratitude

Preface

Motivation A combined search at Mathscinet and Zentralblatt shows more than 800 articles with the expression "condition number" in their title. It is reasonable to assume that the number of articles dealing with conditioning, in one way or another, is a substantial multiple of this quantity. This is not surprising. The occurrence of condition numbers in the accuracy analysis of numerical algorithms is pervasive, and its origins are tied to those of the digital computer. Indeed, the expression "condition number" itself was first introduced in 1948, in a paper by Alan M. Turing in which he studied the propagation of errors for linear equation solving with the then nascent computing machinery [221]. The same subject occupied John von Neumann and Herman H. Goldstine, who independently found results similar to those of Turing [226]. Ever since then, condition numbers have played a leading role in the study of both accuracy and complexity of numerical algorithms.

To the best of our knowledge, and in stark contrast to this prominence, there is no book on the subject of conditioning. Admittedly, most books on numerical analysis have a section or chapter devoted to it. But their emphasis is on algorithms, and the links between these algorithms and the condition of their data are not pursued beyond some basic level (for instance, they contain almost no instances of probabilistic analysis of algorithms via such analysis for the relevant condition numbers).

Our goal in writing this book has been to fill this gap. We have attempted to provide a unified view of conditioning by making condition numbers the primary object of study and by emphasizing the many aspects of condition numbers in their relation to numerical algorithms.

Structure The book is divided into three parts, which approximately correspond to themes of conditioning in linear algebra, linear programming, and polynomial equation solving, respectively. The increase in technical requirements for these subjects is reflected in the different paces for their expositions. Part I proceeds leisurely and can be used for a semester course at the undergraduate level. The tempo increases in Part II and reaches its peak in Part III with the exposition of the recent advances in and partial solutions to the 17th of the problems proposed by Steve Smale for the mathematicians of the 21st century, a set of results in which conditioning plays a paramount role [27, 28, 46].

As in a symphonic poem, these changes in cadence underlie a narration in which, as mentioned above, condition numbers are the main character. We introduce them, along with the cast of secondary characters making up the *dramatis personae* of this narration, in the *Overture* preceding Part I.

We mentioned above that Part I can be used for a semester course at the undergraduate level. Part II (with some minimal background from Part I) can be used as an undergraduate course as well (though a notch more advanced). Briefly stated, it is a "condition-based" exposition of linear programming that, unlike more elementary accounts based on the simplex algorithm, sets the grounds for similar expositions of convex programming. Part III is also a course on its own, now on computation with polynomial systems, but it is rather at the graduate level.

Overlapping with the primary division of the book into its three parts there is another taxonomy. Most of the results in this book deal with condition numbers of specific problems. Yet there are also a few discussions and general results applying either to condition numbers in general or to large classes of them. These discussions are in most of the *Overture*, the two *Intermezzi* between parts, Sects. 6.1, 6.8, 9.5, and 14.3, and Chaps. 20 and 21. Even though few, these pages draft a general theory of condition, and most of the remainder of the book can be seen as worked examples and applications of this theory.

The last structural attribute we want to mention derives from the technical characteristics of our subject, which prominently features probability estimates and, in Part III, demands some nonelementary geometry. A possible course of action in our writing could have been to act like Plato and deny access to our edifice to all those not familiar with geometry (and, in our case, probabilistic analysis). We proceeded differently. Most of the involved work in probability takes the form of estimates—of either distributions' tails or expectations—for random variables in a very specific context. We therefore included within the book a *Crash Course on Probability* providing a description of this context and the tools we use to compute these estimates. It goes without saying that probability theory is vast, and alternative choices in its toolkit could have been used as well. A penchant for brevity, however, prevented us to include these alternatives. The course is supplied in installments, six in total, and contains the proofs of most of its results. Geometry requirements are of a more heterogeneous nature, and consequently, we have dealt with them differently. Some subjects, such as Euclidean and spherical convexity, and the basic properties of projective spaces, are described in detail within the text. But we could not do so with the basic notions of algebraic, differential, and integral geometry. We therefore collected these notions in an appendix, providing only a few proofs.

Paderborn, Germany — Peter Bürgisser
Hong Kong, Hong Kong SAR — Felipe Cucker
May 2013

Acknowledgements

A substantial part of the material in this book formed the core of several graduate courses taught by PB at the University of Paderborn. Part of the material was also used in a graduate course at the Fields Institute held in the fall of 2009. We thank all the participants of these courses for valuable feedback. In particular, Dennis Amelunxen, Christian Ikenmeyer, Stefan Mengel, Thomas Rothvoss, Peter Scheiblechner, Sebastian Schrage, and Martin Ziegler, who attended the courses in Paderborn, had no compassion in pointing to the lecturer the various forms of typos, redundancies, inaccuracies, and plain mathematical mistakes that kept popping up in the early drafts of this book used as the course's main source. We thank Dennis Amelunxen for producing a first LaTeX version of the lectures in Paderborn, which formed the initial basis of the book. In addition, Dennis was invaluable in producing the `TikZ` files for the figures occurring in the book.

Also, Diego Armentano, Dennis Cheung, Martin Lotz, and Javier Peña read various chapters and have been pivotal in shaping the current form of these chapters. We have pointed out in the Notes the places where their input is most notable.

Finally, we want to emphasize that our viewpoint about conditioning and its central role in the foundations of numerical analysis evolved from hours of conversations and exchange of ideas with a large group of friends working in similar topics. Among them it is impossible not to mention Carlos Beltrán, Lenore Blum, Irenée Briquel, Jean-Pierre Dedieu, Alan Edelman, Raphael Hauser, Gregorio Malajovich, Luis Miguel Pardo, Jim Renegar, Vera Roshchina, Michael Shub, Steve Smale, Henryk Woźniakowski, and Mario Wschebor. We are greatly indebted to all of them.

The financial support of the German Research Foundation (individual grants BU 1371/2-1 and 1371/2-2) and the GRF (grant CityU 100810) is gratefully acknowledged. We also thank the Fields Institute in Toronto for hospitality and financial support during the thematic program on the Foundations of Computational Mathematics in the fall of 2009, where a larger part of this monograph took definite form.

