Statistical estimation of roundoff errors and condition numbers

Jocelyne Erhel

UNITÉ DE RECHERCHE
INRIA-RENNES

Institut National
de Recherche
en Informatique
et en Automatique

Domaine de Voluceau
Rocquencourt
B.P. 105
78153 Le Chesnay Cedex
France
Tél.:(1) 39 63 55 11

Rapports de Recherche

N° 1490

Programme 1
Architectures parallèles, Bases de données,
Réseaux et Systèmes distribués

STATISTICAL ESTIMATION OF ROUNDOFF ERRORS AND CONDITION NUMBERS

Jocelyne ERHEL

Septembre 1991

IRISA

INSTITUT DE RECHERCHE EN INFORMATIQUE
ET SYSTEMES ALEATOIRES

Campus Universitaire de Beaulieu
35042 - RENNES CEDEX FRANCE
Tél. : 99 84 71 00 - Télex : UNIRISA 950 473 F
Télécopie : 99 38 38 32

Statistical Estimation of Roundoff Errors and Condition Numbers

Estimation Statistique d'Erreurs d'Arrondi et de Conditionnements

Jocelyne Erhel
IRISA/INRIA, Campus de Beaulieu, 35042 RENNES CEDEX, FRANCE

Publication Interne n°599 - Septembre 1991 - 34 Pages

Abstract

Roundoff errors combine with ill-conditioning to provide inaccurate results in scientific applications. Random perturbations in the initial data allows to derive an error estimation. Moreover, by varying these perturbations, it is possible to distinguish between numerical and problem instabilities. Our statistical method relies on a domain of regularity, where a log-linear regression of the error estimation upon the perturbations is valid. This regression gives an estimation of the regularity and the condition number of the problem.

Résumé

Les erreurs d'arrondi s'ajoutent au mauvais conditionnement pour fournir des résultats imprécis en calcul scientifique. Des perturbations aléatoires des données initiales permettent d'obtenir une estimation de l'erreur. De plus, en faisant varier ces perturbations, il est possible de faire la distinction entre les instabilités numériques et celles dues au problème. Notre méthode statistique repose sur un domaine de régularité, où une régression log-linéaire des estimations d'erreur sur les perturbations est valide. Cette régression fournit une estimation de la régularité et du conditionnement du problème.

CENTRE NATIONAL DE LA RECHERCHE SCIENTIFIQUE (URA 227) UNIVERSITE DE RENNES I INSA DE RENNES
INSTITUT NATIONAL DE RECHERCHE EN INFORMATIQUE ET EN AUTOMATIQUE (UNITE DE RECHERCHE DE RENNES)

Contents

1 Introduction

Numerical simulation plays an important role to solve physical problems. With the advent of very fast supercomputers, more and more complex situations can be modelized. Though the accuracy of the results is crucial, this area has not yet received a great attention and few tools exist to control or improve numerical quality. Moreover existing tools are difficult to use.

Inaccuracies arise on one hand from imprecise initial data and on the other hand from roundoff errors. The sensitivity to initial data depends on the problem and is measured by its regularity and its condition number, using perturbation theory. Even in exact arithmetic, small errors in the initial data lead to inaccurate results for ill-posed or ill-conditioned problems. The other source of inaccuracy is the finite precision arithmetic. The sensitivity to roundoff errors depends on the algorithm and is measured by forward or backward analysis. If the algorithm is unstable, it amplifies the roundoff errors and delivers an inaccurate result.

Deterministic methods have been designed to estimate the stability of algorithms and problems [16, 15]. They are based on a computational graph representing the program. First-order local errors are computed to finally express the global error as a linear combination of these errors, separating rounding effects from errors in initial data. This approach is interesting but seems to be very expensive. Moreover, it does not take into account second-order effects. Tools to use it easily have been recently designed [3], and are currently experimented.

Another idea which is not new is to use probabilities or statistics [19]. The first approach is to simulate rounding errors by perturbing randomly all the floating-point operations, to execute several times the program and then to derive from the samples of results an estimation of the accuracy of the result. The software CESTAC implements this technique [7]. Though it is useful in practice, it has several drawbacks : first, it assumes more or less that the rounding errors are independent giving up any possible correlation between them. Then it cannot make the distinction between algorithm and problem stability. Finally, it is quite expensive because it requires a new arithmetic, emulated by software in general. The software PERTURBE [4] is similar but it solves the inverse problem and analyzes the samples obtained. This technique, based on backward analysis, allows in general to control the stability of the algorithm and then to estimate the condition number of the problem. However, it is not always possible to solve almost exactly the inverse problem. Moreover, it still requires assumptions on the

algorithm and remains expensive.

These observations led to simulate only perturbations in the initial data, keeping the possible correlation between the roundoff errors, making no assumption, implementing these perturbations very easily, and using them at a low cost. However small perturbations (for example when few digits are perturbed) may lead to false conclusions as quoted in [11]. Therefore, several sizes of perturbations must be used to obtain meaningful estimations, increasing of course the cost of execution. This is done in the software SCALP [10] along with a statistical analysis to obtain regularity and condition number estimations. However, this analysis is not valid for unstable algorithms. Varying sizes of perturbations are also used by [5] along with inverse problem solving as in [4] to estimate condition numbers in linear algebra.

This paper presents a new methodology based on SCALP. The idea is to separate numerical instability from problem conditioning, using the following observation. For small perturbations in the initial data, roundoff errors are predominent, whereas for sufficiently large perturbations they become negligible. The perturbations must of course stay small enough to allow perturbation theory. This defines a domain of perturbations where the method SCALP is valid and allows to derive sharp estimations of problem conditioning and error estimation. Moreover, it demonstrates numerical instabilities when they exist. This methodology applies currently to well-posed problems but can detect ill-conditioning.

We used our method in linear algebra and for a simple non linear problem, namely a second-order equation. The results are very promising, proving at least for these examples the capabilities and usefulness of this approach.

The paper is organized as follows. The second section defines precisely the notions of problem and numerical stability. The third section describes in details our methodology. The fourth section contains the numerical experiments, with the obtained results. The last section deals with practical aspects.

2 Stability, Condition Number, Error Analysis

The goal of this section is to define precisely what kind of problem and algorithms are concerned and to recall the notions of problem stability and numerical error analysis.

2.1 Problem

Roughly speaking, a problem consists in finding a solution which depends on initial data. To describe it formally, we have to define the input and output sets. Let D and X be subsets of finite dimension spaces (this hypothesis will be used later on) built on the real or complex field. We suppose that some norms are defined on each space. For sake of simplicity, these two norms will both be denoted by $\|.\|$.

Definition 2.1 *Given a function F from $X \times D$ to the real or complex field, a problem (P) for the initial data d consists in finding a solution x which satisfies $F(x,d) = 0$.*

We want to study the problem P in a neighborhood V of some initial data d_0.

A problem is first characterized by the number of solutions and by the continuity of the solutions.

Definition 2.2 *A problem is well-posed in V if it has a unique solution, denoted by $G(d)$ and if this solution is continuous in V.*

This notion depends on the choice of the input and output sets, as illustrated by the following example :

Example 2.1 *The problem expressed as "find $x \in \mathcal{R}$ solution of the equation $x^2 - a\ x + b = 0$" is ill-posed near a double solution. However, the problem expressed as "find the set of solutions of the equation $x^2 - a\ x + b = 0$" is in turn well-posed.*

2.2 Stability, Regularity and Condition Number

An important characterization of a problem is its sensitivy to perturbations in the initial data. A well-posed problem is stable if small perturbations $\|\Delta\ d\|$ in the data induce small errors $\|\Delta\ x\|$ in the solution. For many problems, the errors are of the same order than the perturbations. The sensitivy to the perturbations in the data is then measured by the condition number, which is roughly the amplification factor $\|\Delta\ d\|/\|\Delta\ x\|$. These notions are well-known [20] but are extended here, following [10], by introducing the notion of regularity. For some problems, the errors are not of the same order than the perturbations. The largest order is called the regularity q, and the corresponding amplification factor is the q-condition

number. For example, a regularity of $1/n$ has been defined by [6] to study the perturbations of eigenproblems. These notions are formally expressed below :

Definition 2.3 *A well-posed problem defined by* $F(x,d) = 0 \Leftrightarrow x = G(d)$ *is q-stable at* d_0 *if there exists a constant* K *such that*

$$\forall d \in V \quad \|x - x_0\| \leq K \ \|d - d_0\|^q$$

where x *and* x_0 *design respectively the solutions at the data* d *and* d_0.

The largest q such that a problem is q-stable at d_0 *is called the regularity of the problem at* d_0.

