Exact arithmetic as a tool for convergence assessment of the IRM-CG method

Dvornik, Josip; Jaguljnjak Lazarević, Antonia; Lazarević, Damir; Uroš, Mario

Source / Izvornik: **Heliyon, 2020, 6, 1 - 7**

Journal article, Published version
Rad u časopisu, Objavljena verzija rada (izdavačev PDF)

https://doi.org/10.1016/j.heliyon.2020.e03225

Permanent link / Trajna poveznica: https://urn.nsk.hr/urn:nbn:hr:237:812360

Rights / Prava: In copyright / Zaštićeno autorskim pravom.

Download date / Datum preuzimanja: **2025-01-05**

Repository / Repozitorij:

Repository of the Faculty of Civil Engineering, University of Zagreb

Heliyon 6 (2020) e03225

Contents lists available at ScienceDirect

Heliyon

journal homepage: www.cell.com/heliyon

Research article

Exact arithmetic as a tool for convergence assessment of the IRM-CG method

Josip Dvornik [a], Antonia Jaguljnjak Lazarevic [b], Damir Lazarevic [a,*], Mario Uros [a]

[a] University of Zagreb, Faculty of Civil Engineering, Kaciceva 26, Zagreb, 10000, Croatia
[b] University of Zagreb, Faculty of Mining, Geology and Petroleum Engineering, Pierottijeva 6, Zagreb, 10000, Croatia

ARTICLE INFO

Keywords:
Applied mathematics
Exact arithmetic
Benchmark
Rounding error
Iterated Ritz Method
Conjugate gradient method

ABSTRACT

Using exact computer arithmetic, it is possible to determine the (exact) solution of a numerical model without any rounding error. For such purposes, a corresponding system of equations should be exactly defined, either directly or by rationalising the numerically given input data. In the latter case, there is an initial round-off error, but this does not propagate during the solution process. If this system is exactly solved first and then using floating-point arithmetic, the convergence of the numerical method easily follows. As an example, the IRM–CG, which is an alternative to the Conjugate Gradient (CG) method and a special case of the more general Iterated Ritz Method (IRM), is verified. The method is not based on conjugacy; therefore, restarting strategies are not required, while an overrelaxation factor and preconditioning like techniques could be easily adopted. The exact arithmetic approach is introduced by means of a simple example and is then applied to small structural engineering problems. The perturbation of the displacement increment and the different condition numbers of the system matrix are used to check the stability of the algorithm. Interestingly, a large difference in the number of steps between the exact and numerical approaches is detected, even for well-conditioned systems. According to the tests, the IRM-CG may be considered to be stable and useful for not well-posed or well-posed but ill-conditioned models. Because the computer demands and execution time grow enormously with the number of unknowns using this strategy, three possibilities for larger systems are also provided.

1. Introduction: The Iterated Ritz Method

The Iterated Ritz Method (IRM) is an iterative approach to solving the symmetric positive definite (SPD) system $\mathbf{Ax} = \mathbf{b}$ based on successive minimisation of the corresponding energy (the quadratic function)

$$f(\mathbf{x}) = \frac{1}{2}\mathbf{x}^{\mathrm{T}}\mathbf{A}\mathbf{x} - \mathbf{x}^{\mathrm{T}}\mathbf{b} \tag{1}$$

inside a small subspace formed at each step [1]. The main strategy is to present solution increments by the Ritz idea:

$$\mathbf{p}_{(i)} = \mathbf{\Phi}_{(i)}\mathbf{a}_{(i)} \tag{2}$$

where $\mathbf{\Phi}_{(i)} = \left[\boldsymbol{\phi}_{1,(i)}\boldsymbol{\phi}_{2,(i)}\ldots\boldsymbol{\phi}_{m,(i)}\right]$ is a matrix of linearly independent coordinate vectors (that form subspace) and $\mathbf{a}_{(i)}$ is the vector of corresponding coefficients. Using this approach, the energy decrement achieved by (2), is also presented in the quadratic form

$$\Delta f(\mathbf{a}_{(i)}) = \frac{1}{2}\mathbf{a}_{(i)}^{\mathrm{T}}\overline{\mathbf{A}}_{(i)}\mathbf{a}_{(i)} - \mathbf{a}_{(i)}^{\mathrm{T}}\overline{\mathbf{r}}_{(i)} \tag{3}$$

where $\overline{\mathbf{A}}_{(i)} = \mathbf{\Phi}_{(i)}^{\mathrm{T}}\boldsymbol{A}\mathbf{\Phi}_{(i)}$ and $\overline{\mathbf{r}}_{(i)} = \mathbf{\Phi}_{(i)}^{\mathrm{T}}\mathbf{r}_{(i)}$ are the SPD generalised (Ritz) matrix and residual vector, both of order m. Minimisation of (3) leads to a system of equations that should be solved at each step:

$$\overline{\mathbf{A}}_{(i)}\mathbf{a}_{(i)} = \overline{\mathbf{r}}_{(i)} \tag{4}$$

This is a very small system, because only several coordinate vectors are applied ($m \ll n$). The solution is used to find the increment in (2), and $\mathbf{x}_{(i+1)} = \mathbf{x}_{(i)} + \omega\mathbf{p}_{(i)}$ is subsequently updated.

