Noname manuscript No.
(will be inserted by the editor)

On the computational cost of Krylov subspace methods for solving linear algebraic systems

J. Liesen · Z. Strakoš

the date of receipt and acceptance should be inserted later

Mathematics Subject Classification (2000) 65F10, 65Y20, 68Q25

cost. the amount or equivalent paid
or charged for something: price.
(Merriam-Webster Online Dictionary)

1 Motivation

In this essay we deal with the cost of a computation, or shortly the *computational cost*. Not just any computation, but a specific one: the iterative solution of a linear algebraic system $Ax = b$ by a Krylov subspace method. Our purpose is not the derivation of new theorems, but a discussion of the main issues in the context of computational cost of iterative methods in general, and Krylov subspace methods in particular. We will make a case that for the evaluation of the computational cost of Krylov subspace methods the given method and data should not be separated, and that the aspects of *convergence* and *numerical stability* (effects of rounding errors) are inherently linked.

To support this point we will contrast our concept of computational cost with classical complexity theory. Furthermore, we will discuss important issues in the convergence and numerical stability analysis of Krylov subspace methods (like nonlinearity, attainable accuracy, and delay of convergence) and describe similarities and differences with other methods for solving $Ax = b$ (like stationary iterative methods and direct methods). We believe that the issues raised and explained in the area of Krylov subspace methods help in the general understanding of concepts like complexity, computational cost, convergence, and stability in numerical analysis and scientific computing.

The work of J. Liesen was supported by the Heisenberg Program of the Deutsche Forschungsgemeinschaft.

The work of Z. Strakoš was supported by the research project MSM0021620839 and partially also by the GACR grant 201/09/0917.

Institute of Mathematics, Technical University of Berlin, Straße des 17. Juni 136, 10623 Berlin, Germany, E-mail: liesen@math.tu-berlin.de · Charles University in Prague, Faculty of Mathematics and Physics, Sokolovská 83, 18675 Prague, Czech Republic, E-mail: strakos@karlin.mff.cuni.cz

Before we go into the details of linear algebraic systems and solvers for them, we will discuss the term computational cost of iterative methods. What is the price to be paid for an iterative solution?

2 Computational Cost of Iterative Methods

In numerical analysis the price of a computation is typically identified with the *memory requirements* and the *execution time* of an algorithm that accomplishes the task at hand. Here we focus on the execution time. Among the many factors that determine the timing, the following are particularly important:

(a) The number of computer operations.
(b) Matching the computation with the given computer resources.

Both factors depend on the given computer architecture. For simplicity of the exposition we do not consider issues related to parallel computer architectures. For basic overviews of concepts related to parallel computations we refer to [14,16,29]. In the sequential context, the number of computer operations, i.e., the factor (a), is approximated by the number of *floating point operations (flops)*; see, e.g., [19, Section 1.2.4], or [31, Section 1.1].

The factor (b) mostly depends on the effectiveness of using the computer's memory, and in particular on the effectiveness of exploiting the memory cache system. The key issue is the probability that with requesting some data from the main memory the subsequent memory requests will concern some data within a sufficiently local neighborhood (measured by the distance in the address space). The locality of the processed data is not a static feature of the input data, but a dynamic feature that is determined by the input data and the given algorithm. Despite these complications, there is no principal problem in evaluating (b). It is only a matter of technicalities, of the level of detail we want to work with, and last but not least of the time which we wish to invest.

The factor (b) is very important for evaluating the computational cost of direct methods for sparse matrices. Modern algorithms in this area are based on elaborated factorization techniques that only perform arithmetic with nonzero entries and (to a large extent) preserve sparsity during the computation. This typically requires a lot of non-numeric processing and data-handling. Since incomplete factorizations are often used as preconditioners, the factor (b) can play an important role also for iterative methods.

Any iterative method for solving a linear algebraic system $Ax = b$ starts from an initial approximation x_0, and generates a sequence of approximations $x_1, x_2, \ldots$ for the solution x. Hopefully, the sequence *converges*, which means that within an acceptable time the method produces an approximation that is *within a user-specified tolerance* of the solution. A thorough discussion of the term "user-specified tolerance" would justify a separate essay. Without going into details, there are two main issues that have to be resolved by the user of an iterative method: The choice of the measure for the method's performance, and the accuracy level that should be reached.

When measured in flops only, the computational cost of iteratively solving $Ax = b$ is equal to the sum of the flops required for the individual steps until the computation is stopped. The flops for each step can be determined from the description of an iterative algorithm. Many publications contain tables with the number of arithmetic operations

for a single step of different iterative methods. The fact that individual steps are computed only until the required (user-specified) tolerance is reached, however, is a challenge that has further consequences and requires a much deeper thought.

3 Particular Computations and Complexity Theory

Linear algebraic systems arise in a huge variety of applications and have vastly different properties. Often these properties are inherited from the underlying application through the mathematical model and its discretization. The matrix A may be symmetric or unsymmetric, or have other structural features, like symmetry with respect to some inner product or bilinear form. It turns out that an iterative method can be highly efficient for some linear algebraic system, and hopelessly inefficient for another. For three standard Krylov subspace methods for general nonsymmetric matrices (GMRES [55], CGS [58], and CGN [30]), Nachtigal, Reddy, and Trefethen [42] have given a convincing numerical illustration of this fact by a set of eight well chosen (though artificial) examples of linear algebraic systems. The difference in the behavior of the three iterative methods when applied to these systems is striking. It ranges from "all methods good" over "method X wins" and "method X looses" (for each of the three methods) to "all methods bad". This means that in the experiments of Nachtigal, Reddy, and Trefethen the different methods required significantly different numbers of steps until they reached the specified tolerance. The conclusion is that the three algorithms are "genuinely distinct in their behavior" [42, p. 793]. These experiments, and many other similar ones that have been reported in the literature, support the following point of view:

> When we think of the computational cost of iteratively solving a linear algebraic system $Ax = b$, we always consider a *particular* method with a *particular* given data A, b, and x_0.

This viewpoint represents a significant conceptual difference to the classical task of *complexity theory* (of computer science), which is the examination of certain problem classes and the determination of *lower bounds* on the cost of *any* algorithm for the solution of *every* problem in the class[1]. A typical problem class studied in the numerical linear algebra context is "solve a linear algebraic system with a general $n \times n$ matrix". It can be shown that this class has the same *complexity*, usually denoted by n^ω, as the class "multiply two general $n \times n$ matrices"; see, e.g., [9, Chapter 16]. The elusive best possible constant ω is called the *exponent of matrix multiplication*. An algorithm devised by Coppersmith and Winograd in 1990 [12] shows that ω is at most 2.376. On the other hand, ω must be at least 2, since all n^2 entries of the general $n \times n$ matrix must be a part of the computation. A few years ago, group-theoretic methods devised by Cohn, Uhmans, et al. [10, 11] have revived the area of fast matrix multiplication; see [53] for a brief summary. A numerical stability analysis of the corresponding algorithms has been performed in [13]. The recent paper [32] gives a survey of complexity theory and the related stability analysis for matrix multiplication and similar problem classes.