If a problem is q-stable, the absolute q-condition number at d_0 *is defined by*

$$C_a = \limsup_{d \in V \text{ and } d \neq d_0} \frac{\|x - x_0\|}{(\|d - d_0\|)^q}$$

If a problem is q-stable and if $d_0 \neq 0$ *and if* $x_0 \neq 0$, *the relative q-condition number at* d_0 *is defined by*

$$C = \limsup_{d \in V \text{ and } d \neq d_0} \frac{\|x - x_0\|/\|x_0\|}{(\|d - d_0\|/\|d_0\|)^q}$$

It should be noted that the condition number depends on the choice of the norms. However, the regularity is independent of the norms because we deal with finite dimension spaces where all the norms are equivalent.

A regularity of one corresponds to the usual definition of stable problems and condition numbers. This is the case for problems defined by a function G of class C^1 for example. A regularity smaller than one corresponds to ill-conditioned problems in the usual terms, but with some weaker regularity.

Proposition 2.1 *Let a problem defined by* $F(x,d) = 0$. *If* F *is of class* C^1 *with a non-singular partial derivative* DF/Dx *and a non-zero partial derivative* DF/Dd, *then the problem is well-posed and defined by* $x = G(d)$ *with* G *of class* C^1. *Further, its regularity is equal to 1 and the condition number at* d_0 *can be approximated by* $\|DG(d_0)\|$.

Proof. The problem is of course well-posed. With our assumptions, we can apply the theorem of implicit functions thus determining a function G of class C^1. Then we apply the differentiation of G at d_0 to get $x - x_0 = DG(d_0).(d - d_0) + \epsilon(\|d - d_0\|)$ so that at the first order we obtain $\|x - x_0\| \leq \|DG(d_0)\| \, \|d - d_0\|$, showing that the problem is 1-regular and that $\|DG(d_0)\|$ is at the first order the condition number. □

Example 2.2 *The data in the resolution of a linear solver are the matrix A and the right-hand side b, with the solution x satisfying $A\,x - b = 0$. If A is non singular, the problem is 1-regular at (A, b) with, for example, a relative condition number equal to $\|A\| \, \|A^{-1}\|$. This example is studied in more details in section 4.2.*

Proposition 2.2 *Let a well-posed problem defined by $x = G(d)$, with the function G Holderienne of class C^q for some $q < 1$. Then the problem is q-regular.*

Proof. It follows directly from the definition of the class C^q. □

Example 2.3 *To find the set of solutions of a second-order equation is a problem of regularity 0.5 at a double solution.*

2.3 Algorithm

A problem is solved by using an algorithm, which may be direct or iterative. We restrict the study to direct algorithms, composed of a finite number of arithmetic operations. We do not deal with errors due to approximation methods. In other words, if the algorithm is executed with infinite precision, it gives the exact solution of the problem.

More formally, we make the following assumptions :

Hypothesis 2.1 *The problem to solve is well-posed and the function $x = G(d)$ can be decomposed into a finite number of elementary operations, which turn out to express the algorithm of resolution.*

However, since this algorithm runs on a computer with finite precision arithmetic, the rounding errors lead to an approximate solution, denoted by $\tilde{x}$. To analyze this approximation error, we have to make some assumptions on the computer arithmetic. We just give here the main principles and refer to [8, 12] for more details.

Definition 2.4 *A floating-point system is defined by a set of floating-point numbers $\mathcal{F}$, a rounding denoted by fl which is a mapping from an interval of the set of reals $\mathcal{R}$ to $\mathcal{F}$, and floating-point arithmetic operations including elementary mathematical functions. The precision of the rounding is called ϵ. More precisely,*

$$\forall x \in \mathcal{R} \text{ such that } fl(x) \text{ exists} \quad |fl(x) - x| \leq |x| \, \epsilon$$

In the following, we use the notation fl to express the result of any sequence of floating-point operations. To derive error analysis for an algorithm, we must assume some bounds on the error of elementary floating-point operations. The standard IEEE [1] requires a correct computer arithmetic in the following sense :

Definition 2.5 *The floating-point arithmetic is said to be correct if, for each standard arithmetic operation between two floating-point numbers, the floating-point result is the rounding of the exact result.*

Unfortunately, this standard is not respected by all computer manufacturers. Furthermore, it does not require anything on elementary functions. Therefore, we will make slightly different assumptions.

Definition 2.6 *Let $\top$ be any arithmetic operation or elementary function and g any monadic operation or function. The floating-point system is consistent if there exists constant K such that the floating-point results satisfy*

$$\begin{aligned} \forall x, y \in \mathcal{F} \quad |fl(x \top y) - x \top y| &\leq K \, \epsilon \\ \forall x \in \mathcal{F} \quad |fl(g(x)) - g(x)| &\leq K \, \epsilon \end{aligned}$$

In the following, we assume that the floating-point system is consistent.

2.4 Backward analysis

The idea of backward analysis [20] is to prove that the approximate solution $\tilde{x}$ is the exact solution of an approximate problem, ie $\tilde{x} = F(\tilde{d})$, with $\|\tilde{d} - d\|$ as small as possible. The algorithm is numerically stable if this backward analysis is feasible, assuming that the floating-point system is consistent. More formally [5] :

Definition 2.7 *An algorithm is numerically stable in V if for any $d \in V$ such that $x = G(d)$, the set $\tilde{D}(d) = \{\tilde{d}/\tilde{x} = G(\tilde{d})\}$ is not empty and if it exists a constant K such that*

$$\forall d \in V \quad \inf_{\tilde{d} \in \tilde{D}(d)} \|\tilde{d} - d\| \quad \leq \quad K\ \epsilon$$

The backward error of a stable algorithm in V is given by

$$B\ \epsilon \quad = \quad \sup_{d \in V} \inf_{\tilde{d} \in \tilde{D}(d)} \|\tilde{d} - d\|$$

This concept of backward error combined with the concept of condition number allows to derive a bound on the perturbation of the approximate solution due to perturbation in the initial data.

2.5 Perturbations of approximate solutions

Let $\tilde{x_0}$ and $\tilde{x}$ be the approximate solutions given by an algorithm for the data d_0 and d. We want the derive bounds on $\|\tilde{x} - \tilde{x_0}\|$ using the rounding precision ϵ and the perturbation $\|d_0 - d\|$. All the initial data are in the set V. We assume that the problem P is q-stable with a regularity q and a condition number C and that the algorithm is stable with a backward error B in V such that the sets $\tilde{D}(d) \bigcap V$ are not empty. We have the following inequality :

Proposition 2.3

$$\|\tilde{x} - \tilde{x_0}\|/\|\tilde{x_0}\| \quad \leq \quad C\ \frac{(B\ \epsilon)^q\ +\ (B\ \epsilon\ +\ \|d - d_0\|)^q}{\|d_0\|^q}\ \frac{\|x_0\|}{\|\tilde{x_0}\|} \qquad (1)$$

Proof. We have

$$\|\tilde{x} - \tilde{x_0}\|/\|x_0\| \quad \leq \quad \|\tilde{x} - x_0\|/\|x_0\| \ +\ \|\tilde{x_0} - x_0\|/\|x_0\|$$

Under the above assumptions, we can apply the condition number definition to get

$$\begin{aligned} \|\tilde{x} - x_0\|/\|x_0\| \quad &\leq \quad C\ (\|\tilde{d} - d_0\|/\|d_0\|)^q \\ \|\tilde{x_0} - x_0\|/\|x_0\| \quad &\leq \quad C\ (\|\tilde{d_0} - d_0\|/\|d_0\|)^q \end{aligned}$$

We then apply the backward error definition to get

$$\begin{aligned} \|\tilde{d} - d_0\| &\leq \|\tilde{d} - d\| + \|d - d_0\| \\ &\leq B\,\epsilon + \|d - d_0\| \\ \|\tilde{d_0} - d_0\| &\leq B\,\epsilon \end{aligned}$$

Combining the inequalities, we finally get the result claimed.

□

This inequality provides an error bound which highlights contributions from numerical instability corresponding to large values of $B\ \epsilon$ and from problem instability corresponding to large values of C. In particular, even with a stable algorithm, the error bound is large if the problem is ill-conditioned, showing that potential inaccuracies may occur in the computed solution.

3 Error analysis and methodology

Our objective is to get an estimation of the regularity and the condition number of a problem and to get an estimation of the numerical stability of the algorithm used to solve it. In linear algebra, numerous condition numbers estimators have been designed [14], even for sparse matrices [2]. However, for general problems, it may be hard to design an algorithm to compute such estimations.

Our methodology uses a statistical approach, by analyzing samples of results. The idea is to perturb the data with random perturbations and to estimate the induced errors in the solutions. But we can only measure the perturbations in the approximate solutions. Therefore, we start from the inequality in (1) which we analyze below.