We thank the staff at Springer-Verlag in Basel and Heidelberg for their help and David Kramer for the outstanding editing work he did on our manuscript.

Finally, we are grateful to our families for their support, patience, and understanding of the commitment necessary to carry out such a project while working on different continents.

Contents

Overture: On the Condition of Numerical Problems

O.1 The Size of Errors

Since none of the numbers we take out from logarithmic or trigonometric tables admit of absolute precision, but are all to a certain extent approximate only, the results of all calculations performed by the aid of these numbers can only be approximately true. [...] It may happen, that in special cases the effect of the errors of the tables is so augmented that we may be obliged to reject a method, otherwise the best, and substitute another in its place.

Carl Friedrich Gauss, *Theoria Motus*

The heroes of numerical mathematics (Euler, Gauss, Lagrange, ...) developed a good number of the algorithmic procedures which constitute the essence of numerical analysis. At the core of these advances was the invention of calculus. And underlying the latter, the field of real numbers.

The dawn of the digital computer, in the decade of the 1940s, allowed the execution of these procedures on increasingly large data, an advance that, however, made even more patent the fact that real numbers cannot be encoded with a finite number of bits and therefore that computers had to work with approximations only. With the increased length of computations, the systematic rounding of all occurring quantities could now accumulate to a greater extent. Occasionally, as already remarked by Gauss, the errors affecting the outcome of a computation were so big as to make it irrelevant.

Expressions like "the error is big" lead to the question, how does one measure an error? To approach this question, let us first assume that the object whose error we are considering is a single number x encoding a quantity that may take values on an open real interval. An error of magnitude 1 may yield another real number $\tilde{x}$ with value either $x-1$ or $x+1$. Intuitively, this will be harmless or devastating depending on the magnitude of x itself. Thus, for $x=10^6$, the error above is hardly noticeable, but for $x=10^{-3}$, it certainly is (and may even change basic features of

x such as being positive). A relative measure of the error appears to convey more meaning. We therefore define[1]

$$\mathsf{RelError}(x) = \frac{|\tilde{x} - x|}{|x|}.$$

Note that this expression is well defined only when $x \neq 0$.

How does this measure extend to elements $x \in \mathbb{R}^m$? We want to consider relative errors as well, but how does one relativize? There are essentially two ways:

Componentwise: Here we look at the relative error in each component, taking as error for x the maximum of them. That is, for $x \in \mathbb{R}^m$ such that $x_i \neq 0$ for $i = 1, \dots, m$, we define

$$\mathsf{RelError}(x) = \max_{i \le m} \mathsf{RelError}(x_i).$$

Normwise: Endowing $\mathbb{R}^m$ with a norm allows one to mimic, for $x \neq 0$, the definition for the scalar case. We obtain

$$\mathsf{RelError}(x) = \frac{\|\tilde{x} - x\|}{\|x\|}.$$

Needless to say, the normwise measure depends on the choice of the norm.

O.2 The Cost of Erring

How do round-off errors affect computations? The answer to this question depends on a number of factors: the problem being solved, the data at hand, the algorithm used, the machine precision (as well as other features of the computer's arithmetic). While it is possible to consider all these factors together, a number of idealizations leading to the consideration of simpler versions of our question appears as a reasonable—if not necessary—course of action. The notion of condition is the result of some of these idealizations. More specifically, assume that the problem being solved can be described by a function

$$\varphi : \mathcal{D} \subseteq \mathbb{R}^m \to \mathbb{R}^q,$$

where $\mathcal{D}$ is an open subset of $\mathbb{R}^m$. Assume as well that the computation of φ is performed by an algorithm with infinite precision (that is, there are no round-off errors during the execution of this algorithm). All errors in the computed value arise as a consequence of possible errors in reading the input (which we will call *perturbations*). Our question above then takes the following form:

How large is the output error with respect to the input perturbation?

[1]To be completely precise, we should write $\mathsf{RelError}(x, \tilde{x})$. In all what follows, however, to simplify notation, we will omit the perturbation $\tilde{x}$ and write simply $\mathsf{RelError}(x)$.

The *condition number* of input $a \in \mathcal{D}$ (with respect to problem φ) is, roughly speaking, the worst possible magnification of the output error with respect to a small input perturbation. More formally,

$$\mathsf{cond}^{\varphi}(a) = \lim_{\delta \to 0} \sup_{\mathsf{RelError}(a) \le \delta} \frac{\mathsf{RelError}(\varphi(a))}{\mathsf{RelError}(a)}. \tag{O.1}$$

This expression defines the condition number as a limit. For small values of δ we can consider the approximation

$$\mathsf{cond}^{\varphi}(a) \approx \sup_{\mathsf{RelError}(a) \le \delta} \frac{\mathsf{RelError}(\varphi(a))}{\mathsf{RelError}(a)}$$

and, for practical purposes, the approximate bound

$$\mathsf{RelError}\big(\varphi(a)\big) \lessapprox \mathsf{cond}^{\varphi}(a)\mathsf{RelError}(a), \tag{O.2}$$

or yet, using "little oh" notation[2] for $\mathsf{RelError}(a) \to 0$,

$$\mathsf{RelError}\big(\varphi(a)\big) \le \mathsf{cond}^{\varphi}(a)\mathsf{RelError}(a) + o\big(\mathsf{RelError}(a)\big). \tag{O.3}$$

Expression (O.1) defines a family of condition numbers for the pair (φ, a). Errors can be measured either componentwise or normwise, and in the latter case, there is a good number of norms to choose from. The choice of normwise or componentwise measures for the errors has given rise to three kinds of condition numbers (condition numbers for normwise perturbations and componentwise output errors are not considered in the literature).