More relevant for our context is the *complexity theory of numerical analysis*. According to Smale, this theory studies *upper bounds* for "the number of arithmetic operations

[1] Discussions of recent developments in complexity theory and of its relationship with computational mathematics can be found, for example, in [5, Section 2.4], [7], and the many references given there.

required to pass from the input to the output of a numerical problem" [57, p. 523]. Here the subject is the examination of basic algorithms rather than problem classes. As an example, consider *direct* methods for solving $Ax = b$ with a general *dense* matrix A of size n. It is well known that the LU factorization with partial pivoting costs $2n^3/3 + \mathcal{O}(n^2)$ flops, while the QR factorization costs $4n^3/3 + \mathcal{O}(n^2)$ flops. When A is symmetric and positive definite, the Cholesky factorization can be applied, which costs $n^3/3 + \mathcal{O}(n^2)$ flops. After each such factorization, a linear algebraic system with an upper triangular matrix needs to be solved, which costs an additional $\mathcal{O}(n^2)$ flops. The computed solution $\widetilde{x}$ is available only after the last computational step is performed. Unlike in iterative methods, there are no intermediate approximations. Consequently, a direct solve of $Ax = b$ with a general dense matrix A and any of the mentioned methods costs $\mathcal{O}(n^3)$ flops. This upper bound on the number of operations is the *complexity* of these methods in Smale's definition; see [57, p. 534].

As Smale points out, "where iterative methods are needed, the complexity theory is not fully developed" [57, p. 533]. Our discussion here is intentionally not directly related to this underdeveloped topic. Rather, as pointed out above, we focus on the *cost of particular computations*. We emphasize that computational cost is then not attributed to a method or its algorithmic realization, but to *application* of the method in the given algorithmic realization to a given particular data.

This view has an important practical motivation. In computational sciences and engineering, the crucial point which often determines success or failure of an approach is *exploiting the hidden inner structural dependencies within the given data*. In short, a particular problem that reflects a real-world situation can be solvable even when the generalized abstract mathematical problem of the same class is not. The success often depends on whether or not the differences between the particular (real-world) problem and the abstract (worst-case) problem can be uncovered and exploited.

4 Convergence in Exact Arithmetic

The purpose of iterative methods for solving linear algebraic equations is to produce an approximation that is within a given tolerance. In real-world applications this tolerance should depend on other stages of the whole solution process. This feature can be effectively used in order to control the cost of the computation. For example, when the linear algebraic system is the result of discretizing a mathematical model, which often also involves estimating parameter values such as material constants, the finite-dimensional algebraic problem need not be solved to a high accuracy. Moreover, continuing the iteration after the algebraic error norm drops below the norm of the discretization error makes little sense. In general, any evaluation of the computational cost of iterative methods must consider their *convergence behavior*.

Let us briefly look at classical iterative methods that are derived from a splitting of the matrix A. Suppose that $A = M - N$ is such a splitting, where M is invertible. Then $Ax = b$ is equivalent to $x = M^{-1}Nx + M^{-1}b$, which yields the *stationary iterative method*

$$x_k = M^{-1}Nx_{k-1} + M^{-1}b \quad \text{for } k = 1, 2, \ldots$$

There are numerous variations and extensions of this general scheme. Many of them are based on relaxation techniques, with the *Successive Overrelaxation Method (SOR)* (introduced simultaneously by Frankel [18] and Young [74] in 1950) forming an important

example. All these methods have in common that the error satisfies $x-x_k = S^k(x-x_0)$, $k = 1, 2, \ldots$, where S is the *iteration matrix* ($S = M^{-1}N$ in the simplest case). Since the kth error is given by the product of the kth power of S and the initial error, the behavior of these iterative methods (after some "transient phase") is *linear*. Much of the theory therefore is concerned with determining the corresponding linear *rate of convergence*. Note that

$$\frac{\|x - x_k\|}{\|x - x_0\|} \leq \|S^k\|$$

(if the vector and matrix norms are consistent). The quantity

$$\sigma \equiv \left(\frac{\|x - x_k\|}{\|x - x_0\|}\right)^{1/k} \leq \|S^k\|^{1/k}$$

is called the *average reduction factor* per iteration of the successive error norms. If $\|S^k\| < 1$, one can define the *average rate of convergence* for k iterations with the matrix S by

$$R(S^k) \equiv -\log(\|S^k\|^{1/k}) = \frac{-\log(\|S^k\|)}{k},$$

so that $\sigma \leq e^{-R(S^k)}$. Then the *asymptotic rate of convergence* is given by

$$\lim_{k\to\infty} R(S^k) \equiv -\log(\rho(S)),$$

where $\rho(S)$ is the spectral radius of the iteration matrix S. Varga's book [67] gives a thorough treatment of the relevant theory. The book is also a historical testimony for the stormy development of SOR-like and the closely related *semi-iterative methods* in the 1950s and early 1960s.

From the 1970s other types of iterative solvers, among them Krylov subspace methods, gained more attention. In a Krylov subspace method we typically have $x_k \in x_0 + \mathcal{K}_k(A, r_0)$ for $k = 1, 2, \ldots$, where $\mathcal{K}_k(A, r_0) = \text{span}\{r_0, Ar_0, \ldots, A^{k-1}r_0\}$ is the kth Krylov subspace generated by the matrix A and the initial residual $r_0 = b-Ax_0$. The approximation x_k is often determined by an orthogonality condition of the form $r_k = b-Ax_k \perp \mathcal{C}_k$, where $\mathcal{C}_k$ is a k-dimensional subspace. This means that Krylov subspace methods are based on a *projection process*. For example, the CG method [30] and the GMRES method [55] are characterized by $\mathcal{C}_k = \mathcal{K}_k(A, r_0)$ and $\mathcal{C}_k = A\mathcal{K}_k(A, r_0)$, respectively. A standard reference that describes these and many other methods is Saad's book [54].

Since the iterates in a Krylov subspace method satisfy

$$x_k \in x_0 + \mathcal{K}_k(A, r_0) = x_0 + A\mathcal{K}_k(A, x - x_0),$$

one can write the error as $x - x_k = p_k(A)(x - x_0)$, where p_k is a polynomial of degree at most k and with $p_k(0) = 1$. Convergence bounds can then be derived using the orthogonality condition $r_k \perp \mathcal{C}_k$; see, e.g., [61] or [38] for a survey. For example, the errors of the CG method satisfy

$$\|x - x_k\|_A = \min_{\substack{p(0)=1 \\ \deg(p)=k}} \|p(A)(x - x_0)\|_A,$$

where $\|y\|_A \equiv (y^T A y)^{1/2}$ is the A-norm of the vector y (the matrix A is assumed symmetric positive definite in the CG context). From this one easily obtains the bound

$$\frac{\|x - x_k\|_A}{\|x - x_0\|_A} \leq \min_{\substack{p(0)=1 \\ \deg(p)=k}} \max_{1 \leq j \leq n} |p(\lambda_j)|, \tag{1}$$

where $0 < \lambda_1 \leq \ldots \leq \lambda_n$ are the eigenvalues of A. A standard approach now is to further bound the right hand side using Chebyshev polynomials on the interval $[\lambda_1, \lambda_n]$. This yields the well known bound

$$\frac{\|x - x_k\|_A}{\|x - x_0\|_A} \leq 2 \left(\frac{\sqrt{\kappa} - 1}{\sqrt{\kappa} + 1} \right)^k, \tag{2}$$

where $\kappa = \lambda_n / \lambda_1$ is the (2-norm) condition number of A. The bound was first derived by Meinardus in 1963 [39].