3.1 Error analysis

The idea is to vary the perturbation in the initial data in order to study the variation in the approximate solution. For very small data perturbations, the rounding errors are the most important and the data perturbations are negligible, whereas for sufficiently large perturbations the rounding errors become negligible. Of course, the perturbations must be small enough to remain in the domain of validity V of the model.

Proposition 3.1 *If the problem is well-conditioned with a regularity q and a relative condition number C and if the algorithm is numerically stable with a backward error B and if the precision of the computation is sufficient, then it exist two thresholds Δ_1 and Δ_2 such that*

$$\forall \alpha \text{ with } \Delta_1 \leq \alpha \leq \Delta_2 \quad C_\alpha/\alpha^q \simeq C \tag{2}$$

where C_α is defined by

$$C_\alpha = \sup_{d \,:\, \|d-d_0\|/\|d_0\|=\alpha} \|\tilde{x} - \tilde{x_0}\|/\|\tilde{x_0}\|$$

Proof. The idea is to neglect the terms relative to ϵ in the inequality 1. Under the above assumptions, it exists a threshold Δ_1 such that $B\ \epsilon \ll \|d_0\|\ \Delta_1$, and a threshold Δ_2 such that the ball of center d_0 and radius $\|d_0\|\ \Delta_2$ is included in V. We apply (1) where we neglect the terms in $B\ \epsilon$. We can also approximate $\|\tilde{x_0}\|$ by $\|x_0\|$. It amounts to approximate $\|\tilde{x} - \tilde{x_0}\|/\|\tilde{x_0}\|$ by $\|x - x_0\|/\|x_0\|$.

We then apply the definition of the condition number, approximating $\sup_{d \in V \text{ and } \|d-d_0\|/\|d_0\|=\alpha} \frac{\|x-x_0\|/\|x_0\|}{(\|d-d_0\|/\|d_0\|)^q}$ by C. □

Our method will then consist in finding the thresholds Δ_1 and Δ_2 and in estimating by a statistical analysis the regularity and the condition number. The thresholds Δ_1 and Δ_2 depend both on the numerical stability and the problem stability. If the precision is not sufficient then it may happen that Δ_1 is larger than Δ_2, so that it is impossible to get an estimation of the regularity and the condition number. This may come from an unstable algorithm or from an ill-conditioned problem. In the first case, our experiments lead to an error almost independent of the data perturbations. In some cases, the error is null because the computed solution is not sensitive at all to the data perturbations. In the second case, we observe large errors and oscillations in the errors.

The threshold Δ_2 has been observed in [5] for the resolution of well-conditioned linear systems. In that paper, data perturbations are also used to derive statistical estimations, but in a slightly different way. First of all, the regularity is assumed to be 1. Secondly, the condition number is estimated by using the inverse problem, in order to retrieve a backward error. The condition number is find to be constant for $\|d - d_0\| < \Delta_2$, as

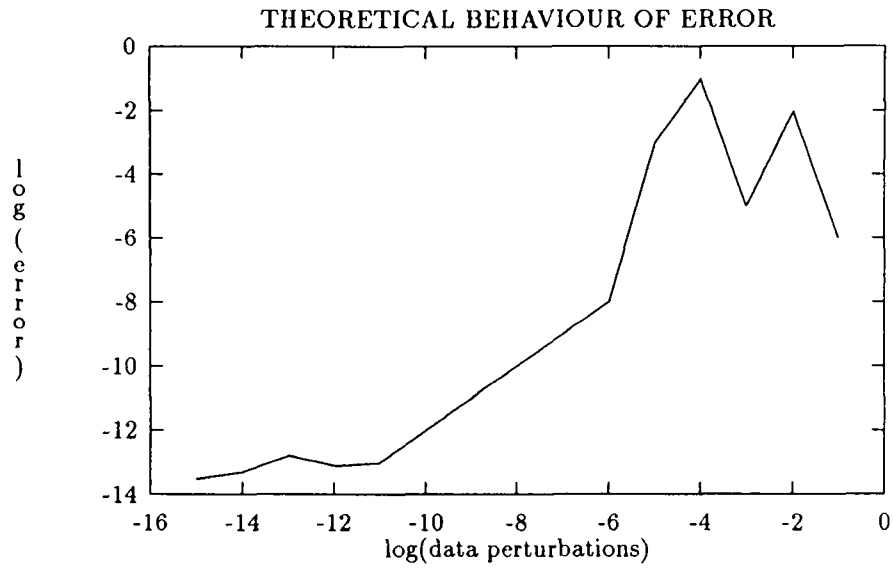

Figure 1: Three domains of data perturbations

predicted by the analysis above. The threshold Δ_1 is not observed in [5] because only stable algorithms are used.

The three cases are illustrated on figure 1, in a log-log scale, with a rounding precision $\epsilon = 10^{-16}$. The thresholds Δ_1 and Δ_2 correspond to data perturbations of size 10^{-11} and 10^{-6}. Between these two values, the logarithm of the error bound varies linearly with the logarithm of the data perturbation. Below Δ_1, the precision is not sufficient to get an accuracy better than roughly 10^{-13}, and beyond Δ_2 oscillations show the limit of problem stability. Between these two values, the equation of the straight line allows to estimate the regularity and the condition number of the problem, as explained in next section.

3.2 Statistical analysis

We use the statistical approach defined by [10] in the so-called SCALP methodology. It follows a statistical approximation generally used in econometry. We recall briefly here the principle and refer to [10] for more details.

If a function $z = f(y)$ is equivalent near 0 to $C\ y^q$ and if we get a statistical sample $z_i = f(y_i)$, then we can use a log-linear regression of the

z_i over the y_i to estimate the numbers C and q. Moreover, the regression yields a coefficient of significance which must be almost equal to 1 (say at least 0.8) to validate the regression.

In SCALP, this method is applied for three different kinds of perturbations, which are data perturbations as in our approach, stochastic perturbations on the arithmetic to simulate rounding errors and troncature perturbations to simulate different precisions (and then to study the influence of ϵ). However, the last two perturbations are quite costly because the computer arithmetic is then replaced by an arithmetic emulated by software. In [10], it is claimed that the data perturbations may give false conclusions if the algorithm is very unstable. This is true if the regression is used directly without looking for the domain of validity bounded by Δ_1 and Δ_2. In our approach, we first begin by analyzing the curve giving the error function of the perturbation (in a log-log scale) in order to find this domain of validity. Only then we apply the log-linear regression by using SCALP. The method fails if the domain of validity does not exists. However, in our experiments, we could conclude if the failure was due to numerical instabilities or problem ill-conditioning.

In the following, we assume that inequality (2) is satisfied. We want to apply the log-linear regression between C_α and α, since we have $C_\alpha \simeq C\ \alpha^q$. We proceed in two steps : for α fixed, we compute a statistical estimation of C_α, then we apply the regression as in SCALP by varying α. We use relative componentwise data perturbation as in [5], defined by

Definition 3.1 *Let n be the dimension of the input space. The data perturbation of size α is defined by*

$$d_i \ = \ d_{0,i}\ (1 + e_i\ \alpha) \quad i = 1, \cdots, n$$

where e_i is random variable which can take the values $+1$ and -1.

In fact, this perturbation does not define a usual norm, but it it straightforward to extend the results above to this metric.

We use the following statistical estimation :

Definition 3.2 *Let N be the number of samples generated for the data perturbation of size α. The statistical error bound is given by*

$$SC_\alpha \ \simeq \ \max_{j=1,\ldots,N} \|\tilde{x_j} - \tilde{x_0}\| / \|\tilde{x_0}\|$$

In practice, we use size of perturbations defined by $\alpha = 2^{-m}$, allowing a large range of perturbations. For each size of perturbation, the software generates N computations where the data are randomly perturbed, yielding a file of samples. It should be noted that this file must be recorded in binary format in order to avoid rounding effects. Then we first plot all these points (all the error estimations versus the size of perturbations) in log-log scale in order to detect the domain of validity. Though this step needs currently manual intervention, it could be automated.

If the method succeeds, that is to say if we find a domain $\Delta_1 \leq \alpha \leq \Delta_2$ where $\log(C_\alpha)$ depends almost linearly on $\log(\alpha)$, we apply a log-linear regression in this domain to estimate the regularity and the corresponding condition number of the problem. Furthermore, for $\alpha = \Delta_1$, the error estimation is valid and can be used to estimate the approximation error $\|\tilde{x_0} - x_0\|/\|x_0\|$, so that we can derive an estimation of the number of significant digits given by the algorithm for the data d_0. It should be noted that this error estimation is validated only because we could find the threshold Δ_1 isolating numerical instabilities.