		PERTURBATION	
		normwise	componentwise
OUTPUT	normwise	*normwise*	*mixed*
ERROR	componentwise		*componentwise*

We will generically denote normwise condition numbers by $\mathsf{cond}^{\varphi}(a)$, mixed condition numbers by $\mathsf{M}^{\varphi}(a)$, and componentwise condition numbers by $\mathsf{Cw}^{\varphi}(a)$. We may skip the superscript ${}^{\varphi}$ if it is clear from the context. In the case of componentwise condition numbers one may be interested in considering the relative error for each of the output components separately. Thus, for $j \le q$ one defines

$$\mathsf{Cw}_j^{\varphi}(a) = \lim_{\delta \to 0} \sup_{\mathsf{RelError}(a) \le \delta} \frac{\mathsf{RelError}(\varphi(a)_j)}{\mathsf{RelError}(a)},$$

and one has $\mathsf{Cw}^{\varphi}(a) = \max_{j \le q} \mathsf{Cw}_j^{\varphi}(a)$.

[2] A short description of the little oh and other asymptotic notations is in the Appendix, Sect. A.1.

The consideration of a normwise, mixed, or componentwise condition number will be determined by the characteristics of the situation at hand. To illustrate this, let's look at data perturbation. The two main reasons to consider such perturbations are inaccurate data reading and backward-error analysis.

In the first case the idea is simple. We are given data that we know to be inaccurate. This may be because we obtained it by measurements with finite precision (e.g., when an object is weighed, the weight is displayed with a few digits only) or because our data are the result of an inaccurate computation.

The idea of backward-error analysis is less simple (but very elegant). For a problem φ we may have many algorithms that solve it. While all of them ideally compute φ when endowed with infinite precision, under the presence of errors they will compute only approximations of this function. At times, for a problem φ and a finite-precision algorithm $\mathcal{A}^\varphi$ solving it, it is possible to show that for all $a \in \mathcal{D}$ there exists $e \in \mathbb{R}^m$ with $a + e \in \mathcal{D}$ satisfying

$$\begin{array}{ll} (*) & \mathcal{A}^\varphi(a) = \varphi(a+e), \quad \text{and} \\ (**) & e \text{ is small with respect to } a. \end{array}$$

In this situation—to which we refer by saying that $\mathcal{A}^\varphi$ is *backward-stable*—information on how small exactly e is (i.e., how large $\mathsf{RelError}(a)$ is) together with the condition number of a directly yields bounds on the error of the computed quantity $\mathcal{A}^\varphi(a)$. For instance, if $(**)$ above takes the form

$$\|e\| \le m^3 10^{-6} \|a\|,$$

we will deduce, using (O.2), that

$$\big\|\mathcal{A}^\varphi(a) - \varphi(a)\big\| \lessapprox \mathsf{cond}^\varphi(a) m^3 10^{-6} \big\|\varphi(a)\big\|. \tag{O.4}$$

No matter whether due to inaccurate data reading or because of a backward-error analysis, we will measure the perturbation of a in accordance with the situation at hand. If, for instance, we are reading data in a way that each component a_i satisfies $\mathsf{RelError}(a_i) \le 5 \times 10^{-8}$, we will measure perturbations in a componentwise manner. If, in contrast, a backward-error analysis yields an e satisfying $\|e\| \le m^3 \|a\| 10^{-6}$, we will have to measure perturbations in a normwise manner.

While we may have more freedom in the way we measure the output error, there are situations in which a given choice seems to impose itself. Such a situation could arise when the outcome of the computation at hand is going to be the data of another computation. If perturbations of the latter are measured, say, componentwise, we will be interested in doing the same with the output error of the former. A striking example in which error analysis can be only appropriately explained using componentwise conditioning is the solution of triangular systems of equations. We will return to this issue in Chap. 3.

At this point it is perhaps convenient to emphasize a distinction between *condition* and (backward) *stability*. Given a problem φ, the former is a property of the input only. That is, it is independent on the possible algorithms used to compute φ.

In contrast, backward stability, at least in the sense defined above, is a property of an algorithm $\mathscr{A}^\varphi$ computing φ that holds for all data $a \in \mathcal{D}$ (and is therefore independent of particular data instances).

Expressions like (O.4) are known as forward-error analyses, and algorithms $\mathscr{A}^\varphi$ yielding a small value of $\frac{\|\mathscr{A}^\varphi(a)-\varphi(a)\|}{\|\varphi(a)\|}$ are said to be *forward-stable*. It is important to mention that while backward-error analyses immediately yield forward-error bounds, some problems do not admit backward-error analysis, and therefore, their error analysis must be carried forward.

It is time to have a closer look at the way errors are produced in a computer.

O.3 Finite-Precision Arithmetic and Loss of Precision

O.3.1 Precision ...

Although the details of computer arithmetic may vary with computers and software implementations, the basic idea was agreed upon shortly after the dawn of digital computers. It consisted in fixing positive integers $\beta \geq 2$ (the *basis* of the representation), t (its *precision*), and e_0, and approximating nonzero real numbers by rational numbers of the form

$$z = \pm \frac{m}{\beta^t}\, \beta^e$$

with $m \in \{1, \dots, \beta^t\}$ and $e \in \{-e_0, \dots, e_0\}$. The fraction $\frac{m}{\beta^t}$ is called the *mantissa* of z and the integer e its *exponent*. The condition $|e| \leq e_0$ sets limits on how big (and how small) z may be. Although these limits may give rise to situations in which (the absolute value of) the number to be represented is too large (*overflow*) or too small (*underflow*) for the possible values of z, the value of e_0 in most implementations is large enough to make these phenomena rare in practice. Idealizing a bit, we may assume $e_0 = \infty$.