The right hand side of (2) decreases linearly, which is in spirit similar to the convergence bounds for the stationary and semi-iterative methods described above. Even the derivation of (2) is similar to the classical theory for semi-iterative methods outlined in Varga's book; see, e.g., [67, Chapter 5.1]. It is therefore tempting to call the quantity $(\sqrt{\kappa} - 1)/(\sqrt{\kappa} + 1)$ the *average reduction factor* for k iterations of the CG method, and derive a corresponding *rate of convergence*. Such a "linear asymptotic" approach to the convergence analysis of CG (and Krylov subspace methods in general) applied to a single linear algebraic system $Ax = b$ is, however, questionable.

First, in exact arithmetic well-defined Krylov subspace methods like CG terminate in a finite number of steps. Therefore no limit can be formed, and terms like "rate of convergence" loose their classical meaning; see [36] for an instructive discussion of related issues. Clearly, this situation requires approaches that are substantially different from the analysis of the stationary iterative methods.

Second, unlike stationary methods, Krylov subspace methods can exhibit an intriguingly *nonlinear* behavior. The nonlinearity is apparent from the fact that the Krylov subspaces are built (although not practically computed) using powers of the matrix A. Consequently, Krylov subspace methods are very closely related to moments and moment matching model reduction; see [69] and the recent papers [62,64]. The CG method in particular can be considered a matrix formulation of the Gauss-Christoffel quadrature; see [30, Sections 14–15] and [62, Section 3]. Therefore the right hand side of the bound (2) typically does *not* reflect the actual behavior – neither quantitatively nor qualitatively.

We will demonstrate this with a numerical example: Let A be a diagonal $n \times n$ matrix with diagonal entries $\lambda_1, \lambda_2, \ldots, \lambda_n$ (being the eigenvalues of A), where

$$\lambda_j = \lambda_1 + \frac{j-1}{n-1} (\lambda_n - \lambda_1) \gamma^{n-j}, \quad j = 2, 3, \ldots, n-1, \tag{3}$$

for given $\lambda_n > \lambda_1 > 0$ and $\gamma > 0$. Matrices with spectra of the form (3) were proposed in [60, Section 2], and they have been used for experiments with the CG and Lanczos methods in a number of publications; see, e.g., [23,24,40,41,63]. With $\gamma = 1$ the eigenvalues are uniformly distributed in the interval $[\lambda_1, \lambda_n]$, while $\gamma < 1$ yields a spectrum where most of the eigenvalues form a "cluster" that is located to the right of the point λ_1. For this experiment we choose $n = 34$, $\lambda_1 = 0.1$, $\lambda_n = 100$, and $\gamma = 0.55$. The resulting diagonal matrix A has a (2-norm) condition number of 1000. To set up

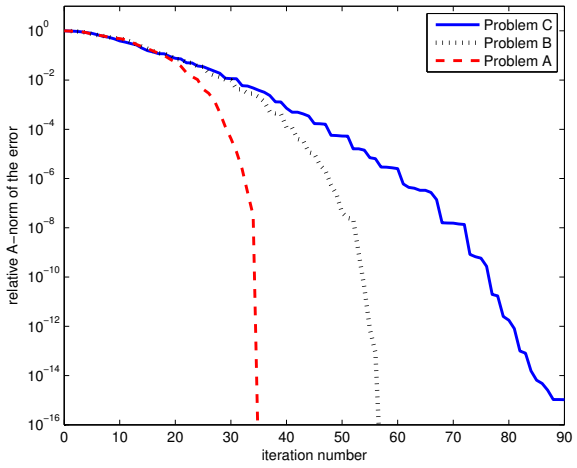

Fig. 1 Convergence of CG in exact arithmetic for three related linear algebraic systems.

a linear algebraic system $Ax = b$ we consider a randomly generated right hand side b with normally distributed entries ranging between 0 and 1.

Remark. *We emphasize that in experiments illustrating the convergence rate of iterative methods applied to practical problems, random right hand sides or initial residuals should be avoided as much as possible. In practice the given right hand side is usually problem-related, and nothing but random. In the example here we are not interested in measuring the performance of the CG method. The right hand side is chosen random in order to demonstrate that the illustrated phenomena are* not *linked to a specific right hand side.*

In addition to the system $Ax = b$ we consider two systems, $By = c$ and $Cz = d$. The $2n \times 2n$ diagonal matrix B is obtained by "splitting" each of the diagonal entries of A into two diagonal entries of B. More precisely, $B = \mathrm{diag}(\mu_1, \ldots, \mu_{2n})$, where

$$\mu_{2j-1} = \lambda_j - \Delta, \quad \mu_{2j} = \lambda_j + \Delta\,, \quad j = 1, 2, \ldots, n\,,$$

and $\Delta = 10^{-10}$ (supposing that $\Delta \ll \lambda_1$, the exact value of Δ is unimportant). The right hand side c is obtained from b by splitting each individual entry of b into two parts of the same absolute value, one corresponding to μ_{2j-1} and one to μ_{2j}, and then normalizing so that $\|c\| = \|b\|$. The system $Cz = d$ is generated analogously, with each of the diagonal entries of A split into *five* close diagonal entries of C.

Fig. 1 shows the convergence of the CG method (measured by $\|x-x_k\|_A/\|x-x_0\|_A$, the relative A-norm of the error) applied to the three linear algebraic systems with the diagonal matrices A, B, and C. In each case the initial guess x_0 is the zero vector. The method has been implemented with double reorthogonalization of the residual vectors. This simulates exact arithmetic (cf. [24]), so the differences in the behavior seen in Fig. 1 do not result from rounding errors.

We first note that the behavior of CG for the three linear algebraic systems is very different – quantitatively and qualitatively – although the three system matrices have

essentially the same condition number. In neither case does the right hand side of (2) describe the actual behavior of the method.

For a more precise explanation of the behavior of the CG method some authors have used the bound (1) instead of (2). This bound is *sharp* in the sense that for each matrix A and each step k there exists an initial error $x - x_0$ so that the bound is attained [20]. Hence the bound completely describes the *worst-case case* behavior of CG for each individual step. In other words, the bound describes an "envelope" for all possible convergence curves that can be attained for the given matrix A. Of course, the behavior for a *particular* right hand side can be very different from the worst case. In discretized boundary value problems, for example, the convergence of an iterative solver may depend strongly on the boundary conditions and outer forces which determine the right hand side; see [37] for an example in the context of the GMRES method.

Axelsson [4] and Jennings [33] have used (1) to develop CG convergence bounds for matrices with outlying eigenvalues[2]. This approach is based on the fact that for any polynomial q of degree $\ell \leq k$ and with $q(0) = 1$ the right hand side of (1) is bounded by

$$\min_{\substack{p(0)=1 \\ \deg(p)=k-\ell}} \max_{1\leq j\leq n} |q(\lambda_j)p(\lambda_j)|.$$

Choosing the polynomial q to have roots at outlying eigenvalues yields a bound that involves a modified condition number (corresponding to the eigenvalues that are not roots of q) and a "penalty term" due to the presence of q. Theoretically, and in case of outliers on the right side of spectrum, this penalty essentially corresponds to the delay of convergence (in number of steps), which is equal to the number of the outlying eigenvalues under consideration. This approach explained, in exact arithmetic, the superlinear convergence behavior of CG in for spectra with right outlying eigenvalues. However, Jennings also noticed that in the presence of eigenvalues on the right side of the spectrum the rate of convergence can be significantly affected by rounding errors [33, p. 72]. This fact crucially limits the practical applicability of the derived bounds. As a consequence, the whole approach considering outlying eigenvalues separately from the rest of the spectrum has been revisited; see our remarks in Section 6 below.