If the method fails, we conclude to instabilities due to rounding effects or ill-conditioning. In our experiments, we could detect numerical instabilites because the error estimations were almost constant, whereas they were large and oscillating for problem instabilities.

3.3 Summary of the methodology

Our methodology can be summarized as follows :

1. generate samples of error estimations for varying size of perturbations ; store them in binary format.
2. Plot all the samples and detect the domain of validity of the regression. If the domain is too small, conclude to instabilities.
3. Compute in the domain of validity a statistical estimation of the error bound for each size of perturbation.
4. Apply a log-linear regression to these estimations to compute the regularity and the condition number of the problem.
5. Estimate the accuracy of the numerical result.

Practical values of the number of sizes and the number of samples are discussed in section 5.

4 Numerical Experiments

4.1 Example of numerical instability

This example emphasizes the requirement for large perturbations, as quoted in [10]. The problem is simply the identity, but the algorithm is intentionaly designed to be unstable. The first step is a contraction so that its intermediate result is not sensitive to perturbations in the initial data. The second step is the inverse operation which is very sensitive to roundoff errors.

Example 4.1 *From an initial value x_0, compute successively :*

$$\begin{cases} x_{n+1} = (x_n + b)/2 & for \quad n = 0, \cdots, p-1, \\ y_0 = x_p & \\ y_{n+1} = 2y_n - b & for \quad n = 0, \cdots, p-1 \end{cases}$$

The final result y_p is evidently equal to x_0. For small data perturbations, the error in the intermediate result x_p is negligible so that the estimation of the error in the final result y_p is also very small and cannot be used to estimate the error in the solution.

This algorithm is implemented in IEEE double precision (53 digits in the mantissa) for $b = 0.5$ and $x_0 = 3.14$ with $p = 30$. It yields an approximate y_{30} accurate up to 9 digits. The exact relative error is $5\ 10^{-9}$.

We perturb randomly the initial value x_0 and compute a statistical estimation of the relative error on y_p. We use data perturbations ranging from 10^{-15} to 1. The results are plotted in figure 2 in a log-log scale. Infinite logarithms are represented by the value -20 by convention. For perturbations smaller than 3.10^{-8}, the error estimation is 0. and jumps to about 4.10^{-8} for a perturbation equal to 3.10^{-8}. Because rounding errors affect the digits left to the k^{eme} say, the results are not sensitive to small perturbations which affect the digits right to this k^{eme}. Therefore, the results are identical for small perturbations. We then obtain a curve with a horizontal line (at the value 10^{-20}) which demonstrates a numerical instability. For perturbations greater than 3.10^{-8}, the curve is a straight line of slope 1, which allows to conclude that the problem is stable. By using the regression of SCALP in the domain $3.10^{-8} < |\Delta d| < 1$, we can estimate the regularity and the condition number of the problem. We find as expected a regularity equal to 1. and a condition number equal to 0.5 (instead of 1), with a coefficient of significance equal to 0.99. Furthermore, we estimate the numerical relative error to 4.10^{-8} in good agreement to the exact relative error.

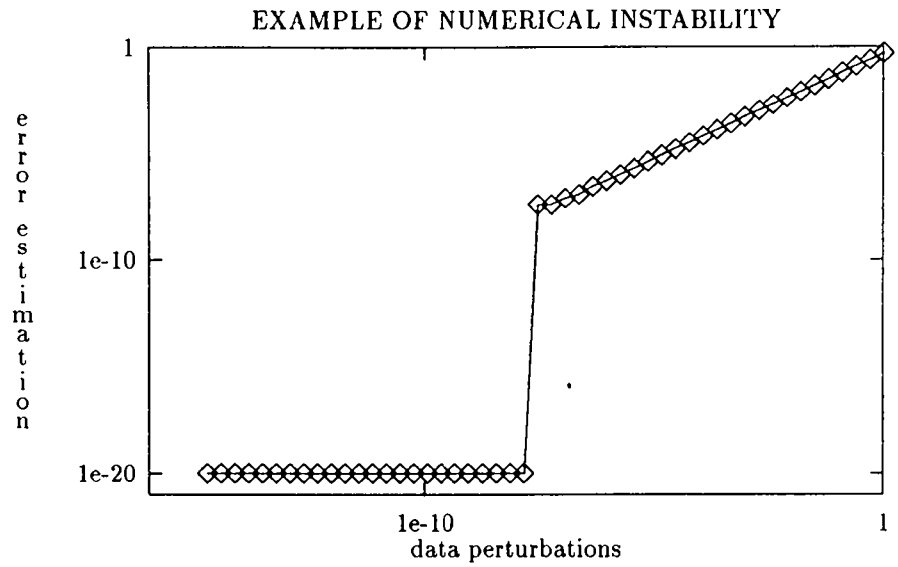

Figure 2: Example of Requirement for Large Perturbations

4.2 Resolution of a linear system

The problem we deal with is to solve a dense non singular linear system.

Example 4.2 *Given a matrix A of order n and a right-hand side b, find x such that $A\ x\ =\ b$.*

This problem has been thoroughly studied and main results can be found for example in [17, 13]. The regularity is shown to be 1., and various condition numbers have been proposed. The most current used is $\|A^{-1}\|\ \|A\|$. However, it may in practice overestimate the actual error due to artificial ill-conditioning [17]. This phenomenon comes from the sensitivity of the condition number to row or column scaling. The Bauer-Skeel condition number, defined by $\||A^{-1}|\ |A|\|$, is independent of row scaling and therefore avoids this artificial ill-conditioning. But it is quite sensitive to column scaling. This condition number is used in practice for sparse matrices [2].

We use two algorithms to solve this linear system : Gaussian elimination and QR factorization, which are described for example in [13]. For both of them, a backward analysis can be performed. Gaussian elimination with

Matrix	Conditioning	Numerical Difficulty
TRIDIAG(10)	Well-Cond.	None
MATPIV(7)	Well-Cond.	Pivoting in Gauss
DESCALED(10)	Artificial Ill-Cond.	Balancing in QR
HILBERT (5 or 10)	Ill-Cond.	None

Table 1: Characteristics of the matrices tested

no pivot is shown to be unstable for some matrices while partial pivoting yields a stable algorithm. In the same way, QR factorization with no balancing is unstable for descaled matrices whereas balancing avoids numerical difficulties.

To exhibit both numerical and problem instabilities, we use different matrices : a tridiagonal diagonal dominant well-conditioned matrix (TRIDIAG(n)), a well-conditioned matrix requiring pivoting in Gauss (MATPIV(n)), a descaled matrix requiring balancing in QR and exhibiting artificial ill-conditioning (DESCALED(n)), Hilbert matrices of varying orders with condition numbers increasing with the order (HILBERT(n)). The tridiagonal matrix and Hilbert matrices are classical ones, the matrix MATPIV has been designed by the author, and the descaled matrix is issued from [5]. The features of these matrices are summarized in table 1.

For all these matrices, we have experimented the four algorithms in single and double precision. We report here only the main significant results, but all results are consistent with our methodology. We perturb both the right-hand side and the matrix, and compute a statistical error bound on the solution. We use infinite norms for the error bound. In all our experiments, we take $N = 10$ samples to compute the statistical error estimation. Data perturbations range from 10^{-7} to 1 in single precision, and from 10^{-15} to 1 in double precision. Of course, this range can be restricted for practical use, as discussed in section 5. In order to compute the exact error, we start from a known solution x and derive the right-hand side b by simply computing $b = A\,x$. We have experimented with various right-hand sides, and report only those with $x = (1, 1, \ldots, 1)$.

The tables 2,3, 4,5 contain for each matrix the spectral condition number (computed using MATLAB), the exact error (comparing the result with the exact solution which is given), and the estimations of the regularity, the condition number and the error. In all cases, the coefficient of significance of

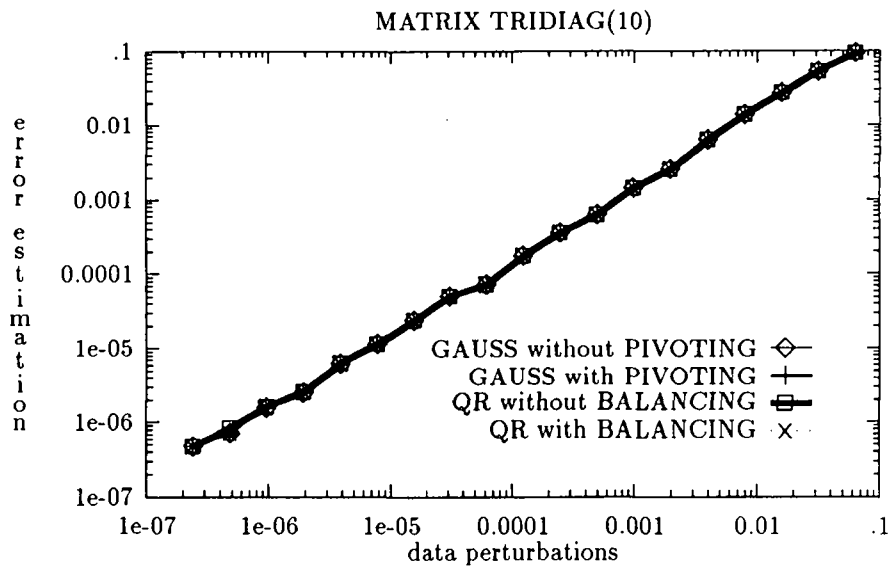

Figure 3: Stable Algorithms and Well-Conditioned Matrix

the regression was roughly equal to 1. (in general 0.99). For ill-conditioned matrices or unstable algorithms, we determinate first the domain of validity to estimate the regularity and the condition number. The figures 3,4,5, 6 plot the error estimations for varying data perturbations, in a log-log scale. For all cases, the domain of validity corresponds to a straight line of slope 1.