As an example, taking $\beta = 10$ and $t = 12$, we can approximate

$$\pi^8 \approx 0.948853101607 \times 10^4.$$

The relative error in this approximation is bounded by 1.1×10^{-12}. Note that t is the number of correct digits of the approximation. Actually, for any real number x, by appropriately rounding and truncating an expansion of x we can obtain a number $\tilde{x}$ as above satisfying $\tilde{x} = x(1+\delta)$ with $|\delta| \leq \frac{\beta^{-t+1}}{2}$. That is,

$$\mathsf{RelError}(x) \leq \frac{\beta^{-t+1}}{2}.$$

More generally, whenever a real number x is approximated by $\tilde{x}$ satisfying an in-equality like the one above, we say that $\tilde{x}$ *approximates* x *with* t *correct digits*.[3]

Leaving aside the details such as the choice of basis and the particular way a real number is truncated to obtain a number as described above, we may summarize the main features of computer arithmetic (recall that we assume $e_0 = \infty$) by stating the existence of a subset $\mathbb{F} \subset \mathbb{R}$ containing 0 (the *floating-point numbers*), a *rounding map* $\mathsf{round} : \mathbb{R} \to \mathbb{F}$, and a *round-off unit* (also called *machine epsilon*) $0 < \epsilon_{\mathsf{mach}} < 1$, satisfying the following properties:

(a) For any $x \in \mathbb{F}$, $\mathsf{round}(x) = x$. In particular $\mathsf{round}(0) = 0$.
(b) For any $x \in \mathbb{R}$, $\mathsf{round}(x) = x(1+\delta)$ with $|\delta| \le \epsilon_{\mathsf{mach}}$.

Furthermore, one can take $\epsilon_{\mathsf{mach}} = \frac{\beta^{-t+1}}{2}$ and therefore $|\log_\beta \epsilon_{\mathsf{mach}}| = t - \log_\beta \frac{\beta}{2}$.

Arithmetic operations on $\mathbb{F}$ are defined following the scheme

$$x \mathbin{\tilde{\circ}} y = \mathsf{round}(x \circ y)$$

for any $x, y \in \mathbb{F}$ and $\circ \in \{+, -, \times, /\}$ and therefore

$$\tilde{\circ} : \mathbb{F} \times \mathbb{F} \to \mathbb{F}.$$

It follows from (b) above that for any $x, y \in \mathbb{F}$ we have

$$x \mathbin{\tilde{\circ}} y = (x \circ y)(1+\delta), \quad |\delta| \le \epsilon_{\mathsf{mach}}.$$

Other operations may also be considered. Thus, a floating-point version $\tilde{\sqrt{\ }}$ of the square root would similarly satisfy

$$\tilde{\sqrt{x}} = \sqrt{x}(1+\delta), \quad |\delta| \le \epsilon_{\mathsf{mach}}.$$

When combining many operations in floating-point arithmetic, expressions such as $(1+\delta)$ above naturally appear. To simplify round-off analyses it is useful to consider the quantities, for $k \ge 1$ and $k\epsilon_{\mathsf{mach}} < 1$,

$$\gamma_k := \frac{k\epsilon_{\mathsf{mach}}}{1 - k\epsilon_{\mathsf{mach}}} \tag{O.5}$$

and to denote by θ_k *any* number satisfying $|\theta_k| \le \gamma_k$. In this sense, θ_k represents a set of numbers, and different occurrences of θ_k in a proof may denote different numbers. Note that

$$\gamma_k \le (k+1)\epsilon_{\mathsf{mach}} \quad \text{if } k(k+1) \le \epsilon_{\mathsf{mach}}^{-1}. \tag{O.6}$$

The proof of the following proposition can be found in Chap. 3 of [121].

Proposition O.1 *The following relations hold* (*assuming all quantities are well defined*):

[3]This notion reflects the intuitive idea of significant figures modulo carry differences. The number 0.9999 approximates 1 with a precision $t = 10^{-4}$. Yet their first significant digits are different.

(a) $(1+\theta_k)(1+\theta_j) = 1+\theta_{k+j}$,
(b)

$$\frac{1+\theta_k}{1+\theta_j} = \begin{cases} 1+\theta_{k+j} & \textit{if } j \le k, \\ 1+\theta_{k+2j} & \textit{if } j > k, \end{cases}$$

(c) $\gamma_k\gamma_j \le \gamma_{\min\{k,j\}}$ *if* $\max\{k\epsilon_{\mathsf{mach}}, j\epsilon_{\mathsf{mach}}\} \le 1/2$,
(d) $i\gamma_k \le \gamma_{ik}$,
(e) $\gamma_k + \epsilon_{\mathsf{mach}} \le \gamma_{k+1}$,
(f) $\gamma_k + \gamma_j + \gamma_k\gamma_j \le \gamma_{k+j}$. □

O.3.2 . . . and the Way We Lose It

In computing an arithmetic expression q with a round-off algorithm, errors will accumulate, and we will obtain another quantity, which we denote by $\mathsf{fl}(q)$. We will also write $\mathsf{Error}(q) = |q - \mathsf{fl}(q)|$, so that $\mathsf{RelError}(q) = \frac{\mathsf{Error}(q)}{|q|}$.

Assume now that q is computed with a real-number algorithm $\mathcal{A}$ executed using floating-point arithmetic from data a (a formal model for real-number algorithms was given in [37]). No matter how precise the representation we are given of the entries of a, these entries will be rounded to t digits. Hence t (or, being roughly the same, $|\log_\beta \epsilon_{\mathsf{mach}}|$) is the precision of our data. On the other hand, the number of correct digits in $\mathsf{fl}(q)$ is approximately $-\log_\beta \mathsf{RelError}(q)$. Therefore, the value

$$\mathsf{LoP}(q) := \log_\beta \frac{\mathsf{RelError}(q)}{\epsilon_{\mathsf{mach}}} = |\log_\beta \epsilon_{\mathsf{mach}}| - \left|\log_\beta \mathsf{RelError}(q)\right|$$

quantifies the *loss of precision* in the computation of q. To extend this notion to the computation of vectors $v = (v_1, \dots, v_q) \in \mathbb{R}^q$, we need to fix a measure for the precision of the computed $\mathsf{fl}(e) = (\mathsf{fl}(v_1), \dots, \mathsf{fl}(v_q))$: componentwise or normwise.

In the componentwise case, we have

$$-\log_\beta \mathsf{RelError}(e) = -\log_\beta \max_{i \le q} \frac{|\mathsf{fl}(v_i) - v_i|}{|v_i|} = \min_{i \le q}\left(-\log_\beta \frac{|\mathsf{fl}(v_i) - v_i|}{|v_i|}\right),$$

so that the precision of v is the smallest of the precisions of its components.