Coefficient $\omega \in (0, 2)$ is the relaxation factor, known from the Successive Overrelaxation method, and might improve the convergence. Some optimal ω exists, but they vary for every unknown and solution step. Moreover, the determination is usually more 'expensive' than the benefit of possible improvement. Generally, ω may be guessed by intuition or experience, and kept constant during the solution process.

The residual is recursively defined as $\mathbf{r}_{(i+1)} = \mathbf{r}_{(i)} - \omega\mathbf{A}\mathbf{p}_{(i)}$ and should always be corrected after some (say k) number of steps using the equilibrium

* Corresponding author.
E-mail address: damir@grad.hr (D. Lazarevic).

https://doi.org/10.1016/j.heliyon.2020.e03225
Received 22 July 2019; Received in revised form 14 October 2019; Accepted 10 January 2020
2405-8440/© 2020 The Author(s). Published by Elsevier Ltd. This is an open access article under the CC BY license (http://creativecommons.org/licenses/by/4.0/).

relation $\mathbf{r}_{(i+1)} = \mathbf{b} - \mathbf{A}\mathbf{x}_{(i+1)}$, because of accumulated round-off errors. The process is terminated after the convergence criterion is reached, i.e. $\| \mathbf{r}_{(i)} \|_2 \leq \varepsilon \| \mathbf{r}_{(0)} \|_2$, where ε is a very small positive number. To avoid useless calculations (if the algorithm has trouble converging) the maximum number of steps (n_{max}) should also be defined. Simple pseudocode, with sequence of instructions common for iterative solution methods, is given by the Algorithm 1.

Therefore, at each step, coordinate vectors spanning the subspace are created, within which the energy of the system is reduced. This is why the small system (4) needs to be solved (most often by some direct solver). If $\omega = 1$ it is the largest reduction (local energy minimum), which is not necessarily optimal for global convergence. The procedure leads to a gradient class solution method that combines iterative and direct solution strategies. If the iterative process is convergent, the sum of small-system solutions approaches the large (original) system solution, and the sum of small-system energies monotonically decreases and approaches the minimum of the large system [2].

For the convergence of the IRM one coordinate vector not orthogonal to the current residual is sufficient. It can be $\mathbf{r}_{(i+1)}$ itself, or multiplied by some SPD matrix. A previous solution increment $\mathbf{p}_{(i)}$ contributes to faster convergence, and it is also frequently used. This vector is known from the previous step, therefore it 'costs' nothing, contrary to other vectors that should be generated somehow. Such vectors are of different efficiency, but are very freely selected – they should only be linearly independent.

The main difficulty is that the number of potentially good coordinate vectors is very large. Unfortunately, adequate criteria for the selection of generally efficient vectors is not known and the background theory is not well developed. For one group of models some vectors work fine but respond badly for another. Under such conditions, an efficient (small, fast and model independent) subspace would be of great significance. Obviously, a novel and promising iterative solver is established, but further theoretical and practical research is needed.

The IRM can also be considered as a generalisation of some iterative methods [3]. Depending on the choice of coordinate vectors, solvers can be represented or interpreted as special cases of this approach. Furthermore, it is possible to combine good properties of several methods simultaneously. If appropriate vectors are selected, convergence should proceed faster than using any single method considered. Here, an improved conjugate gradient (CG) algorithm (named the IRM-CG) is briefly presented [4], which is applicable to sparse and large SPD systems arising in many physics applications [5, 6, 7, 8].

Algorithm 1. The basic IRM algorithm.

Require: $\mathbf{A}$, $\mathbf{b}$, $\mathbf{x}_{(0)}$, ω, ε, k, n_{max} {usually $\mathbf{x}_{(0)} \leftarrow 0$}
Ensure: $\mathbf{x}_{(i+1)}$ {close to $\mathbf{x}$}
1: $i \leftarrow 0$ {initialisation: the Steepest Descent}
2: $\mathbf{r}_{(0)} \leftarrow \mathbf{b} - \mathbf{A}\mathbf{x}_{(0)}$ {the initial residual}
3: $q \leftarrow \mathbf{r}_{(0)}^T \mathbf{r}_{(0)} / (\mathbf{r}_{(0)}^T \mathbf{A} \mathbf{r}_{(0)})$ {the initial step length}
4: $\mathbf{p}_{(0)} \leftarrow q\mathbf{r}_{(0)}$ {the initial solution increment}
5: **while** $(\| \mathbf{r}_{(i)} \|_2 > \varepsilon \| \mathbf{r}_{(0)} \|_2) \wedge (i \leq n_{max})$ **do** {the Iterated Ritz method}
6: generate $[\boldsymbol{\phi}_{1,(i)} \boldsymbol{\phi}_{2,(i)} \ldots \boldsymbol{\phi}_{m,(i)}]$ {define $\boldsymbol{\Phi}_{(i)}$ by columns}
7: $\overline{\mathbf{A}}_{(i)} \leftarrow [\boldsymbol{\phi}_{1,(i)} \boldsymbol{\phi}_{2,(i)} \ldots \boldsymbol{\phi}_{m,(i)}]^T \mathbf{A} [\boldsymbol{\phi}_{1,(i)} \boldsymbol{\phi}_{2,(i)} \ldots \boldsymbol{\phi}_{m,(i)}]$ {define $\overline{\mathbf{A}}_{(i)} = \boldsymbol{\Phi}_{(i)}^T \mathbf{A} \boldsymbol{\Phi}_{(i)}$}
8: $\overline{\mathbf{r}}_{(i)} \leftarrow [\boldsymbol{\phi}_{1,(i)} \boldsymbol{\phi}_{2,(i)} \ldots \boldsymbol{\phi}_{m,(i)}]^T \mathbf{r}_{(i)}$ {define $\overline{\mathbf{r}}_{(i)} = \boldsymbol{\Phi}_{(i)}^T \mathbf{r}_{(i)}$}
9: $\mathbf{a}_{(i)} \leftarrow \overline{\mathbf{A}}_{(i)}^{-1} \overline{\mathbf{r}}_{(i)}$ {solve the small system $\overline{\mathbf{A}}_{(i)} \mathbf{a}_{(i)} = \overline{\mathbf{r}}_{(i)}$}
10: $\mathbf{p}_{(i)} \leftarrow [\boldsymbol{\phi}_{1,(i)} \boldsymbol{\phi}_{2,(i)} \ldots \boldsymbol{\phi}_{m,(i)}] \mathbf{a}_{(i)}$ {calculate $\mathbf{p}_{(i)} = \boldsymbol{\Phi}_{(i)} \mathbf{a}_{(i)}$}
11: $\mathbf{x}_{(i+1)} \leftarrow \mathbf{x}_{(i)} + \omega \mathbf{p}_{(i)}$ {solution update}
12: **if** i mod $k \neq 0$ **then** {update residual every k steps}
13: $\mathbf{r}_{(i+1)} \leftarrow \mathbf{r}_{(i)} - \omega \mathbf{A} \mathbf{p}_{(i)}$ { from the recursive relation }
14: **else**
15: $\mathbf{r}_{(i+1)} \leftarrow \mathbf{b} - \mathbf{A}\mathbf{x}_{(i+1)}$ { from the equilibrium equation }
16: **end if** {end of residual update}
17: $i \leftarrow i + 1$ { update the step counter }
18: **end while** {end of the Iterated Ritz method}