For the matrix $TRIDIAG(10)$ (see figure 3), the four algorithms are stable and the problem is very well-conditioned. We get four almost identical straight lines. The condition number and the error estimations are in very good agreement with the exact condition number and error.

The matrix $MATPIV(7)$ requires pivoting in Gaussian elimination. The figure 4 shows the numerical instability when no pivot is done. We get as expected an horizontal line, corresponding to the domain $\Delta d < \Delta_1$, followed by a line of slope about one corresponding to the domain $\Delta_1 < \Delta d < \Delta_2$, with the values $\Delta_1 = 2\ 10^{-3}$ and $\Delta_2 = 0.1$. On the other hand, pivoting eliminates numerical instabilities, giving $\Delta_1 = 10^{-7}$ and $\Delta_2 = 6\ 10^{-2}$. For both algorithms, by using SCALP in the domain of validity, we find good estimations of the condition number and of the error.

Algorithm	Spectral Condition Number	Estimated Condition Number	Estimated Regularity	Exact Error	Estimated Error
Gauss Without Pivoting	3	2	1	0.	$4\ 10^{-7}$
Gauss With Pivoting	3	2	1	0.	$4\ 10^{-7}$
QR Without Balancing	3	2	1	$7\ 10^{-8}$	$4\ 10^{-7}$
QR With Balancing	3	2	1	$7\ 10^{-8}$	$4\ 10^{-7}$

Table 2: Results in Single Precision for Matrix TRIDIAG(10)

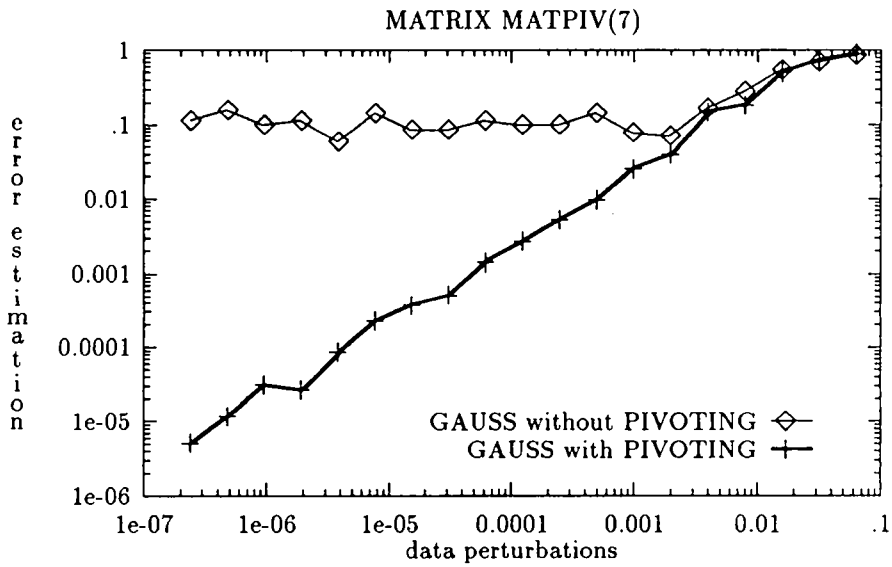

Figure 4: Unstable Gaussian Elimination for Well-Conditioned Matrix

Algorithm	Spectral Condition Number	Estimated Condition Number	Estimated Regularity	Exact Error	Estimated Error
Gauss Without Pivoting	50	20	1	$4\ 10^{-2}$	10^{-1}
Gauss With Pivoting	50	20	1	10^{-6}	$5\ 10^{-6}$

Table 3: Results in Single Precision for Matrix MATPIV(7)

Algorithm	Spectral Condition Number	Estimated Condition Number	Estimated Regularity	Exact Error	Estimated Error
QR Without Balancing	10^{12}	1	1	$6\ 10^{-4}$	$5\ 10^{-4}$
QR With Balancing	10^{12}	1	1	$6\ 10^{-15}$	$3\ 10^{-15}$

Table 4: Results in Double Precision for Matrix DESCALED(10)

The same phenomenon is observed for QR factorization with the matrix $DESCALED(10)$ (see figure 5). With no balancing, the algorithm is unstable, as shown by the horizontal line, and becomes stable after balancing the matrix. Here, the matrix is intentionnaly row-descaled, so that the balancing technique consists simply in rescaling the rows. Furthermore, this matrix exhibits the phenomenon of artificial ill-conditioning. The spectral condition number is very large, equal to 10^{12}, while the estimated condition number, equal to 3, is much smaller, and is in accordance with the error. The estimated error is of the same order as the exact error.

Ill-conditioned matrices, such as Hibert matrices, give rise to problem instabilities which are visualized by high oscillations in figure 6. The condition number increases with the order of the matrix, as shown by the results in single precision with matrices of order 5 and 10. For the order 5, it remains a domain of validity which disappears for the order 10. Double precision allows to push further this barrier on the order, as illustrated by the results

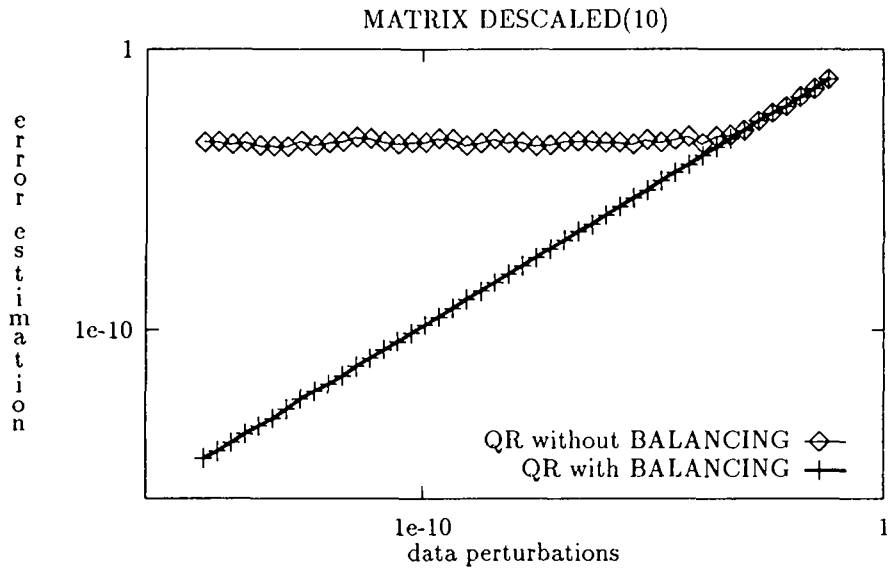

Figure 5: Unstable QR Factorization and artificial ill-conditioning

on the order 10, comparing single and double precision. But even in double precision, it is not possible to estimate a condition number for higher orders, because there is no domain of validity.

4.3 Example of a non linear equation

As we mentioned earlier, some problems can have a regularity different from one. To illustrate this, we take a simple example, which consists in solving a second order equation. It should be noted that our methodology could also be applied to more sophisticated problems such as eigenproblems.

Example 4.3 *Given the reals a and b, find the set of complex solution of the equation* $x^2 - a\,x + b = 0$.