For the normwise measure, we take the precision of v to be

$$-\log_\beta \mathsf{RelError}(e) = -\log_\beta \frac{\|\mathsf{fl}(e) - v\|}{\|v\|}.$$

This choice has both the pros and cons of viewing v as a whole and not as the aggregation of its components.

For both the componentwise and the normwise measures we can consider ϵ_{mach} as a measure of the worst possible relative error $\mathsf{RelError}(a)$ when we read data a with round-off unit ϵ_{mach}, since in both cases

$$\max_{|\tilde{a}_i - a_i| \le \epsilon_{\mathsf{mach}} |a_i|} \mathsf{RelError}(a) = \epsilon_{\mathsf{mach}}.$$

Hence, $|\log_\beta \epsilon_{\mathsf{mach}}|$ represents in both cases the precision of the data. We therefore define the loss of precision in the computation of $\varphi(a)$ to be

$$\mathsf{LoP}\big(\varphi(a)\big) := \log_\beta \frac{\mathsf{RelError}(\varphi(a))}{\epsilon_{\mathsf{mach}}} = |\log_\beta \epsilon_{\mathsf{mach}}| + \log_\beta \mathsf{RelError}\big(\varphi(a)\big). \tag{O.7}$$

Remark O.2 By associating $\mathsf{RelError}(a) \approx \epsilon_{\mathsf{mach}}$, we may view the logarithm of a condition number $\log_\beta \mathsf{cond}^\varphi(a)$ as a measure of the worst possible loss of precision in a computation of $\varphi(a)$ in which the only error occurs in reading the data.

To close this section we prove a result putting together—and making precise—a number of issues dealt with so far. For data $a \in \mathcal{D} \subseteq \mathbb{R}^m$ we call m the *size* of a and we write $\mathsf{size}(a) = m$. Occasionally, this size is a function of a few integers, the *dimensions* of a, the set of which we denote by $\mathsf{dims}(a)$. For instance, a $p \times q$ matrix has dimensions p and q and size pq.

Theorem O.3 *Let $\mathcal{A}^\varphi$ be a finite-precision algorithm with round-off unit ϵ_{mach} computing a function $\varphi : \mathcal{D} \subseteq \mathbb{R}^m \to \mathbb{R}^q$. Assume $\mathcal{A}^\varphi$ satisfies the following backward bound: for all $a \in \mathcal{D}$ there exists $\tilde{a} \in \mathcal{D}$ such that*

$$\mathcal{A}^\varphi(a) = \varphi(\tilde{a})$$

and

$$\mathsf{RelError}(a) \le f\big(\mathsf{dims}(a)\big)\epsilon_{\mathsf{mach}} + o(\epsilon_{\mathsf{mach}})$$

for some positive function f, and where the "little oh" is for $\epsilon_{\mathsf{mach}} \to 0$. Then the computed $\mathcal{A}^\varphi(a)$ satisfies the forward bound

$$\mathsf{RelError}\big(\varphi(a)\big) \le f\big(\mathsf{dims}(a)\big)\mathsf{cond}^\varphi(a)\epsilon_{\mathsf{mach}} + o(\epsilon_{\mathsf{mach}}),$$

and the loss of precision in the computation (in base β) is bounded as

$$\mathsf{LoP}\big(\varphi(a)\big) \le \log_\beta f\big(\mathsf{dims}(a)\big) + \log_\beta \mathsf{cond}^\varphi(a) + o(1).$$

Here cond^φ refers to the condition number defined in (O.1) with the same measures (normwise or componentwise) for $\mathsf{RelError}(a)$ and $\mathsf{RelError}(\varphi(a))$ as those in the backward and forward bounds above, respectively.

Proof The forward bound immediately follows from the backward bound and (O.3). For the loss of precision we have

$$\begin{aligned}\log_\beta \mathsf{RelError}\big(\varphi(a)\big) &\le \log_\beta f\big(\mathsf{dims}(a)\big)\mathsf{cond}^\varphi(a)\epsilon_{\mathsf{mach}}\big(1 + o(1)\big) \\ &\le \log_\beta f\big(\mathsf{dims}(a)\big) + \log_\beta \mathsf{cond}^\varphi(a) - |\log_\beta \epsilon_{\mathsf{mach}}| + o(1),\end{aligned}$$

from which the statement follows. □

O.4 An Example: Matrix–Vector Multiplication

It is perhaps time to illustrate the notions introduced so far by analyzing a simple problem, namely, matrix–vector multiplication. We begin with a (componentwise) backward stability analysis.

Proposition O.4 *There is a finite-precision algorithm $\mathscr{A}$ that with input $A \in \mathbb{R}^{m\times n}$ and $x \in \mathbb{R}^n$, computes the product Ax. If $\epsilon_{\mathsf{mach}}(\lceil \log_2 n\rceil + 2)^2 < 1$, then the computed vector $\mathsf{fl}(Ax)$ satisfies $\mathsf{fl}(Ax) = \tilde{A}x$ with*

$$|\tilde{a}_{ij} - a_{ij}| \le \big(\lceil \log_2 n\rceil + 2\big)\epsilon_{\mathsf{mach}}|a_{ij}|.$$

Proof Let $b = Ax$. For $i = 1, \dots, m$ we have

$$b_i = a_{i1}x_1 + a_{i2}x_2 + \cdots + a_{in}x_n.$$

For the first product on the right-hand side we have $\mathsf{fl}(a_{i1}x_1) = a_{i1}x_1(1+\delta)$ with $|\delta| \le \epsilon_{\mathsf{mach}} \le \frac{\epsilon_{\mathsf{mach}}}{1-\epsilon_{\mathsf{mach}}} = \gamma_1$. That is, $\mathsf{fl}(a_{i1}x_1) = a_{i1}x_1(1+\theta_1)$ and similarly $\mathsf{fl}(a_{i2}x_2) = a_{i2}x_2(1+\theta_1)$. Note that the two occurrences of θ_1 here denote two different quantities. Hence, using Proposition O.1,

$$\begin{aligned}\mathsf{fl}(a_{i1}x_1 + a_{i2}x_2) &= \big(a_{i1}x_1(1+\theta_1) + a_{i2}x_2(1+\theta_1)\big)(1+\theta_1)\\ &= a_{i1}x_1(1+\theta_2) + a_{i2}x_2(1+\theta_2).\end{aligned}$$