2. Non-recursive CG-like algorithm without the need to restart

The IRM-CG also starts with the Steepest Descent step. Other steps are executed using a simple alternative to the CG simulated by the IRM with two coordinate vectors: $\mathbf{r}_{(i+1)}$ and $\mathbf{p}_{(i)}$. Vectors span a two-dimensional subspace. At each step, a system of two equations is solved and a local energy minimum within that plane is found (Algorithm 2). As in the CG, only one matrix-vector multiplication is required per step (line 13, Algorithm 2), with appropriate transformations.

Local energy minima are numerically 'exact', contrary to the standard CG where the solution of two equations is sought by the equivalent recursive $\mathbf{A}$-orthogonalisation. Because of the accumulation of round-off errors, orthogonality only really exists for a few adjacent vectors and convergence difficulties are present for large ill-conditioned systems. Many restart [9] and preconditioning techniques [10, 11, 12, 13, 14] improve convergence.

In the IRM-CG, error in $\mathbf{A}$-orthogonality also exists, but orthogonality is not used nor accumulated during an iterative process. Therefore, inherited errors decrease, though non-exact arithmetic (as in every numerical process) affects (but does not threaten) the convergence. As a consequence, restarting of the IRM-CG is not needed. Further, preconditioning-like techniques [15] can be adopted easily [4].

One more advantage of this formulation is natural adoption of the relaxation factor. Using $\omega \neq 1$, $\mathbf{A}$-orthogonality is lost, which contradicts the standard CG algorithm, which is just based on the $\mathbf{A}$-orthogonality. Therefore, only preconditioners of the classical CG could be modified by ω.

3. The IRM-CG is equivalent to the CG

These methods are equivalent, and it is possible for them to be interchanged [4]. Each step may be performed by the CG or the IRM-CG, regardless of how the earlier steps were realised. If the CG is preferred, we suggest that a single IRM-CG step is occasionally executed, before the orthogonality error becomes too large. This could be termed 'refresh' rather than 'restart'.

If exact arithmetic is considered, the IRM-CG and the CG have an identical sequence of step results. The exact solution is obtained after the total number of m steps, where m is the number of different 'active' eigenvalues [16]. If $\mathbf{b}$ is represented as a sum of eigenvectors $\mathbf{v}_j$, i.e. $\mathbf{b} = \sum a_j \mathbf{v}_j$, eigenvectors (and corresponding eigenvalues) with $a_j \neq 0$ are 'active' ('inactive' otherwise). Of course, m can be found only if all n eigenpairs are detected. Multiple eigenvalues should be counted as one, and 'inactive' eigenvalues are not counted at all. This comment is of less practical significance, because the IRM-CG is interesting as an iterative, not as a direct solution method [17]. It can be concluded that the IRM-CG (and the CG) can be used in the study of linear switched systems [18].

4. Simple illustrative example

The above considerations are easily proved by the exact arithmetic approach [4], which is extended to the linear structural models that are exactly defined by the direct stiffness method. This is demonstrated by the simple example (Figure 1). Elements of $\mathbf{A}$ and $\mathbf{b}$ are rational numbers and integers. A system of equations is solved by the CG and the IRM-CG with the exact arithmetic (subscript E).

Algorithm 2. The basic IRM-CG algorithm.