For sake of simplicity and without loss of generality, the coefficient of x^2 is normalized. This problem has a regularity of 1 if it has two distinct solutions, or more precisely if $|a^2 - 4b| > \alpha$, where α is some positive constant. On the other hand, it has a regularity of 0.5 if it has almost equal solutions, that is to say if $|a^2 - 4b| \leq \alpha$. The first objective of the experiment is to find these

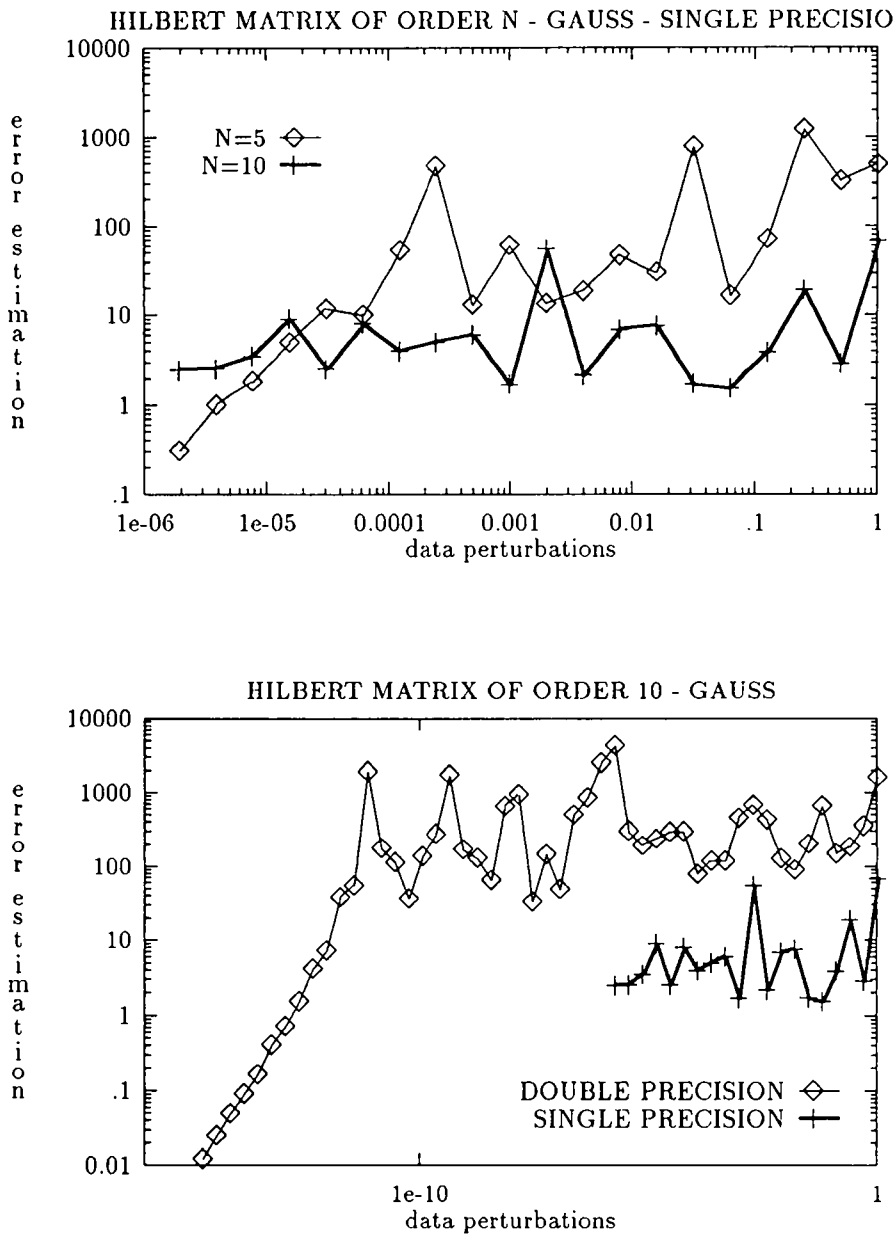

Figure 6: Ill-conditioned Matrix : influence of the order and of the precision

Order	Precision	Spectral Condition Number	Estimated Condition Number	Estimated Regularity	Exact Error	Estimated Error
5	Single	$4\ 10^5$	$2\ 10^5$	1.1	$9\ 10^{-3}$	$4\ 10^{-2}$
5	Double	$4\ 10^5$	$3\ 10^5$	1	$3\ 10^{-12}$	$3\ 10^{-10}$
10	Double	10^{13}	10^{13}	1	$3\ 10^{-4}$	10^{-2}

Table 5: Results with Gauss Algorithm for Hilbert Matrices

results with our methodology. We recall below the results on the condition numbers.

Proposition 4.1 *If x_1 and x_2 are two distinct solutions of the equation $x^2 - a\,x + b = 0$, then the problem has a regularity 1 with the condition numbers C_1 and C_2 corresponding to each solution given by*

$$C_i = \frac{|a|\,|x_i| + |b|}{|x_i|\,|2x_i - a|} \quad i = 1, 2$$

If x is a double solution of the same equation, then the problem has a regularity 0.5 with a condition number given by

$$C = \sqrt{3}$$

Proof. If the equation has two solutions, then $2x_i - a \neq 0$ and we can write at the first order,

$$(2x - a)\Delta x - x\Delta a + \Delta b = 0$$

Applying it to perturbations of the form $\Delta a = \alpha\, a$ and $\Delta b = \alpha\, b$ we obtain with $|\alpha| \leq \eta$ and $|\beta| \leq \eta$:

$$|\Delta x_i|/|x_i| \leq C_i\, \eta$$

which is reached for some α and β.

If the equation has one solution, then we must develop the perturbations

$$\begin{aligned}(\Delta x)^2 - \Delta a\,\Delta x - x\,\Delta a + \Delta b &= 0\\ (\Delta x - \Delta a/2)^2 &= x\,\Delta a - \Delta b + (\Delta a)^2/4\end{aligned}$$

Applying it to perturbations of the form $\Delta a = \alpha\, a$ and $\Delta b = \alpha\, b$ we obtain with $|\alpha| \leq \eta$ and $|\beta| \leq \eta$:

$$|\Delta x_i|/|x| \leq C\,\eta^{0.5}$$

which is reached for some α and β.

□

The simplest algorithm to solve this second order equation is to compute the discriminant Δ and the solutions in the complex field by

$$\Delta = a^2 - 4b \tag{3}$$
$$x_1 = (a - \sqrt{\Delta})/2 \tag{4}$$
$$x_2 = (a + \sqrt{\Delta})/2 \tag{5}$$

Here, $\sqrt{\Delta}$ means $\sqrt{\Delta}$ if $\Delta \geq 0$ and $i\sqrt{|\Delta|}$ if $\Delta < 0$.

However, this algorithm is unstable to compute x_1 if this root is very small or in other words if $|b|$ is small compared to $|a|$, as mentioned in [18]. In that case, a stable algorithm to compute x_1 is given by

$$x_1 = 2b/(a + \sqrt{\Delta}) \tag{6}$$

The second objective of the experiment is to detect this numerical instability by our methodology.

As far as we know, the backward analysis of these two algorithms is not well documented. Therefore, we perform this analysis below.

Proposition 4.2 *Let $\tilde{x}_1$ and $\tilde{x}_2$ be the floating-point results obtained by the first algorithm (3),(4), (5). If the floating-point system is consistent and if ϵ denotes the rounding precision, then $\tilde{x}_1$ and $\tilde{x}_2$ are the solutions of the second-order equation $x^2 - \tilde{a}\,x + \tilde{b} = 0$, where the reals $\tilde{a}$ and $\tilde{b}$ satisfy*

$$\begin{aligned}|\tilde{a} - a|/|a| &\leq (1 + \sqrt{|\Delta|}/|a|)\,\epsilon\\ |\tilde{b} - b|/|b| &\leq (1 + |a|^2/|b| + |\Delta|/|b|)\,\epsilon\end{aligned}$$

Proof. The floating-point results are the solutions of the equation $x^2 - \tilde{a}\, x + \tilde{b} = 0$, where $\tilde{a} = \tilde{x_1} + \tilde{x_2}$ and $\tilde{b} = \tilde{x_1} * \tilde{x_2}$.

All relative rounding errors are denoted by α_i, such that $\alpha_i \leq \epsilon$, where ϵ is the precision of the consistent floating-point system.