By the same reasoning, $\mathsf{fl}(a_{i3}x_3 + a_{i4}x_4) = a_{i3}x_3(1+\theta_2) + a_{i4}x_4(1+\theta_2)$, and therefore

$$\begin{aligned}&\mathsf{fl}(a_{i1}x_1 + a_{i2}x_2 + a_{i3}x_3 + a_{i4}x_4)\\ &\quad= \big(a_{i1}x_1(1+\theta_2) + a_{i2}x_2(1+\theta_2) + a_{i3}x_3(1+\theta_2) + a_{i4}x_4(1+\theta_2)\big)(1+\theta_1)\\ &\quad= a_{i1}x_1(1+\theta_3) + a_{i2}x_2(1+\theta_3) + a_{i3}x_3(1+\theta_3) + a_{i4}x_4(1+\theta_3).\end{aligned}$$

Continuing in this way, we obtain

$$\mathsf{fl}(b_i) = \tilde{a}_{i1}x_1 + \tilde{a}_{i2}x_2 + \cdots + \tilde{a}_{in}x_n$$

with $\tilde{a}_{ij} = a_{ij}(1+\theta_{\lceil \log_2 n\rceil + 1})$. The result follows from the estimate (O.6), setting $k = \lceil \log_2 n\rceil + 1$. □

Remark O.5 Note that the algorithm computing Ax is implicitly given in the proof of Proposition O.4. This algorithm uses a balanced treelike structure for the sums. The order of the sums cannot be arbitrarily altered: the operations $\tilde{+}$ and $\tilde{\cdot}$ are nonassociative.

We next estimate the componentwise condition number of matrix–vector multiplication. In doing so, we note that in the backward analysis of Proposition O.4, only the entries of A are perturbed. Those of x are not. This feature allows one to consider the condition of data (A,x) for perturbations of A only. Such a situation is common and also arises when data are structured (e.g., unit upper-triangular matrices have zeros below the diagonal and ones on the diagonal) or contain entries that are known to be integers.

Proposition O.6 *The componentwise condition numbers* $\mathsf{Cw}_i(A,x)$ *of matrix–vector multiplication, for perturbations of A only, satisfy*

$$\mathsf{Cw}_i(A,x) \le \left|\sec(a_i,x)\right|,$$

where a_i denotes the ith row of A and $\sec(a_i,x)=\frac{1}{\cos(a_i,x)}$ *denotes the secant of the angle it makes with x (we assume $a_i, x \neq 0$).*

Proof Let $\tilde{A}=A+E$ be a perturbation of A with $E=(e_{ij})$. By definition, $|e_{ij}| \le \mathsf{RelError}(A)|a_{ij}|$ for all i,j, whence $\|e_i\| \le \mathsf{RelError}(A)\|a_i\|$ for all i (here $\|\ \|$ denotes the Euclidean norm in $\mathbb{R}^n$). We obtain

$$\mathsf{RelError}\big((Ax)_i\big) = \frac{|e_i^{\mathrm{T}}x|}{|a_i^{\mathrm{T}}x|} \le \frac{\|e_i\|\|x\|}{|a_i^{\mathrm{T}}x|} \le \mathsf{RelError}(A)\frac{\|a_i\|\|x\|}{|a_i^{\mathrm{T}}x|}.$$

This implies that

$$\begin{aligned}\mathsf{Cw}_i(A,x) &= \lim_{\delta\to 0}\sup_{\mathsf{RelError}(A)\le\delta}\frac{\mathsf{RelError}((Ax)_i)}{\mathsf{RelError}(A)}\\ &\le \frac{\|a_i\|\|x\|}{|a_i^{\mathrm{T}}x|} = \frac{1}{|\cos(a_i,x)|} = \left|\sec(a_i,x)\right|.\end{aligned}$$ □

A bound for the loss of precision in the componentwise context follows.

Corollary O.7 *In the componentwise setting, for all i such that $b_i=(Ax)_i\neq 0$,*

$$\mathsf{RelError}(b_i) \le \left|\sec(a_i,x)\right|\big(\lceil\log_2 n\rceil+2\big)\,\epsilon_{\mathsf{mach}} + o(\epsilon_{\mathsf{mach}}),$$

$$\mathsf{LoP}(b_i) \le \log_\beta\left|\sec(a_i,x)\right| + \log_\beta\big(\lceil\log_2 n\rceil+2\big) + o(1),$$

provided $\log_2 n \le \epsilon_{\mathsf{mach}}^{-1/2}+3$.

Proof Immediate from Propositions O.4 and O.6 and Theorem O.3. □

The corollary above states that if we are working with $|\log_\beta \epsilon_{\mathsf{mach}}|$ bits of precision, we compute a vector $\mathsf{fl}(Ax)$ whose nonzero entries have, approximately, at least

$$|\log_\beta \epsilon_{\mathsf{mach}}| - \log_\beta\left|\sec(a_i,x)\right| - \log_\beta \log_2 n$$

bits of precision. (The required bound on n is extremely weak and will be satisfied in all cases of interest.) This is a satisfying result. One may, nevertheless, wonder about the (absolute) error for the zero components of Ax. In this case, a normwise analysis may be more appropriate.