Require: $\mathbf{A}$, $\mathbf{b}$, $\mathbf{x}_{(0)}$, ω, ε, k, n_{max} {usually $\mathbf{x}_{(0)} \leftarrow 0$}
Ensure: $\mathbf{x}_{(i+1)}$ {close to $\mathbf{x}$}
1: $i \leftarrow 0$ {initialisation: the Steepest Descent}
2: $\mathbf{r}_{(0)} \leftarrow \mathbf{b} - \mathbf{A}\mathbf{x}_{(0)}$ {the initial residual}
3: $q \leftarrow \mathbf{r}_{(0)}^T \mathbf{r}_{(0)} / (\mathbf{r}_{(0)}^T \mathbf{A} \mathbf{r}_{(0)})$ {the initial step length}
4: $\mathbf{p}_{(0)} \leftarrow q\mathbf{r}_{(0)}$ {the initial solution increment}

(*continued on next page*)

(*continued*)

5: $\boldsymbol{\beta}_{(0)} \leftarrow \mathbf{A}\mathbf{p}_{(0)}$ {the new initialisation}
6: **while** $(\| \mathbf{r}_{(i)}\|_2 > \varepsilon \| \mathbf{r}_{(0)}\|_2) \wedge (i \leq n_{max})$ **do** {the IRM-CG method}
7: $\mathbf{x}_{(i+1)} \leftarrow \mathbf{x}_{(i)} + \omega \mathbf{p}_{(i)}$ {solution update}
8: **if** i mod $k \neq 0$ **then** {update residual every k steps}
9: $\mathbf{r}_{(i+1)} \leftarrow \mathbf{r}_{(i)} - \omega \boldsymbol{\beta}_{(i)}$ {from the recursive relation}
10: **else**
11: $\mathbf{r}_{(i+1)} \leftarrow \mathbf{b} - \mathbf{A}\mathbf{x}_{(i+1)}$ {from the equilibrium equation}
12: **end if** {end of residual update}
13: $\boldsymbol{\alpha}_{(i)} \leftarrow \mathbf{A}\mathbf{r}_{(i+1)}$ {sole matrix-vector multiplication}
14: $\overline{\mathbf{A}}_{(i)} \leftarrow [\mathbf{r}_{(i+1)} \ \mathbf{p}_{(i)}]^T [\boldsymbol{\alpha}_{(i)} \ \boldsymbol{\beta}_{(i)}]$ {$\overline{\mathbf{A}}_{(i)}$ is symmetric: $\mathbf{r}^T_{(i+1)}\boldsymbol{\beta}_{(i)} = \mathbf{p}^T_{(i)}\boldsymbol{\alpha}_{(i)}$}
15: $\overline{\mathbf{r}}_{(i)} \leftarrow [\mathbf{r}^T_{(i+1)}\mathbf{r}_{(i+1)} \ \omega \mathbf{r}^T_{(i+1)}\mathbf{p}_{(i)}]^T$ {if $\omega = 1$ the second term is zero}
16: $\mathbf{a}_{(i)} \leftarrow \overline{\mathbf{A}}^{-1}_{(i)}\overline{\mathbf{r}}_{(i)}$ {solve the small system $\overline{\mathbf{A}}_{(i)}\mathbf{a}_{(i)} = \overline{\mathbf{r}}_{(i)}$}
17: $\mathbf{p}_{(i+1)} \leftarrow [\mathbf{r}_{(i+1)} \ \mathbf{p}_{(i)}]\mathbf{a}_{(i)}$ {update the solution increment}
18: $\boldsymbol{\beta}_{(i+1)} \leftarrow [\boldsymbol{\alpha}_{(i)} \ \boldsymbol{\beta}_{(i)}]\mathbf{a}_{(i)}$ {update $\boldsymbol{\beta}$}
19: $i \leftarrow i + 1$ {update the step counter}
20: **end while** {end of the IRM-CG method}

In numerical computation, the set of real numbers is in fact approximated by the set of rational numbers represented to a fixed number of decimal digits chosen in advance [19, 20]. On the contrary, by using the exact arithmetic approach we always manipulate with the whole numbers, fractions, constants like π and e, the nth root of such numbers and so on. Generally, we operate with symbolic expressions. Compared to numerical computation, this is a demanding strategy which may result in very complicated and lengthy output, but it is not affected by the rounding error. Such error is completely avoided.

Both algorithms are realised with the Wolfram Language [21] used in the Wolfram Mathematica, version 11.3 [22]. Briefly, if all input data are given as fractions and integers, the Wolfram Language retains the rational arithmetic during execution. Results at every step, (such as residuals, displacements, reactions or internal forces) are also exact and are the same by both methods. The system has 8 DoF and 6 active eigenvectors, because a_3 and a_7 are zero (as an integer). Therefore, six steps are needed to obtain the solution. After initialisation and $\| \mathbf{r}_{(0)}\|_2 = 1$, remaining relative residual norms are:

$$\frac{1}{2},\ 3\frac{\sqrt{15881}}{814},\ 48\frac{\sqrt{1134618045}}{20474189},\ 24\frac{\sqrt{\cdots}}{\cdots},\ 2306892210264\frac{\sqrt{\cdots}}{\cdots},\ 0 \qquad (5)$$

The fourth and the fifth norm contain very large integers (marked with dots) and are not included here. The sixth norm is exact zero. Final displacement components are

$$\mathbf{x}_{(6)} = \frac{1}{a}\left[1440 \ \ 11440004167 \ \ 3840 \ \ 0 \ \ 22880008334 \ \ -1440 \ \ \frac{\cdots}{64} \ \ \cdots\right]^{\mathrm{T}} \qquad (6)$$

where $a = 39440013077$. The last two values are also too large to show them here. It is worth noting that, because of the beam symmetry condition, the fourth DoF (rotation of the second joint) is also the integer zero. Even for a such small system precision deteriorates, if a double-precision arithmetic is used (subscript DP). Therefore, the solution is found after two more (eight) steps, providing $\varepsilon \leq 10^{-10}$ is satisified (see Figure 1).