The right hand side of (1) represents a polynomial minimization problem on the discrete set of the eigenvalues of A. In some publications it has been claimed that for the value of this problem it does not matter much whether A has single (well separated) eigenvalues or tight (well separated) clusters of eigenvalues. Our experiment shows what has been explained theoretically by Greenbaum in [21]: Such claim is in general not true. If we replace individual eigenvalues of A by very tight "clusters" (the eigenvalues of B or C), or if the tight "clusters" of eigenvalues of B or C are replaced by single representatives (the eigenvalues of A), then the CG method can exhibit a dramatic change of its convergence behavior. This means that a "bird's eye view" of the spectrum is not sufficient to determine the actual behavior of the method, which makes the quantitative a-priori convergence analysis a very hard (and largely open) problem.

Finally, and perhaps most remarkably, the CG method shows a *nonlinear* staircase-like behavior for the problems with the matrices C and (to a lesser extent) B. Given a behavior at a single step, or a few consecutive steps, nothing can in general be said

[2] In this context one should also mention related contributions of van der Sluis, van der Vorst, Linskog, and others; see the surveys in [43, 60].

about the behavior in the subsequent iterations. Moreover, the experiment indicates that nonlinear behavior of Krylov subspace methods is not related to "undesirable" properties of the matrix A (like indefiniteness, ill conditioning or nonnormality). Apparently, such behavior can occur even for well conditioned, diagonal, and symmetric positive definite matrices. We emphasize that here we deal with *mathematical* phenomena that are not affected by rounding errors.

Of course, in practice nobody uses double reorthogonalization, and close approximations of the exact precision iterates are not available. The computed approximations, and with them any observed convergence curve, will be affected by roundoff. Hence in practice the iteration is stopped only when a computed approximation satisfies the user-specified tolerance. For a meaningful evaluation of the computational cost we therefore additionally have to consider effects of finite precision arithmetic, with their significance being strongly dependent on the particular data. This means that in the evaluation of the computational cost of iterative solvers for $Ax = b$ the analysis of convergence and numerical stability are *inherently linked together*.

In summary, the evaluation of computational cost in computations using modern iterative methods (such as Krylov subspace methods) must be based on the particular data and it must include effects of rounding errors.

5 Effects of Rounding Errors

Let us first look, for comparison, at effects of rounding errors in *direct methods*. As a typical example we consider the LU factorization with column pivoting, which is the most common implementation of the classical Gaussian elimination; see, e.g., [15, Sections 2.3 and 2.4], [19, Section 3.4], [31, Chapter 9], [65, Lectures 20-22], [70, Sections 1.8 and 2.7]. In exact arithmetic, this method computes a factorization of the form

$$PA = LU, \tag{4}$$

where P is a permutation matrix, L is a lower triangular and U is an upper triangular matrix. The column pivoting strategy ensures that $|l_{ij}| \leq 1$ holds for all entries l_{ij} of L. It cannot, however, guarantee a similar property for the entries of U.

A major role in the numerical stability analysis of the algorithm is played by the *element growth factor* γ. This quantity is the ratio of the largest to the smallest element of U (both in absolute value). It is well known (see, e.g., [31, Section 9.4]) that there exist matrices for which γ grows exponentially with the matrix size n (γ can become as large as 2^{n-1}). If γ is very large, then the numerical solution $\widetilde{x}$ of $Ax = b$ computed in finite precision arithmetic is likely to be spoiled by rounding errors. On the other hand, if γ is small or moderate, then the computed solution is likely to be very accurate.

Following Wilkinson [71,73], the results can be stated in the following quantitative form: The computed solution $\widetilde{x}$ satisfies

$$(A + E)\widetilde{x} = b, \tag{5}$$

where

$$\|E\| \leq \gamma\varepsilon\nu_1(n)\|A\|, \qquad \|b - A\widetilde{x}\| \leq \gamma\varepsilon\nu_2(n)\|\widetilde{x}\|, \tag{6}$$

and $\nu_1(n)$, $\nu_2(n)$ are (positive) low-degree polynomial expressions in n (for a detailed review see [31, Chapter 9]). These reflect technicalities of the worst-case rounding error analysis, but do not contain other useful information. A fundamental message

is encoded in the equality (5): The rounding errors of the numerical floating point calculations here appear as (possibly tiny) perturbations E of the original data (here the matrix A), so that the method *in exact arithmetic but applied to the perturbed data* would produce the actually computed solution $\widetilde{x}$ [3].

Computing $\widetilde{x}$ requires the whole factorisation (4) and the subsequent solve of the resulting triangular system. The (local) growth factor γ can be tested during the computation. If the computation is completed, then an a-posteriori argument of probabilistic type shows to which extent the user-specified accuracy of the computed solution is guaranteed. Cases when the computation is not completed due to uncontrollable growth of rounding errors are relatively rare. Apart from these cases, where the incomplete computation does not yield a useful approximation of the solution x, the specified tolerance does *not* affect the computational cost of the algorithm. Hence numerical stability analysis and computational cost are in this case essentially *unrelated*[4].

Let us now turn to *iterative methods*. Here rounding errors can have two main effects: Similar to computations using direct methods they can limit the maximal attainable accuracy of the computed approximate solutions. Unlike in direct methods, they can substantially affect the cost of computing an approximate solution to the user-specified tolerance. In short, rounding errors can delay convergence.

Examining the maximal attainable accuracy of iterative methods is philosophically similar to rounding error analysis of direct methods. However, in the case of iterative methods there is no a-priori known finite sequence of operations that leads to a "final" approximate solution. When performed for a sufficiently large number of steps, an iterative computation reaches a point after which any continuation would lead to no further improvement. At this point the computed error or residual norms stagnate, and sometimes they even start to diverge. In general, estimating the maximal attainable accuracy of iterative methods requires different techniques than those used for direct methods; see, e.g., [23, Section 7.3] and [41, Section 5.4].

The maximal attainable accuracy of an iterative method depends on its algorithmic realization. This means that mathematically equivalent algorithms may achieve a different final accuracy in finite precision. An example is given in [6], where four different iterative algorithms for solving the system of the normal equations $A^TAx = A^Tb$ are studied. When solving this system the explicit formation of the matrix A^TA should be avoided. All algorithms studied in the paper use two matrix-vector multiplications per step; the first one of the form Ap, and then A^Tq. The attainable accuracy of the four mathematically equivalent algorithms can differ by a factor of up to $\kappa(A)$, the condition number of A. Numerical experiments in [6, Section 6] demonstrate this with

[3] This *backward error idea* already appeared in the classical paper of von Neumann and Goldstine [68] (as well as in Turing's [66]), but it was not developed further there. A possible explanation, given by Wilkinson in [72, p. 557], is that the paper [68] is devoted primarily to matrix inverses. Though one can (now!) easily formulate a backward error result for each individual column of A^{-1}, it seems much more difficult to formulate and prove a backward error for the whole inverse (the perturbation matrices will change with each individual column). The results (5)–(6) belong to the foundation of the modern numerical stability analysis in numerical linear algebra. Wilkinson did not invent backward error analysis but he performed it for most existing methods. Through his epochal work, the backward error analysis has been established as a rigorous analytical tool of fundamental importance.