To proceed with a normwise analysis we first need to choose a norm in the space of $m \times n$ matrices. For simplicity, we choose

$$\|A\|_\infty = \max_{\|x\|_\infty=1} \|Ax\|_\infty.$$

It is well known that

$$\|A\|_\infty = \max_{i\le n} \|a_i\|_1. \tag{O.8}$$

Now note that it follows from Proposition O.4 that the perturbation $\tilde{A}$ in its statement satisfies, for n not too large,

$$\|\tilde{A} - A\|_\infty \le \big(\lceil \log_2 n\rceil + 2\big)\,\epsilon_{\mathsf{mach}}. \tag{O.9}$$

Therefore, we do have a normwise backward-error analysis. In addition, a normwise version of Proposition O.6 can be easily obtained.

Proposition O.8 *The normwise condition number* $\mathsf{cond}(A,x)$ *of matrix–vector multiplication, for perturbations on A only, satisfies, for $Ax \ne 0$,*

$$\mathsf{cond}(A,x) = \frac{\|A\|_\infty \|x\|_\infty}{\|Ax\|_\infty}.$$

Proof We have

$$\begin{aligned}
\mathsf{cond}(A,x) &= \lim_{\delta\to 0} \sup_{\mathsf{RelError}(A)\le\delta} \frac{\mathsf{RelError}(Ax)}{\mathsf{RelError}(A)} \\
&= \lim_{\delta\to 0} \sup_{\|\tilde{A}-A\|_\infty\le\delta\|A\|_\infty} \frac{\|\tilde{A}x - Ax\|_\infty}{\|Ax\|_\infty} \frac{\|A\|_\infty}{\|\tilde{A} - A\|_\infty} \\
&\le \frac{\|A\|_\infty \|x\|_\infty}{\|Ax\|_\infty}.
\end{aligned}$$

Actually, equality holds. In order to see this, assume, without loss of generality, that $\|x\|_\infty = |x_1|$. Set $\tilde{A} = A + E$, where $e_{11} = \delta$ and $e_{ij} = 0$ otherwise. Then we have $\|\tilde{A}x - Ax\|_\infty = \|Ex\|_\infty = \delta|x_1| = \|E\|_\infty\|x\|_\infty = \|\tilde{A} - A\|_\infty\|x\|_\infty$. □

Again, a bound for the loss of precision immediately follows.

Corollary O.9 *In the normwise setting, when $Ax \ne 0$,*

$$\mathsf{LoP}(Ax) \le \log_\beta\left(\frac{\|A\|_\infty\|x\|_\infty}{\|Ax\|_\infty}\right) + \log_\beta\big(\lceil \log_2 n\rceil + 2\big) + o(1),$$

provided $\log_2 n \le \epsilon_{\mathsf{mach}}^{-1/2} + 3$.

Proof It is an immediate consequence of (O.9), Proposition O.8, and Theorem O.3. □

Remark O.10 If $m = n$ and A is invertible, it is possible to give a bound on the normwise condition that is independent of x. Using that $x = A^{-1}Ax$, we deduce $\|x\|_\infty \leq \|A^{-1}\|_\infty \|Ax\|_\infty$ and therefore, by Proposition O.8, $\mathsf{cond}(A, x) \leq \|A^{-1}\|_\infty \|A\|_\infty$. A number of readers may find this expression familiar.

O.5 The Many Faces of Condition

The previous sections attempted to introduce condition numbers by retracing the way these numbers were introduced: as a way of measuring the effect of data perturbations. The expression "condition number" was first used by Turing [221] to denote a condition number for linear equation solving, independently introduced by him and by von Neumann and Goldstine [226] in the late 1940s. Expressions like "ill-conditioned set [of equations]" to denote systems with a large condition number were also introduced in [221].

Conditioning, however, was eventually related to issues in computation other than error-propagation analysis and this fact—together with the original role of conditioning in error analysis—triggered research on different aspects of the subject. We briefly describe some of them in what follows.

O.5.1 Condition and Complexity

In contrast with direct methods (such as Gaussian elimination), the number of times that a certain basic procedure is repeated in iterative methods is not data-independent. In the analysis of this dependence on the data at hand it was early realized that, quite often, one could express it using its condition number. That is, the number of iterations the algorithm $\mathcal{A}^\varphi$ would perform with data $a \in \mathbb{R}^m$ could be bounded by a function of m, $\mathsf{cond}^\varphi(a)$, and—in the case of an algorithm computing an ε-approximation of the desired solution—the accuracy ε. A very satisfying bound for the number of iterations $\#\,\mathsf{iterations}(\mathcal{A}^\varphi(a))$ of algorithm $\mathcal{A}^\varphi$ would have the form

$$\#\,\mathsf{iterations}\big(\mathcal{A}^\varphi(a)\big) \leq \left(m + \log \mathsf{cond}^\varphi(a) + \log\left(\frac{1}{\varepsilon}\right)\right)^{\mathcal{O}(a)}, \tag{O.10}$$

and a less satisfying (but often still acceptable) bound would have $\log \mathsf{cond}^\varphi(a)$ replaced by $\mathsf{cond}^\varphi(a)$ and/or $\log(\frac{1}{\varepsilon})$ replaced by $\frac{1}{\varepsilon}$. We will encounter several instances of this *condition-based complexity analysis* in the coming chapters.

O.5.2 Computing Condition Numbers

Irrespective of whether relative errors are measured normwise or componentwise, the expression (O.1) defining the condition number of a (for the problem φ) is hardly usable. Not surprisingly then, one of the main lines of research regarding condition numbers has focused on finding equivalent expressions for $\mathsf{cond}^{\varphi}(a)$ that would be directly computable or, if this appears to be out of reach, tight enough bounds with this property. We have done so for the problem of matrix–vector multiplication in Propositions O.6 and O.8 (for the componentwise and normwise cases, respectively). In fact, in many examples the condition number can be succinctly expressed in terms of the norm of a derivative, which facilitates its analysis (cf. Sect. 14.1).