5. Check of the algorithm stability

For such a small example, an exact check of the algorithm stability is possible. Because the number of operations is small and the rounding error is negligible, both methods are intentionally perturbed by $\delta = 1$[4], added (for example) to the seventh component of $\mathbf{p}_{(1)}$ (at the end of the first step):

$$p_{7,(1)} \leftarrow p_{7,(1)} + \delta \qquad (7)$$

This effect is similar to the loss of orthogonality, which is common for the iterative methods, if applied to large and ill-conditioned systems. Then, exact arithmetic is again used to obtain the solution. Even with the perturbation, IRM–CG gives an exact result, because δ is in the loading (residual) direction and therefore lies in the plane spanned by the coordinate vectors. In this particular case, the behaviour of the IRM-CG is as if $\delta = 0$ (Figure 2).

Roughly, if δ is split into two components at each step, the one inside, and the other orthogonal to the plane, the first component is exactly resolved and does not produce inherited error. Using CG, both components cause propagation of error and two more steps are needed to obtain the solution. Similar behaviour is noticed in the double-precision environment (added to Figure 2).

Such perturbations may be 'induced' by the program code of any solution method. Convergence is then verified by comparing the results obtained with exact and floating point implementation.

6. More general examples

Consider a larger model – minimally supported (externally statically determinate) cube, loaded with the unit force at the top (Figure 3a). The cube is discretised by a single Lagrangian $\mathscr{C}^0$ finite element with 192 DoF. Inner nodes are not statically condensed (but without loss of generality they can be) and the stiffness matrix is resolved exactly, using rational numbers [23]. Corner stiffnesses are defined similarly and are

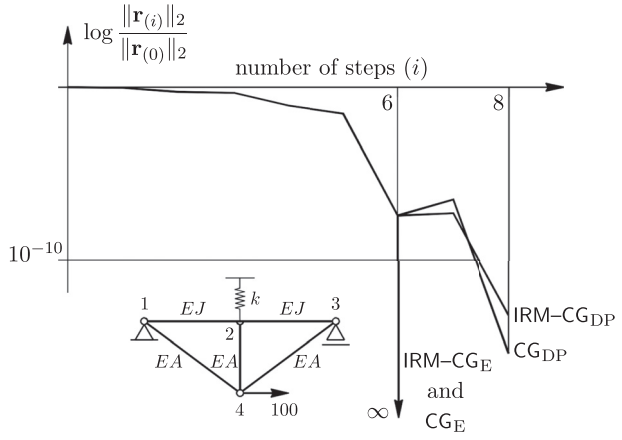

Figure 1. Relative residual norms of the exact (E) and double-precision (DP) implementation of the IRM–CG and the CG applied to simple structural system.

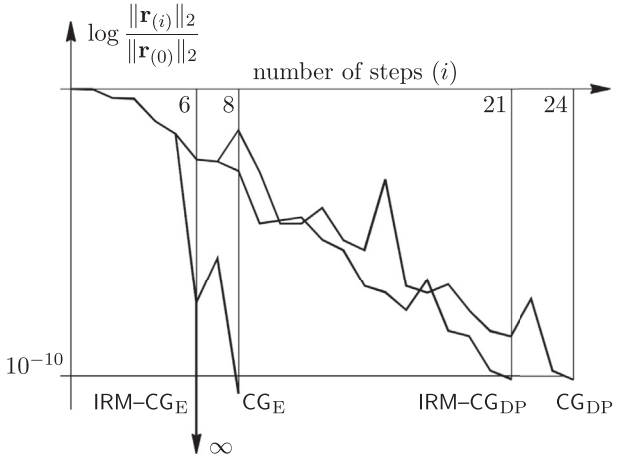

Figure 2. Previous example solved similarly, but with the perturbation of $p_{7,(1)}$.

used to control the condition number $\kappa(\mathbf{A})$, calculated as the ratio of the extreme eigenvalues.

Regardless of $\kappa(\mathbf{A})$, exact relative residual norms are the same for both methods, and the processes stop after $m = 188$ steps. This is because four inactive eigenvalues (eigenvectors orthogonal to the loading) are detected. The last residual norm is integer zero, $\| \mathbf{r}_{(188)} \|_2 = 0$, which means that exact solution of a numerical model is obtained, $\mathbf{x}_{(188)} = \mathbf{x}$. Just for curiosity's sake, vertical deflection of the loaded point (last, 192th component of vector $\mathbf{x}$ is:

$$x_{192,(188)} = -\frac{64464038682410325066453295792043745775932 5413}{28874430348756646673740974021717946019786 0000} \quad (8)$$

In the numerical environment, methods respond similarly to a well-conditioned model (Figure 3a), but for the ill-conditioned case the IRM–CG is more stable, especially if higher accuracy ($\varepsilon = 10^{-10}$) is needed (Figure 3b). Of course, approximate results (for example $\mathbf{x}_{(405)}$ and $\mathbf{x}_{(420)}$, or $\mathbf{x}_{(659)}$ and $\mathbf{x}_{(847)}$) are mutually close and match the exact solution $\mathbf{x}_{(188)}$ reasonably well.