[4] There are important exceptions such as the solution of discrete ill-posed problems arising, e.g., from image reconstruction and restoration; see [17, 27, 28]. Furthermore, in our brief discussion we have omitted some subtle algorithmic details which can increase numerical stability at the price of increasing computational cost. Such important tricks can be (and they typically are) applied adaptively, in particular in case of *sparse* matrices A.

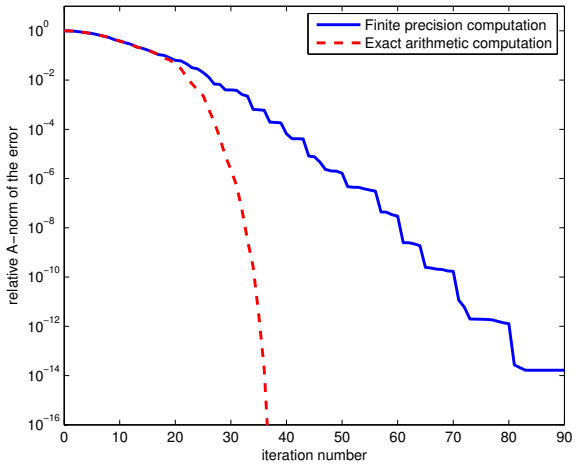

Fig. 2 Delay of convergence of the CG method due to rounding errors.

a matrix that has a condition number of 10^8: for one algorithm (CGLS1) the final Euclidean norm of the error is 10^{-9}, while for another one (CGLS2) this norm never drops below 10^{-1}. Other studies for algorithms that are mathematically equivalent to the CG method can be found in [22, 26].

In practice, it is essential to know whether the maximal attainable accuracy of an iterative method is below or above the user-specified tolerance. In the second case the given problem is *numerically unsolvable* by the given method. This, however, happens quite rarely. As pointed out by Greenbaum [22, p. 550], the maximal attainable accuracy of an iterative method is of less practical importance than the rate of convergence before the final accuracy level is reached. Greenbaum adds that "the analysis of convergence rates in finite precision arithmetic (and sometimes even in exact arithmetic) may be a much more difficult problem." We will now discuss why this is so.

To analyze the convergence rate of an iterative method in finite precision arithmetic one needs to investigate how *individual approximate solutions* are affected by rounding errors. Two different questions may be asked:

First, one can consider a fixed iteration step and ask how much the numerically computed approximate solution at this step differs from its exact precision counterpart. This question is immensely complex. Any attempt to answer it must resolve very difficult open problems. We are not aware of any satisfactory attempt to answer this question.

Second, one can ask whether rounding errors cause any *delay of convergence*. Given a fixed accuracy level, this approach investigates whether the finite precision computation requires more iterations in order to reach the prescribed accuracy than its exact precision counterpart. A numerical illustration is shown in Fig. 2. The dashed line shows the relative A-norm of the error in a computation with CG using double reorthogonalization [24]. The solid line shows the error norms for the same problem and the standard implementation of the CG method by Hestenes and Stiefel [30]. For example, in the exact precision computation (simulated by the dashed line) the error

norm reaches a level of 10^{-6} after 32 steps, while rounding errors cause a delay of approximately 18 additional steps until the standard implementation of CG reaches the same accuracy level.

It turns out that it is possible to quantify the link between propagation of rounding errors and the delay. For the standard implementation of CG it has been shown that the delay of its convergence is determined by the *rank-deficiency* of the computed Krylov subspaces [21,24]. More precisely, the CG method in exact arithmetic computes an orthogonal basis $r_0, r_1, \ldots, r_{k-1}$ of the Krylov subspace $\mathcal{K}_k(A, r_0)$. Due to rounding errors, orthogonality and even linear independence of the computed basis vectors is (quickly) lost. Hence the computed vectors span a subspace of smaller dimension. This loss of dimension (or rank-deficiency) gives exactly the number of steps the solid curve in Fig. 2 lags behind the dashed curve.

Both A and r_0 determine the Krylov subspace, and indeed, the delay of convergence due to rounding errors depends on the initial residual r_0. In principle, for each symmetric positive definite matrix A there is an initial residual r_0 such that the finite precision CG computation with A and r_0 is *not* delayed due to rounding errors until the final step; see [56] and [41, Theorem 4.7]. Note that rank-deficiency and hence delay can occur only when multiple copies of the same eigenvalue of A are approximated in the computed Krylov subspace; see the seminal paper of Paige [47][5]. This can (in theory) be avoided by "properly chosen" initial residuals having some very special properties. However, usually those are of little interest in practical computations. In practice, the computation on the algebraic level must be linked with other parts of solving the problem; see, e.g, [34][6].

The following experiments give further illustrations of the described effects. We apply CG to $Ax = b$ with a symmetric positive definite matrix A of size 48×48. The initial approximation is $x_0 = 0$. The systems are designed to illustrate several different points. They are generated using a technique proposed in [26]; the details are here not important.

Fig. 3 shows the relative residual norm $\|r_k\|/\|r_0\| = \|b - Ax_k\|/\|b\|$, $k = 1, 2, \ldots$, for the exact CG method (computed with double reorthogonalization) and for three finite precision computations using the CG implementations of Hestenes and Stiefel, Rutishauser, and Hageman and Young. Descriptions of these implementations and references to the original literature can be found in [26]. We see that the residual norm

[5] The credit for the development of the rounding error analysis of the Lanczos method and CG belongs to Chris Paige and Anne Greenbaum. Paige laid the foundations for all later developments in his ingenious Ph.D. thesis of 1971 [44], with the results published within the following decade in [45–47]. Greenbaum built upon this work her backward-like theory [21]. An instrumental role was also played by Beresford Parlett, who in addition to his original contributions (see, e.g., [52,51]) influenced many of his students. His effort has led to further dissemination and development of Paige's theory; see the review in [41, Section 4.5].

[6] While the theory on computing over the reals developed by Smale and his collaborators (see, e.g, [7,57]) is fascinating and of unquestionable importance in the theory of computing, its application to practical computational problems is by no means straightforward. The behavior of CG in finite precision may serve as an example: The method is highly nonlinear, and it cannot be viewed as a sequence of linearized steps, contrary, e.g., to Newton's method for solving nonlinear equations; cf. [7]. Therefore the rounding error analysis of CG is highly intriguing. Translating the theory of computing over reals (i.e. exact arithmetic), which would cover the nonlinearity of CG, to finite precision seems to be a difficult problem. This confirms the point that the task of dealing with the numerical stability of algorithms "frequently involves some highly nontrivial mathematics and algorithmic ingenuity"; see the section "The Task of a Numerical Analyst" in [1].

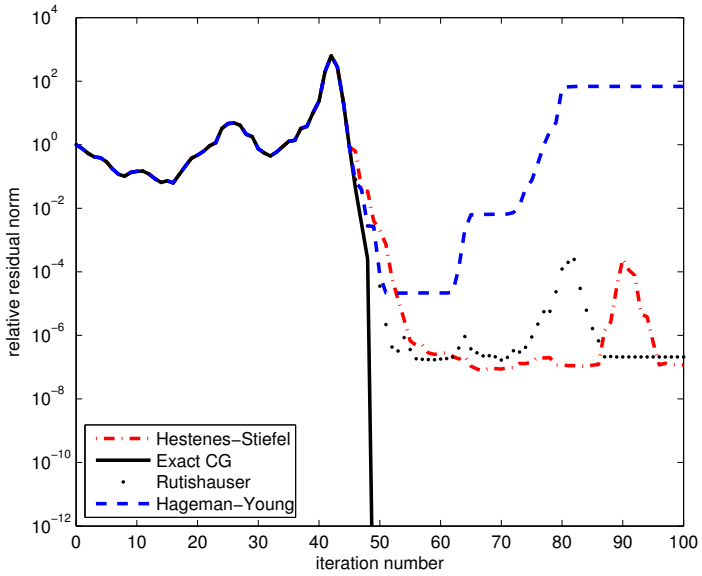

Fig. 3 Relative residual norms of different implementations of the CG method. Note that the relative residual norm can grow for many steps.