O.5.3 Condition of Random Data

How many iterations does an iterative algorithm need to perform to compute $\varphi(a)$? To answer this question we need $\mathsf{cond}^{\varphi}(a)$. And to compute $\mathsf{cond}^{\varphi}(a)$ we would like a simple expression like those in Propositions O.6 and O.8. A second look at these expressions, however, shows that they seem to require $\varphi(a)$, the quantity in which we were interested in the first place. For in the componentwise case, we need to compute $\sec(a_i, x)$—and hence $a_i^{\mathrm{T}}x$—for $i = 1, \ldots, n$, and in the normwise case the expression $\|Ax\|_\infty$ speaks for itself. Worst of all, this is not an isolated situation. We will see that the condition number of a matrix A with respect to matrix inversion is expressed in terms of A^{-1} (or some norm of this inverse) and that a similar phenomenon occurs for each of the problems we consider. So, even though we do not formalize this situation as a mathematical statement, we can informally describe it by saying that the computation of a condition number $\mathsf{cond}^{\varphi}(a)$ is never easier than the computation of $\varphi(a)$. The most elaborate reasoning around this issue was done by Renegar [164].

A similar problem appears with perturbation considerations. If we are given only a perturbation $\tilde{a}$ of data a, how can we know how accurate $\varphi(\tilde{a})$ is? Even assuming that we can compute cond^{φ} accurately and fast, the most we could do is to compute $\mathsf{cond}^{\varphi}(\tilde{a})$, not $\mathsf{cond}^{\varphi}(a)$.

There are a number of ways in which this seemingly circular situation can be broken. Instead of attempting to make a list of them (an exercise that can only result in boredom), we next describe a way out pioneered by John von Neumann (e.g., in [108]) and strongly advocated by Steve Smale in [201]. It consists in randomizing the data (i.e., in assuming a probabilistic distribution $\mathscr{D}$ in $\mathbb{R}^m$) and considering the tail

$$\mathop{\mathsf{Prob}}_{a\sim\mathscr{D}}\bigl\{\mathsf{cond}^{\varphi}(a) \geq t\bigr\}$$

or the expected value (for $q \geq 1$)

$$\mathop{\mathbb{E}}_{a\sim\mathscr{D}}\bigl(\log^q \mathsf{cond}^{\varphi}(a)\bigr).$$

The former, together with a bound as in (O.10), would allow one to bound the probability that $\mathscr{A}^\varphi$ needs more than a given number of iterations. The latter, taking q to be the constant in the $\mathcal{O}(a)$ notation, would make it possible to estimate the expected number of iterations. Furthermore, the latter again, now with $q=1$, can be used to obtain an estimate of the average loss of precision for a problem φ (together with a backward stable algorithm $\mathscr{A}^\varphi$ if we are working with finite-precision arithmetic).

For instance, for the example that formed the substance of Sect. O.4, we will prove for a matrix $A \in \mathbb{R}^{m\times n}$ with standard Gaussian entries that

$$\mathbb{E}\big(\log_\beta \mathsf{Cw}_i(A)\big) \le \frac{1}{2}\log_\beta n + 2.$$

In light of Corollary O.7, this bound implies that the expected loss of precision in the computation of $(Ax)_i$ is at most $\frac{1}{2}\log_\beta n + \log_\beta \log_2 n + \mathcal{O}(1)$.

The probabilistic analysis proposed by von Neumann and Smale relies on the assumption of "evenly spread random data." A different approach was recently proposed that relies instead on the assumption of "nonrandom data affected by random noise." We will develop both approaches in this book.

O.5.4 Ill-posedness and Condition

Let us return once more to the example of matrix–vector multiplication. If A and x are such that $Ax=0$, then the denominator in $\frac{\|A\|_\infty\|x\|_\infty}{\|Ax\|_\infty}$ is zero, and we can define $\mathsf{cond}(A,x)=\infty$. This reflects the fact that no matter how small the absolute error in computing Ax, the relative error will be infinite. The quest for any relative precision is, in this case, a battle lost in advance. It is only fair to refer to instances like this with a name that betrays this hopelessness. We say that a is *ill-posed for* φ when $\mathsf{cond}^\varphi(a)=\infty$. Again, one omits the reference to φ when the problem is clear from the context, but it goes without saying that the notion of ill-posedness, like that of condition, is with respect to a problem. It also depends on the way we measure errors. For instance, in our example, $\mathsf{Cw}(A,x)=\infty$ if and only if there exists $i \le n$ such that $a_i^{\mathrm{T}}x=0$, while for $\mathsf{cond}(A,x)$ to be infinity, it is necessary (and sufficient) that $Ax=0$.

The subset of $\mathbb{R}^m$ of ill-posed inputs is denoted by Σ^φ (or simply by Σ), and it has played a distinguished role in many developments in conditioning. To see why, let us return (yes, once again) to matrix–vector multiplication, say in the componentwise setting. Recall that we are considering x as fixed (i.e., not subject to perturbations). In this situation we take $\Sigma \subset \mathbb{R}^{n\times m}$ to be the set of matrices A such that $\mathsf{Cw}(A,x)=\infty$. We have $\Sigma = \bigcup_{i\le n}\Sigma_i$ with

$$\Sigma_i = \big\{A \in \mathbb{R}^{n\times m} \mid \mathsf{Cw}_i(A,x)=\infty\big\} = \big\{A\in\mathbb{R}^{n\times m} \mid a_i^{\mathrm{T}}x=0\big\}.$$

Now recall $\mathsf{Cw}_i(A,x) \le \frac{1}{|\cos(a_i,x)|}$. If we denote by $\bar{a}_i$ the orthogonal projection of a_i on the space $x^\perp = \{y \in \mathbb{R}^m \mid y^{\mathrm{T}}x = 0\}$, then

$$\frac{1}{|\cos(a_i,x)|} = \frac{\|a_i\|}{\|a_i - \bar{a}_i\|},$$

and it follows that

$$\mathsf{Cw}_i(A,x) \le \frac{\|a_i\|}{\mathsf{dist}(A, \Sigma_i)}. \tag{O.11}$$

That is, componentwise, the condition number of (A,x) is bounded by the inverse of the relativized distance from A to ill-posedness.

This is not an isolated phenomenon. On the contrary, it is a common occurrence that condition numbers can be expressed as, or at least bounded by, the inverse of a relativized distance to ill-posedness. We will actually meet this theme repeatedly in this book.