This strategy can also be applied to more complex models (not only) from structural engineering practice, in a combination of various finite elements. If a stiffness matrix is not exactly defined, it is always possible to be rationalised. Elements that are very close to zero could be replaced by the exact integer zero. Using this strategy, the initial roundoff error remains, but it is not accumulated during the solution process (providing exact arithmetic is used). The result is very close to the exact solution of a numerical model $\mathbf{x}_{(m)} \approx \mathbf{x}$. In other words, using rationalisation a slightly different model is obtained, but it can be solved exactly. Again, detailed algorithm performance (various steps and final results) can be compared with that obtained by the floating point arithmetic, executed with various numbers of significant digits.

For example, the model from Figure 4 comprises beam and thin shell elements with displacements and rotations as unknowns. The system has pinned supports and is loaded by two unit moments applied at the centres of the plates. Here, the condition number is controlled by changing the

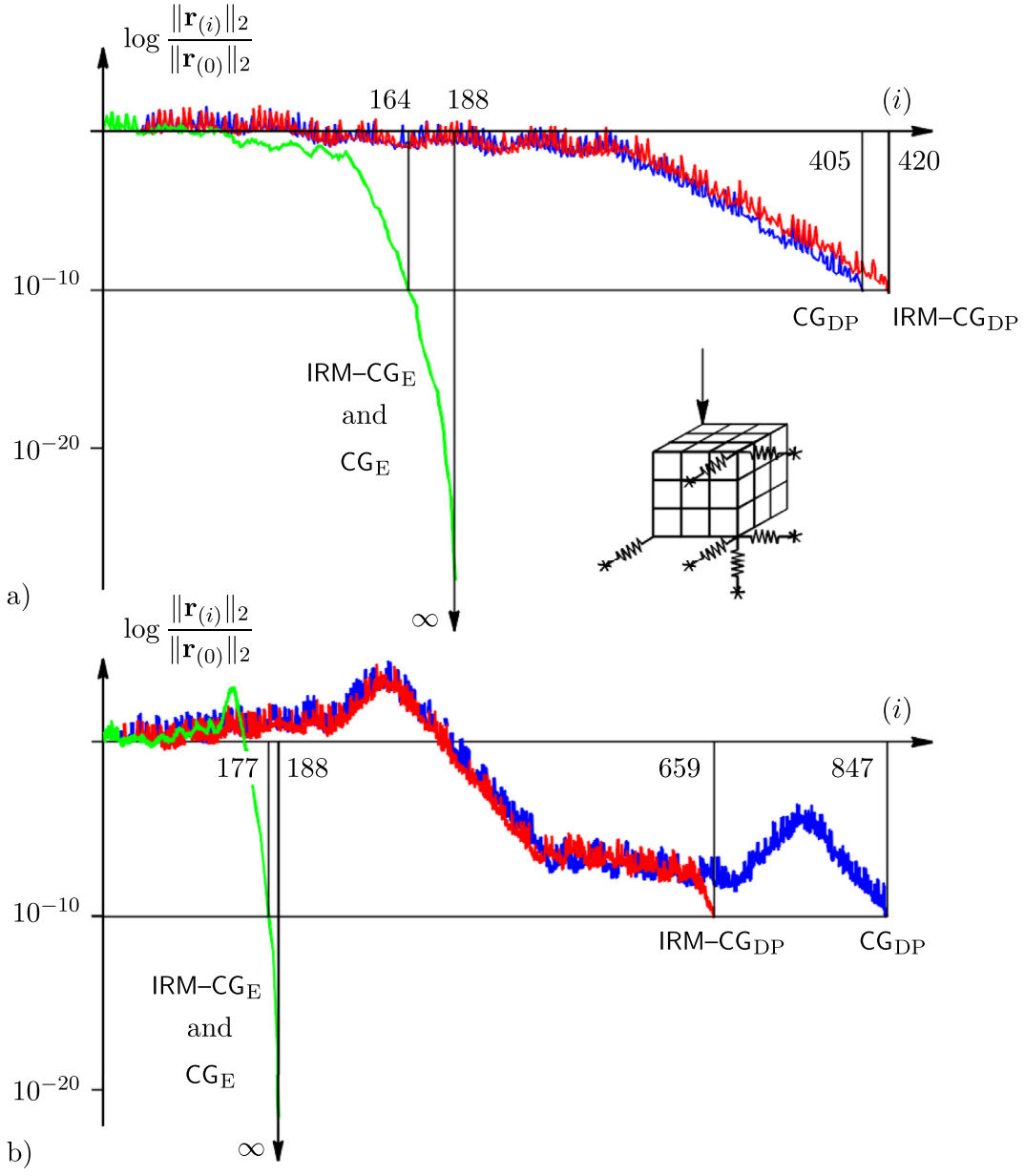

Figure 3. Relative residual norms of the exact and double-precision implementation of the CG and the IRM–CG applied to the finite element analysis of cube model: a) $\kappa(\mathbf{A}) = 7.7 \cdot 10^5$, b).$\kappa(\mathbf{A}) = 6.4 \cdot 10^{12}$.

stiffnesses of two corner columns denoted by A. The system has 183 unknowns and all eigenvectors are active (all $a_j \neq 0$, thus $m = n$). Therefore, using exact arithmetic, the solution is found after 183 steps (Figures 4a and 4b). As in the previous example, the IRM–CG is better for the non-well-posed problem, if higher accuracy hasto be achieved.