The majorations below are at the first order in ϵ :

$$
\begin{aligned}
fl(\Delta) &= (a^2\,(1+\alpha_1) - 4b)(1+\alpha_2) \\
fl(\Delta) &= \Delta + \gamma_1 \\
|\gamma_1| &\leq (2|a^2| + 4|b|)\,\epsilon \\
fl(\sqrt{\Delta})^2 &= \Delta + \gamma_1 + 2\Delta\alpha_3 \\
\tilde{x_1} &= (a - fl(\sqrt{\Delta})\,(1+\alpha_4)/2 \\
\tilde{x_2} &= (a + fl(\sqrt{\Delta})\,(1+\alpha_5)/2 \\
\tilde{a} &= a + 1/2\,[(a - fl(\sqrt{\Delta})\,\alpha_4) + (a + fl(\sqrt{\Delta})\,\alpha_5] \\
|\tilde{a} - a| &\leq (|a| + \sqrt{|\Delta|})\,\epsilon \\
\tilde{b} &= (a^2 - fl(\sqrt{\Delta})^2)\,(1+\alpha_4)\,(1+\alpha_5)/4 \\
\tilde{b} &= b + 1/4\,[a^2\,(\alpha_4+\alpha_5) - \gamma_1 - 2\Delta\alpha_3 - \Delta(\alpha_4+\alpha_5) \\
|\tilde{b} - b| &\leq (a^2 + |b| + |\Delta|)\,\epsilon
\end{aligned}
$$

□

Proposition 4.3 *Let $\hat{x}_1$ and $\tilde{x}_2$ be the floating-point results obtained by the second algorithm (3), (6),(5). If the floating-point system is consistent and if ϵ denotes the rounding precision, then $\hat{x}_1$ and $\tilde{x}_2$ are the solutions of the second-order equation $x^2 - \hat{a}\, x + \hat{b} = 0$, where the reals $\hat{a}$ and $\hat{b}$ satisfy*

$$
\begin{aligned}
|\hat{a} - a|/|a| &\leq (0.5 + \frac{|a|}{|a + \sqrt{|\Delta|}|} + 0.5\frac{\sqrt{|\Delta|}}{|a|} + 0.5\frac{|\Delta| + |b|}{|a|\;|a + \sqrt{|\Delta|}|})\,\epsilon \\
|\hat{b} - b|/|b| &\leq 3\,\epsilon
\end{aligned}
$$

Proof. The proof is similar to the previous one. □

This backward analysis shows that the product of the roots $\tilde{b}$ may be inaccurate if $|b|$ is small compared to a^2, hence if x_1 is small. This numerical difficulty arises because of two rounding effects when computing $\tilde{x}_1$. First, rounding errors occurs in the computation of Δ because of the absorption of b by a^2. Then these errors are magnified by a cancellation between a

Equation	a	b	Regularity	Numerical Difficulty
1	0.3	0.02	1	None
2	1.00001 10^5	1	1	Small root x_1
3	0.2	0.01	0.5	None

Table 6: Description of Equations Tested

and $\sqrt{\Delta}$. If $|b|$ is very small compared to a^2 then Δ is rounded to a^2 yielding the computed solutions 0 and a instead of about b/a and a. These rounding effects, absorption and cancellation, are described for example in [8]. Cancellation disappear in the second algorithm, as demonstrated by its backward analysis. In any case, the product $\hat{b}$ is near b. If $|b|$ is very small compared to a^2, the computed solutions b/a and a are still accurate.

We have experimented our methodology with three equations, defined in table 6. The first one is well-conditioned, with two distinct roots, the second one is well-conditioned with a very small root and the third one is ill-conditioned with a double root.

We study separately the two roots x_1 and x_2 in order to detect the numerical instability in the root x_1. The results of our methodology are plotted in figures 7,8, 9 and summarized in table 7. The first case demonstrates numerical stability and well-conditioned problem. In the second case, the numerical instability in the root x_1 is illustrated by an almost horizontal line with some oscillations. In general, the results are identical (giving an error estimation plotted at 10^{-16}) except for some data perturbations which combines with the cancellation to yield error estimations at about 10^{-8}. In the third case, we clearly obtain a regularity of 0.5 with a small condition number. In all cases, our results are in good agreement with the theory.

5 Practical Use

The methodology described so far requires two technical steps : first to prepare the source code, then to analyze the resulting file. The source code must contain calls to a library routine to perturb the chosen initial data and to write instruction to save the data to be analyzed ; they must be embedded into two nested loops to execute the code for different sizes of perturbations

Equation	Exact Condition Number	Estimated Condition Number	Estimated Regularity	Exact Error on x_1	Estimated Error on x_1
1	5	5	1	0	$4\ 10^{-15}$
2	2	2	1	$3\ 10^{-7}$	$7\ 10^{-7}$
3	1.7	1.7	0.5	10−8	$5\ 10^{-8}$
Equation	Exact Condition Number	Estimated Condition Number	Estimated Regularity	Exact Error on x_2	Estimated Error on x_2
1	4	4	1	0	$4\ 10^{-15}$
2	1	1	1	0	$8\ 10^{-16}$
3	1.7	1.7	0.5	10−8	$5\ 10^{-8}$

Table 7: Results for second-order equations in double precision

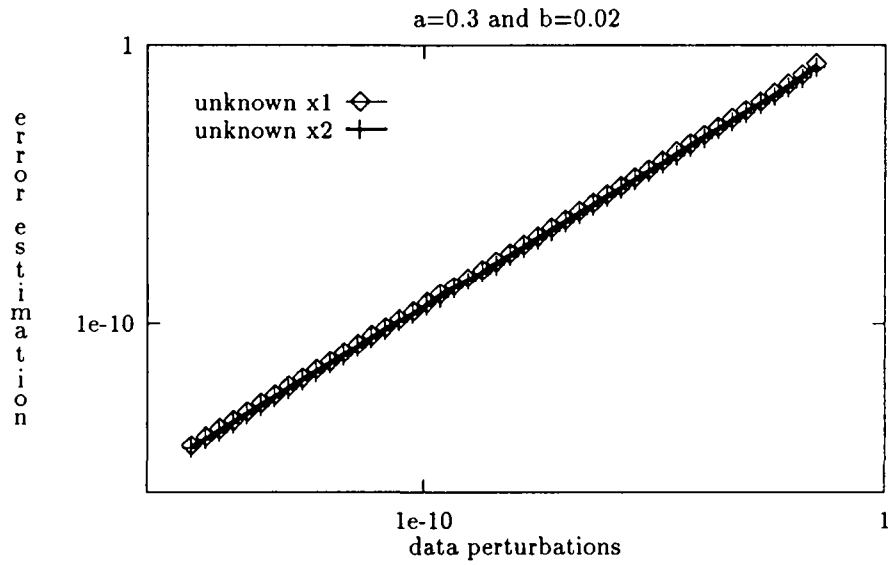

Figure 7: Well-Conditioned Equation and Stable Algorithm

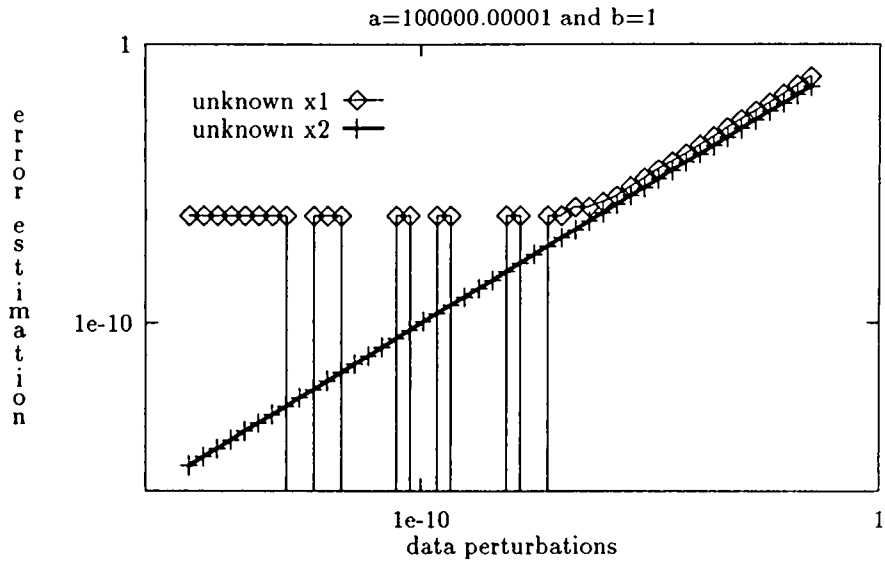

Figure 8: Well-Conditioned Equation and Unstable Algorithm

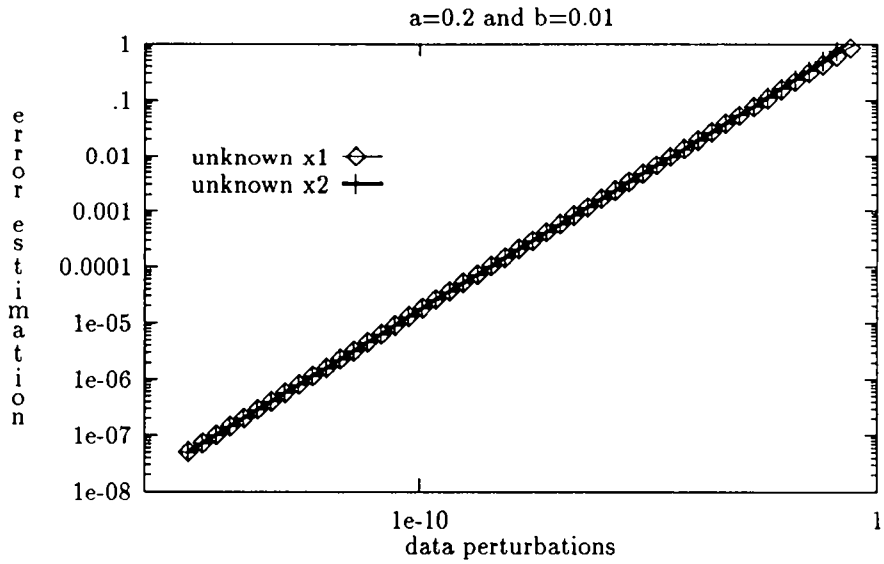

Figure 9: Ill-Conditioned Equation and Stable Algorithm

and, for each perturbation, for several samples.