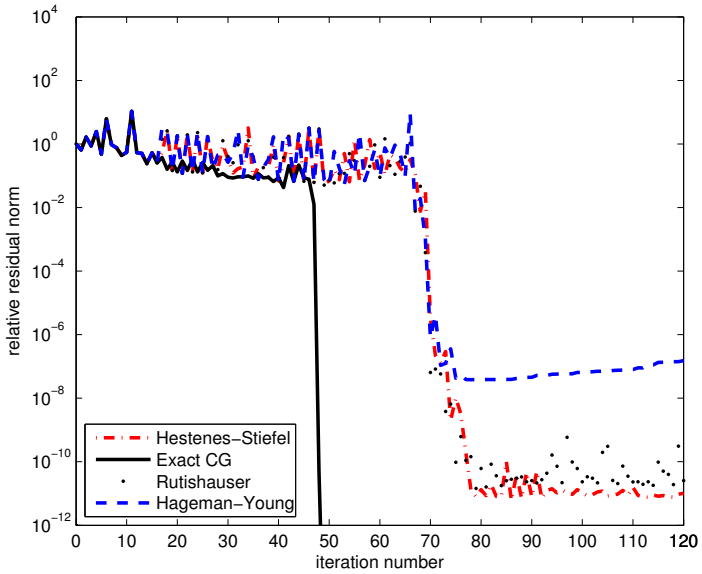

Fig. 4 Relative residual norms of different implementations of the CG method (as in Fig. 3). Note that the relative residual norm can oscillate, and rounding errors can cause a very significant delay.

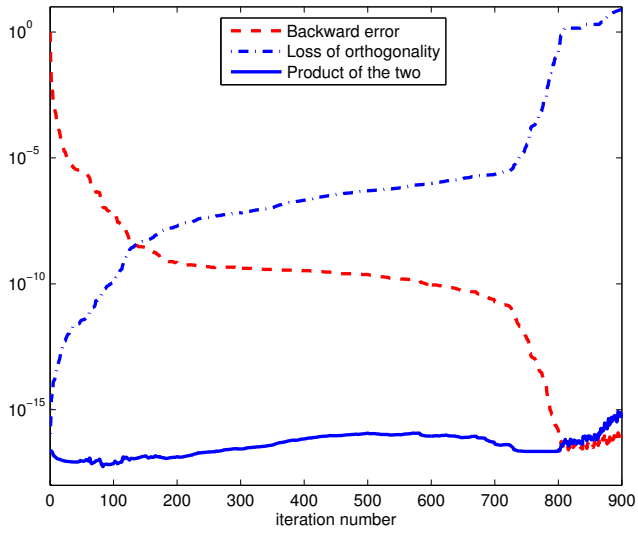

Fig. 5 Throughout the MGS GMRES computation, the product of the normwise backward error and the loss of orthogonality among the MGS Arnoldi vectors is close to machine precision.

is far from monotonic. Moreover, the three implementations differ significantly in their behavior after step $k = 50$ including the maximal attainable accuracy.

Fig. 4 shows the same quantities for CG applied to another system. The residual norm behaves rather erratically for almost 70 iterations, and the delay is much more pronounced than in the previous case. It is worth to comment that there is no simple relationship between the observed phenomena and the condition number of the system matrix. In the first case $\kappa(A) = 1.86 \times 10^{10}$, in the second case $\kappa(A) = 1.97 \times 10^{6}$.

The rounding error analysis of iterative methods applied to nonsymmetric matrices differs from the symmetric case. In particular, the tools of the analysis do not involve eigenvalues, since the relationship between the approximation of eigenvalues and the convergence of Krylov subspace methods is immensely complicated in the nonsymmetric case; see, e.g., [2,25]. An example is the rounding error analysis of the implementation of the GMRES method by Saad and Schultz [55]. This implementation is based on a modified Gram-Schmidt (MGS) version of Arnoldi's method [3] for computing orthogonal bases of Krylov subspaces. Since the orthogonality among the computed basis vectors is lost only gradually in MGS Arnoldi, one intuitively expects that rounding errors cause no significant delay of convergence. This has indeed been observed (with no exception known to us) in practical computations.

In Fig. 5 we show numerical results for MGS GMRES applied to a linear algebraic system with the matrix Sherman2 taken from Matrix Market[7]. For $k = 1, 2, \ldots$, the figure shows the normwise backward error $\|r_k\|/(\|b\| + \|A\|\|x_k\|)$, the loss of orthogonality among the computed MGS Arnoldi basis vectors (stored in the matrix V_k) measured in the Frobenius norm $\|I - V_k^* V_k\|_F$, and the product of the two. Through-

[7] http://math.nist.gov/MatrixMarket/

out the computation this product is very close to machine precision. This means that when the orthogonality is lost completely, the normwise backward error must be close to machine precision, so that the MGS GMRES computation is backward stable. The backward stability of the MGS GMRES implementation has indeed been proven in [48]. The way to the proof has been long and it has revealed many unexpected results and connections; see, e.g., [49,50].

6 A Bigger Picture

The problem we have portrayed above seems intractable: Evaluate the cost of a nonlinear and finite process that is (possibly strongly) affected by rounding errors. However, not all hope is lost. To approach the problem it is sometimes useful to ignore a part of the difficulties, for example by focussing, where appropriate, only on convergence in exact arithmetic, or by considering quantitative description only of a significant part of the rounding error effects.

Of course, one must always avoid simplifications that substantially change (even qualitatively) the investigated phenomena. Frequently met examples for such simplifications in the context of CG are the identification of the method's convergence with the bound (2) and relating a "complexity result" for CG to the condition number of the system matrix (see, e.g., [57, p. 535]), as well as the replacement of tight clusters of eigenvalues by their representatives regardless of the clusters' position within the spectrum. Another example is related to applications of the technique of Axelsson [4] and Jennings [33] described in Section 4 to spectra with large outlying eigenvalues. As mentioned there, in the presence of large outlying eigenvalues rounding errors can cause a substantial delay of convergence. This was demonstrated numerically in [60], and the approach of Axelsson and Jennings was thoroughly revisited by Notay in [43]. In light of these facts, recent results that do not take into account effects of rounding errors and capitalize upon bounds assuming exact arithmetic (see, e.g., [8,59]) should be reexamined.

A possibility for simplifying investigations of difficult problems is to focus on model problems. Two basic reasons for considering a model problem instead of a "practical" problem have been pointed out by Kratzer, Parter, and Steuerwalt [35, pp. 256-257]: The theory can be developed in great generality but is easier to explain for a simplified model ("pedagogical aspect"), and, on the other hand, the theory can be developed only for a model but the results nevertheless give insight into more general problems ("generalizability aspect").