Interestingly, there is a large difference in the number of steps between the exact and double-precision approach, even for a well-conditioned systems. Curves are mutually close only at the early stage of calculation (at the very beginning they practically collide), because rounding error is not accumulated enough.

Obviously, an increase of the accuracy to more than 16 decimal digits of mantissa may be justified. With more significant figures, residual curves that correspond to the numerical implementation of methods are closer to each other, and converge to the curve of the exact approach. In the limiting case (theoretically, for an infinite number of digits), all three curves must collide. For reasons of clarity, residual functions obtained by the higher precision arithmetic are not added to the figures.

It should be emphasized that, albeit more realistic, small systems are analysed herein. If the results from Figures 3 or 4 were to be valid for a large system, the methods would be inefficient. In practice, it is just the opposite – the number of steps is much smaller than the number of unknowns. An explanation is given in Figure 5, where a typical decrease of the residual norm for a large number of DoFs (active eigenvalues m) is sketched.

If we look at the energy (hyper)ellipsoids, the exact CG path to the minimum (solution point) usually consists of smaller number of steps than the path that is obtained by the numerical approaches. However, in very rare situations, rounding errors may be beneficial and direct the path more towards the subspace spanned by only several ellipsoid axes, or push it closer to the just one axis, or much better, near the minimum point, making the number of steps even smaller. In such exceptional cases, the residual curve corresponding to the floating-point arithmetic lies below the curve of the exact approach. Furthermore, in that scenario, round-off errors are not highly accumulated, and they

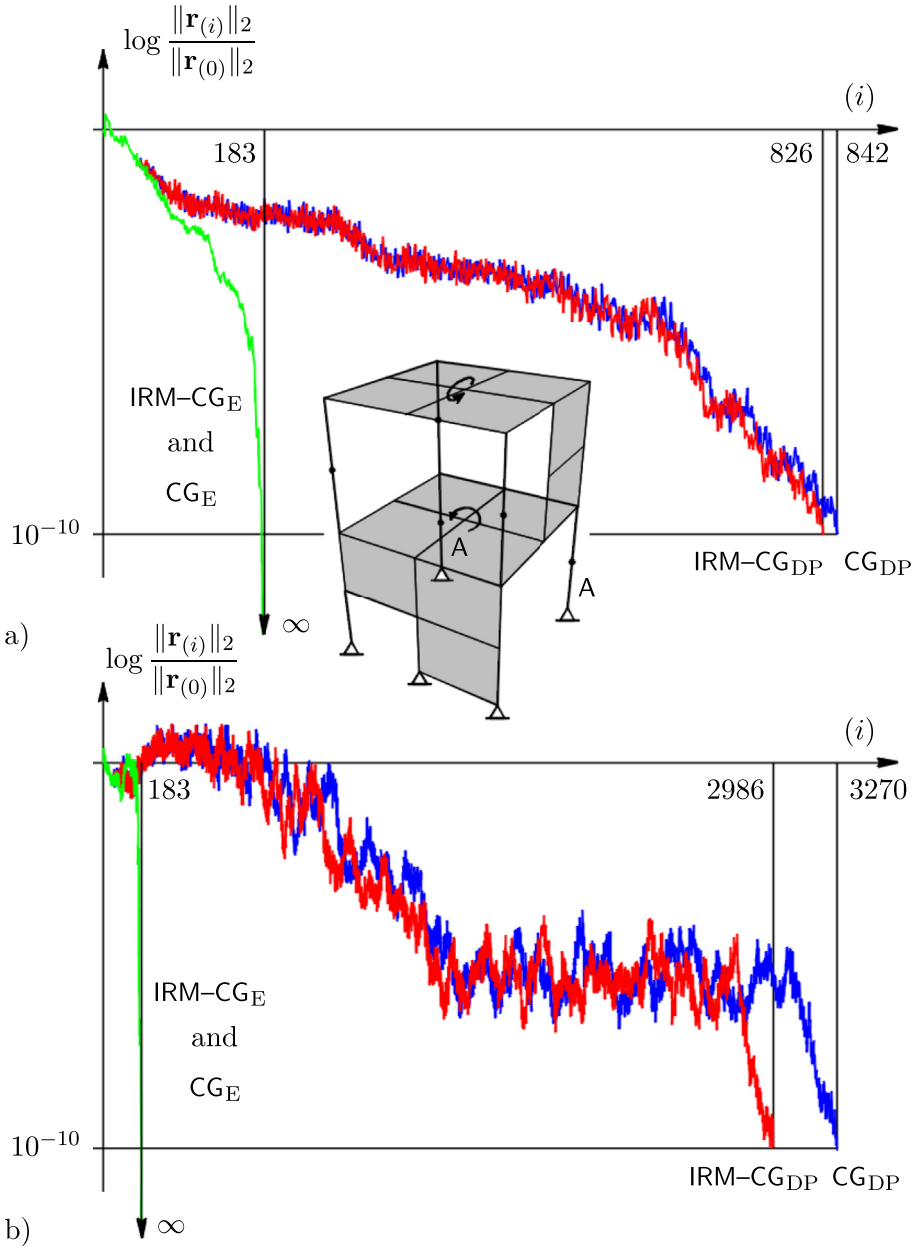

Figure 4. Relative residual norms of the exact and double-precision implementation of the CG and the IRM – CG applied to the finite element analysis of two-story structure: a) $\kappa(\mathbf{A}) = 1.1 \cdot 10^5$, b).$\kappa(\mathbf{A}) = 7.9 \cdot 10^8$.