The analyzis of the file consists in first detecting the domain of validity by plotting for example the error estimations and then applying the log-linear regression of SCALP to estimate the regularity and condition numbers.

This tool will be integrated in the next future into a toolbox, called Aquarels, which is currently under development [9]. This toolbox integrates into a consistent and user-friendly software structure tools contributing to improve or control the numerical quality of scientific software. Aquarels integrates in particular interval analysis, multiple precision and perturbation techniques. In order to use those tools easily, extended Fortran has been defined and a preprocessor has been designed to translate it into standard Fortran by including library calls. Therefore, the preparation step of our methodology will be greatly simplified by using Aquarels. In the same way, a tool is planned to analyze automatically the resulting file. Various perturbations methods can actually be easily included in this toolbox.

The cost at execution is roughly the number of executions, in addition with the output operations. The code must be executed $nb_{size} \times nb_{samples}$ times, where nb_{size} is the number of sizes of the perturbations and $nb_{samples}$ is the number of samples for each perturbation size. All our experiments make use of large numbers (50 sizes in double precision and 10 samples). In practice, if we want an estimation of the regularity and condition number, the regression must be applied with about 10 points and 5 samples seem sufficient, which amounts to a relative overhead of about 50. But the method can also be applied to visualize only the instabilities. In that case, 6 sizes with 1 sample for each may be enough. It should be noted that the method is applied only to a small part of a scientific code, thus reducing the global cost. By the way, all the executions are of course independent and can be run easily in parallel. Other perturbations techniques such as CESTAC require only three executions, but they are actually quite expensive because they must perturb each result of a floating-point operation. Moreover, they only give an error estimation, with no regularity and condition number estimations.

6 Conclusion

Problem conditioning and numerical stability combine together to deliver results with some inaccuracy. We have designed a tool which allows :

- to make the distinction between unstable algorithms and ill-conditioned problems ;
- to estimate the regularity and the condition number of the problem ;
- to estimate the error in the solution.

This tool is based on random perturbations in the initial data and on a statistical regression of the error estimation versus the size of the perturbation. Experiments conducted so far prove the effectiveness of the method. Moreover, this method is quite cheap and easy to use. Its use will be even more simplified thanks to the system Aquarels.

Our method currently applies to direct algorithms. Iterative algorithms add a new challenge because of the approximation error. The statistical analysis must take into account the number of iterations or the convergence error.

In practice the method is applied only to a specific part of a scientific code. It may be applied successively to more and more restricted parts, in order to isolate the difficulties. For such experiments, the choice of the input data to perturb and of the output data to analyze is relevant. We plan to investigate such an approach on real applications instead of well-defined algorithms.

References

[1] American National Standard. *IEEE Standard for Binary Floating-Point Arithmetic.* ANSI/IEEE Std 754-1985, publié par IEEE, New-York.

[2] M. Arioli, J.W. Demmel, I.S. Duff. *Solving sparse linear systems with sparse backward error.* Rapport CSS 214, Harwell Laboratory, England, 1988. To appear in SIAM J. Matrix Anal. and Applics.

[3] B. Bliss, M.C. Brunet and E. Gallopoulos. *Automatic Program Instrumentation with Applications in Performance and Error Analysis*

[4] M.C. Brunet. *Contribution à la fiabilité de logiciels numériques et à l'analyse de leur comportement: une approche statistique.* Thèse de Doctorat de l'Université Paris-IX, Janvier 1989.

[5] F. Chatelin, V. Frayssé. *Arithmetic Reliability of Algorithms* Symposium on High Performance Computers, Montpellier (France), October 1991.

[6] F. Chatelin. *Résolution approchée d'équations sur ordinateurs.* Notes de cours de DEA de statistique, Université Paris VI, 1990.

[7] J.M. Chesneaux. *Modélisation et conditions de validité de la méthode CESTAC.* C.R. Acad. Sci. Paris, t.307, Série I, p. 417-422, 1988.

[8] J. Erhel. *Erreurs de calcul des ordinateurs* IRISA Report no 552, September 1990.

[9] J. Erhel, B. Philippe *AQUARELS : a Problem-Solving Environment for Numerical Quality* 13^{th} IMACS World Congress on Computation and Applied Mathematics, Dublin (Ireland), June 1991.

[10] P. François. *Contribution à l'étude de méthodes de contrôle automatique de l'erreur d'arrondi : la méthodologie SCALP.* Thèse de l'INPG, Décembre 89.

[11] P. François, J.M. Muller. *Faut-il faire confiance aux ordinateurs ?* Research Report LIP, 1990.

[12] D. Goldberg. *What every Computer Scientist Should Know about Floating-point Arithmetic.* ACM Computing Survey, 3, no 1, March 1991.

[13] G. H. Golub, C. F. Van Loan. *Matrix Computations.* The John Hopkins University press, 1983.

[14] N.J. Higham. *A Survey of Condition Number Estimation for Triangular matrices.* SIAM Review, 29, pp. 575-596, 1987.

[15] J.L. Larson, A.H. Sameh. *Algorithms for roundoff error analysis- A relative approach.* Computing, 24, pp. 275-297, 1980.

[16] W. Miller, D. Spooner. *Software for roundoff analysis, II.* ACM TOMS, 4, pp. 369-387, 1978.

[17] G. W. Stewart. *Introduction to Matrix Computations* Academic Press, 1973.

[18] G. W. Stewart, Ji-guang Sun. *Matrix Perturbation Theory* Academic Press, 1990.

[19] J. Vignes. *Review on stochastic approach to round-off error analysis and its applications.* Mathematics and Computers in Simulation, 30, pp.481-492, Special issue on stochastic methods and round-off error analysis, 1988.

[20] J.H. Wilkinson. *Rounding errors in algebraic processes* Englewoods Cliffs, N.J.: Prentice-Hall, 1963.

LISTE DES DERNIERES PUBLICATIONS INTERNES IRISA

PI 592 SCHEDULING IN DISTRIBUTED SYSTEMS : SURVEY AND QUESTIONS
Yasmina BELHAMISSI, Maurice JEGADO
Juin 1991, 36 pages.

PI 593 APPLICATION OF BELLEN'S PARALLEL METHOD TO ODE's WITH DISSIPATIVE RIGHT-HAND SIDE
Philippe CHARTIER
Juin 1991, 24 pages.

PI 594 PROGRAMMATION D'UN NOYAU UNIX EN GAMMA
Pascale LE CERTEN, Hector RUIZ BARRADAS
Juillet 1991, 48 pages.

PI 595 CALCULATING THE BUSY PERIOD DISTRIBUTION OF THE M/M/1 QUEUE
Louis-Marie LE NY, Gerardo RUBINO, Bruno SERICOLA
Juillet 1991, 11 pages.

PI 596 EFFICIENT CODE GENERATION FOR DISTRIBUTED MEMORY MACHINES
Françoise ANDRE, Olivier CHERON, Jean-Louis PAZAT, Henry THOMAS
Juillet 1991, 14 pages.

PI 597 KOAN : A SHARED VIRTUAL MEMORY FOR THE iPSC/2 HYPERCUBE
Zakaria LAHJOMRI, Thierry PRIOL
Juillet 1991, 32 pages.

PI 598 KOAN : A VERSATILE TOOL FOR PARALLELIZING REALISTIC RENDERING ALGORITHMS
Didier BADOUEL, Kadi BOUATOUCH, Zakaria LAHJOMRI, Thierry PRIOL
Juillet 1991, 28 pages.

PI 599 STATISTICAL ESTIMATION OF ROUNDOFF ERRORS AND CONDITION NUMBERS
Jocelyne ERHEL
Septembre 1991, 34 pages.

Imprimé en France
par
.l'Institut National de Recherche en Informatique et en Automatique.

ISSN 0249-6399