The problems in analysis of Krylov subspace methods are so difficult that one rarely is able to use model problems for pedagogical reasons only. The use of model problems is justified with the second aspect. Moreover, good model problems represent "common structural denominators" [5, p. 30] of real-world applications. The fact that in the evaluation of the computational cost we suggest to consider *particular computations*, and thus often are not interested in the most general or worst-case setting, gives another strong motive for studying model problems.

The key issues for the computational cost of Krylov subspace methods identified above (namely, data-dependency and the inherent link between convergence and numerical stability) are brought to the point in the following quote from an essay by Baxter and Iserles [5, p. 27]:

> "Clearly, good computation requires the algorithm to respond to the data it is producing and change the allocation of computing resources (step size, size of the grid, number of iterations, even the discretization method itself) accordingly. The purpose of computation is not to produce a solution with least error but to produce reliably, robustly and *affordably* a solution which is within user-specified tolerance."

This remark appears to be addressed directly to using Krylov subspace methods or iterative methods in general. Yet, it is made in a discussion of the concept of *adaptivity* and the meaning of the word "computational" in "computational mathematics". This shows that the issues in the context of computational cost of Krylov subspace methods are *typical* for modern computational mathematics, to which the essay by Baxter and Iserles is devoted. According to them, the drive towards "general methods" and "general software" may even be dangerous ("inimical to progress"; cf. [5, p. 30]), since it falls short of the modern challenges of computational mathematics. These include more and more the uncovering and exploitation of the inner structure. Clearly, this requires to focus on the particular rather than the general.

References

1. Allender, E., Bürgisser, P., Kjeldgaard-Pedersen, J., Miltersen, P.B.: On the complexity of numerical analysis. SIAM J. Comput., to appear (2010)
2. Arioli, M., Pták, V., Strakoš, Z.: Krylov sequences of maximal length and convergence of GMRES. BIT **38**(4), 636–643 (1998)
3. Arnoldi, W.E.: The principle of minimized iteration in the solution of the matrix eigenvalue problem. Quart. Appl. Math. **9**, 17–29 (1951)
4. Axelsson, O.: A class of iterative methods for finite element equations. Comput. Methods Appl. Mech. Engrg. **9**(2), 123–127 (1976)
5. Baxter, B.J.C., Iserles, A.: On the foundations of computational mathematics. In: Handbook of numerical analysis, Vol. XI, pp. 3–34. North-Holland, Amsterdam (2003)
6. Björck, Å., Elfving, T., Strakoš, Z.: Stability of conjugate gradient and Lanczos methods for linear least squares problems. SIAM J. Matrix Anal. Appl. **19**(3), 720–736 (1998)
7. Blum, L.: Computing over the reals: where Turing meets Newton. Notices Amer. Math. Soc. **51**(9), 1024–1034 (2004)
8. Boman, E.G., Hendrickson, B., Vavasis, S.: Solving elliptic finite element systems in near-linear time with support preconditioners. SIAM J. Numer. Anal. **46**(6), 3264–3284 (2008)
9. Bürgisser, P., Clausen, M., Shokrollahi, M.A.: Algebraic complexity theory, *Grundlehren der Mathematischen Wissenschaften*, vol. 315. Springer-Verlag, Berlin (1997)
10. Cohn, H., Kleinberg, R., Szegedy, B., Umans, C.: Group-theoretic algorithms for matrix multiplication. In: Proceedings of the 46th Annual IEEE Symposium on Foundations of Computer Science, pp. 379–388. IEEE Computer Society, Pittsburgh, PA (2005)
11. Cohn, H., Umans, C.: A group-theoretic approach to fast matrix multiplication. In: Proceedings of the 44th Annual IEEE Symposium on Foundations of Computer Science, pp. 438–449. IEEE Computer Society, Cambridge, MA (2003)
12. Coppersmith, D., Winograd, S.: Matrix multiplication via arithmetic progressions. J. Symbolic Comput. **9**(3), 251–280 (1990)
13. Demmel, J., Dumitriu, I., Holtz, O., Kleinberg, R.: Fast matrix multiplication is stable. Numer. Math. **106**(2), 199–224 (2007)
14. Demmel, J.W.: Trading off parallelism and numerical stability. In: Linear algebra for large scale and real-time applications (Leuven, 1992), *NATO Adv. Sci. Inst. Ser. E Appl. Sci.*, vol. 232, pp. 49–68. Kluwer Acad. Publ., Dordrecht (1993)
15. Demmel, J.W.: Applied Numerical Linear Algebra. SIAM, Philadelphia, PA (1997)
16. Dongarra, J.J., Duff, I.S., Sorensen, D.C., van der Vorst, H.A.: Numerical Linear Algebra for High-Performance Computers. Software, Environments, and Tools. SIAM, Philadelphia, PA (1998)

17. Engl, H.W., Hanke, M., Neubauer, A.: Regularization of Inverse Problems, *Mathematics and its Applications*, vol. 375. Kluwer Academic Publishers Group, Dordrecht (1996)
18. Frankel, S.P.: Convergence rates of iterative treatments of partial differential equations. Math. Tables and Other Aids to Computation **4**, 65–75 (1950)
19. Golub, G.H., Van Loan, C.F.: Matrix Computations, third edn. Johns Hopkins Studies in the Mathematical Sciences. Johns Hopkins University Press, Baltimore, MD (1996)
20. Greenbaum, A.: Comparison of splittings used with the conjugate gradient algorithm. Numer. Math. **33**(2), 181–193 (1979)
21. Greenbaum, A.: Behavior of slightly perturbed Lanczos and conjugate-gradient recurrences. Linear Algebra Appl. **113**, 7–63 (1989)
22. Greenbaum, A.: Estimating the attainable accuracy of recursively computed residual methods. SIAM J. Matrix Anal. Appl. **18**(3), 535–551 (1997)
23. Greenbaum, A.: Iterative Methods for Solving Linear Systems, *Frontiers in Applied Mathematics*, vol. 17. SIAM, Philadelphia, PA (1997)
24. Greenbaum, A., Strakoš, Z.: Predicting the behavior of finite precision Lanczos and conjugate gradient computations. SIAM J. Matrix Anal. Appl. **13**(1), 121–137 (1992)
25. Greenbaum, A., Strakoš, Z.: Matrices that generate the same Krylov residual spaces. In: Recent advances in iterative methods, *IMA Vol. Math. Appl.*, vol. 60, pp. 95–118. Springer, New York (1994)
26. Gutknecht, M.H., Strakoš, Z.: Accuracy of two three-term and three two-term recurrences for Krylov space solvers. SIAM J. Matrix Anal. Appl. **22**(1), 213–229 (2000)
27. Hansen, P.C.: Rank-Deficient and Discrete Ill-Posed Problems. SIAM Monographs on Mathematical Modeling and Computation. SIAM, Philadelphia, PA (1998)
28. Hansen, P.C.: Discrete Inverse Problems: Insight and Algorithms, *Fundamentals of Algorithms*, vol. 7. SIAM, Philadelphia, PA (2010)
29. Heller, D.: A survey of parallel algorithms in numerical linear algebra. SIAM Rev. **20**(4), 740–777 (1978)
30. Hestenes, M.R., Stiefel, E.: Methods of conjugate gradients for solving linear systems. J. Research Nat. Bur. Standards **49**, 409–436 (1952)
31. Higham, N.J.: Accuracy and Stability of Numerical Algorithms, second edn. SIAM, Philadelphia, PA (2002)
32. Holtz, O., Shorom, N.: Computational complexity and numerical stability of linear problems. In: European Congress of Mathematics, Amsterdam, 14–18 July, 2008, Ran, A.C.M., te Riele, H., Wiegerinck, J., eds., pp. 381-400, European Mathematical Society (2010)
33. Jennings, A.: Influence of the eigenvalue spectrum on the convergence rate of the conjugate gradient method. J. Inst. Math. Appl. **20**(1), 61–72 (1977)
34. Jiránek, P., Strakoš, Z., Vohralík, M.: A posteriori error estimates including algebraic error: computable upper bounds and stopping criteria for iterative solvers. SIAM J. Sci. Comput. **32**(3), 1567–1590 (2010)
35. Kratzer, D., Parter, S.V., Steuerwalt, M.: Block splittings for the conjugate gradient method. Comput. & Fluids **11**(4), 255–279 (1983)
36. Kuijlaars, A.B.J.: Convergence analysis of Krylov subspace iterations with methods from potential theory. SIAM Rev. **48**(1), 3–40 (2006)
37. Liesen, J., Strakoš, Z.: GMRES convergence analysis for a convection-diffusion model problem. SIAM J. Sci. Comput. **26**(6), 1989–2009 (2005)
38. Liesen, J., Tichý, P.: Convergence analysis of Krylov subspace methods. GAMM Mitt. Ges. Angew. Math. Mech. **27**(2), 153–173 (2004)
39. Meinardus, G.: Über eine Verallgemeinerung einer Ungleichung von L. V. Kantorowitsch. Numer. Math. **5**, 14–23 (1963)
40. Meurant, G.: The Lanczos and conjugate gradient algorithms, *Software, Environments, and Tools*, vol. 19. SIAM, Philadelphia, PA (2006)
41. Meurant, G., Strakoš, Z.: The Lanczos and conjugate gradient algorithms in finite precision arithmetic. Acta Numerica **15**, 471–542 (2006)
42. Nachtigal, N.M., Reddy, S.C., Trefethen, L.N.: How fast are nonsymmetric matrix iterations? SIAM J. Matrix Anal. Appl. **13**(3), 778–795 (1992)
43. Notay, Y.: On the convergence rate of the conjugate gradients in presence of rounding errors. Numer. Math. **65**(3), 301–317 (1993)
44. Paige, C.C.: The computation of eigenvalues and eigenvectors of very large and sparse matrices. Ph.D. thesis, London University, London, England (1971)
45. Paige, C.C.: Computational variants of the Lanczos method for the eigenproblem. J. Inst. Math. Appl. **10**, 373–381 (1972)

46. Paige, C.C.: Error analysis of the Lanczos algorithm for tridiagonalizing a symmetric matrix. J. Inst. Math. Appl. **18**(3), 341–349 (1976)
47. Paige, C.C.: Accuracy and effectiveness of the Lanczos algorithm for the symmetric eigenproblem. Linear Algebra Appl. **34**, 235–258 (1980)
48. Paige, C.C., Rozložník, M., Strakoš, Z.: Modified Gram-Schmidt (MGS), least squares, and backward stability of MGS-GMRES. SIAM J. Matrix Anal. Appl. **28**(1), 264–284 (2006)
49. Paige, C.C., Strakoš, Z.: Bounds for the least squares distance using scaled total least squares. Numer. Math. **91**(1), 93–115 (2002)
50. Paige, C.C., Strakoš, Z.: Residual and backward error bounds in minimum residual Krylov subspace methods. SIAM J. Sci. Comput. **23**(6), 1898–1923 (2002)
51. Parlett, B.N.: The rewards for maintaining semi-orthogonality among Lanczos vectors. J. Numer. Linear Algebra Appl. **1**(2), 243–267 (1992)
52. Parlett, B.N.: The symmetric eigenvalue problem, *Classics in Applied Mathematics*, vol. 20. SIAM, Philadelphia, PA (1998). Corrected reprint of the 1980 original
53. Robinson, S.: Toward an optimal algorithm for matrix multiplication. SIAM News **38**(9) (2005)
54. Saad, Y.: Iterative Methods for Sparse Linear Systems, second edn. SIAM, Philadelphia, PA (2003)
55. Saad, Y., Schultz, M.H.: GMRES: a generalized minimal residual algorithm for solving nonsymmetric linear systems. SIAM J. Sci. Statist. Comput. **7**(3), 856–869 (1986)
56. Scott, D.S.: How to make the Lanczos algorithm converge slowly. Math. Comp. **33**(145), 239–247 (1979)
57. Smale, S.: Complexity theory and numerical analysis. Acta Numer. **6**, 523–551 (1997)
58. Sonneveld, P.: CGS, a fast Lanczos-type solver for nonsymmetric linear systems. SIAM J. Sci. Statist. Comput. **10**(1), 36–52 (1989)
59. Spielman, D.A., Woo, J.: A note on preconditioning by low-stretch spanning trees. CoRR **abs/0903.2816** (2009)
60. Strakoš, Z.: On the real convergence rate of the conjugate gradient method. Linear Algebra Appl. **154/156**, 535–549 (1991)
61. Strakoš, Z.: Convergence and numerical behaviour of the Krylov space methods. In: Algorithms for large scale linear algebraic systems (Gran Canaria, 1996), *NATO Adv. Sci. Inst. Ser. C Math. Phys. Sci.*, vol. 508, pp. 175–196. Kluwer Acad. Publ., Dordrecht (1998)
62. Strakoš, Z.: Model reduction using the Vorobyev moment problem. Numer. Algorithms **51**(3), 363–379 (2009)
63. Strakoš, Z., Tichý, P.: On error estimation in the conjugate gradient method and why it works in finite precision computations. Electron. Trans. Numer. Anal. **13**, 56–80 (2002)
64. Strakoš, Z., Tichý, P.: On efficient numerical approximation of the bilinear form c^*A^b. submitted to SIAM J. Sci. Comput. (2010)
65. Trefethen, L.N., Bau III, D.: Numerical Linear Algebra. SIAM, Philadelphia, PA (1997)
66. Turing, A.M.: Rounding-off errors in matrix processes. Quart. J. Mech. Appl. Math. **1**, 287–308 (1948)
67. Varga, R.S.: Matrix Iterative Analysis. Prentice-Hall Inc., Englewood Cliffs, N.J. (1962)
68. Von Neumann, J., Goldstine, H.H.: Numerical inverting of matrices of high order. Bull. Amer. Math. Soc. **53**, 1021–1099 (1947)
69. Vorobyev, Y.V.: Methods of Moments in Applied Mathematics. Translated from the Russian by Bernard Seckler. Gordon and Breach Science Publishers, New York (1965)
70. Watkins, D.S.: Fundamentals of Matrix Computations, second edn. Wiley-Interscience, New York (2002)
71. Wilkinson, J.H.: Error analysis of direct methods of matrix inversion. J. Assoc. Comput. Mach. **8**, 281–330 (1961)
72. Wilkinson, J.H.: Modern error analysis. SIAM Rev. **13**, 548–568 (1971)
73. Wilkinson, J.H.: Rounding Errors in Algebraic Processes. Dover Publications Inc., New York (1994). Reprint of the 1963 original published by Prentice-Hall, Englewood Cliffs, NJ
74. Young, D.M.: Iterative methods for solving partial difference equations of elliptic type. Ph.D. thesis, Harvard University, Cambridge, MA (1950)