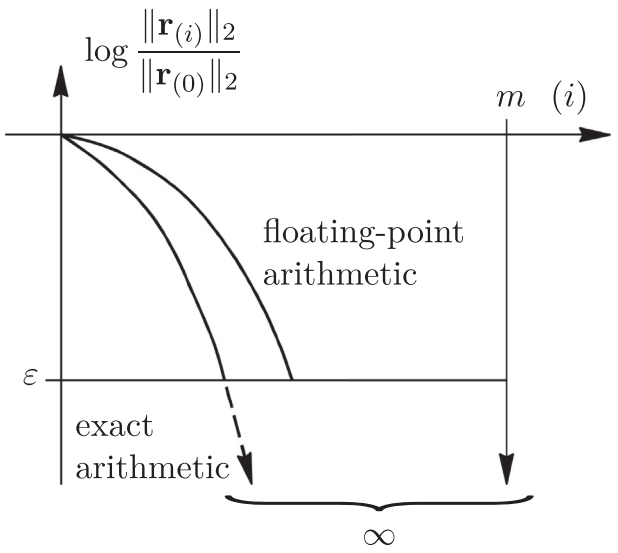

Figure 5. Typical decrease of the relative residual norms for large system.

rather very often cancel each other out during the solution process [24].

Consider a diagonal system with $a_{jj} = j - 1/2$ and $b_j = 1$ ($j = 1, \ldots, 10$), which we intend to solve exactly. If the initial perturbation $\mathbf{s}$ of the IRM–CG is imposed such that $q \leftarrow \mathbf{s}\mathbf{r}_{(0)}/(\mathbf{s}^{\mathrm{T}}\mathbf{A}\mathbf{s})$ and $p_{(0)} \leftarrow q\mathbf{s}$ (lines 3 and 4 of the Algorithm 2 are modified for that purpose); after some trial and error we found that $\mathbf{s} = [\,201 \quad 60 \quad 29 \quad 22 \quad 17 \quad 15 \quad 14 \quad 11 \quad 10 \quad 9\,]^{\mathrm{T}}$ directs the solution path more towards the minimum and for $i < 10$ the iterative process behaves better than if such perturbation is missing (Figure 6). Of course, at $i = 10$ the residual curve of the unperturbed process finishes – drops to infinity, because $m = 10$ and $\mathbf{r}_{(10)} = \mathbf{0}$ (this is the exact zero vector). For $10 < i \leq 30$ the perturbed process continues until $\varepsilon = 10^{-10}$ is satisfied.

It should be stressed that if large systems are considered, a large number of operations would be executed and the rounding error may seriously deteriorate the solution, especially if the condition number is large. In the nonlinear environment, a system should be solved multiple times and the number of operations additionally rises. It is not possible to estimate the influence of the rounding error just by checking the results. Furthermore, double precision is not the complete answer to such troubles, as has been frequently said.

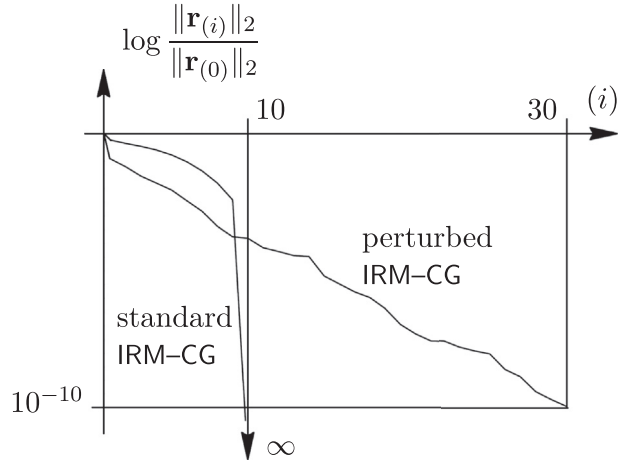

Figure 6. Exact relative residual norms of standard and perturbed IRM–CG.

7. Possibilities for large systems

If a large system of equations needs to be solved, more steps, computer memory, and time are required to obtain (not only) an exact solution. Symbolic expressions become very complicated, while whole numbers and fractions grow enormously. Such problems can be overcome by three strategies.

First, instead of solving exactly, arbitrary-precision arithmetic (with more significant figures than fixed-precision arithmetic) may be applied. In that case, not exact, but very accurate solutions may be found.

Second, an inverse method can be used to establish a benchmark. Simply put, a system matrix is multiplied by some solution to find the corresponding right-hand side vector. The input data and matrix-vector multiplication need to be exact. Such an example is then solved by some numerical (here iterative) method, and the results should be easily compared.

Third, systems with a diagonal matrix may provide an interesting approach. This strategy is based on the consequences of the spectral theorem. Briefly, every SPD matrix can be diagonalised as $\mathbf{V}^{\mathrm{T}}\mathbf{A}\mathbf{V}$, with diagonal elements as eigenvalues and $\mathbf{V} = [\,\mathbf{v}_1 \quad \mathbf{v}_2 \quad \ldots \quad \mathbf{v}_n\,]$. The spectra of the original and diagonal matrix are the same. The corresponding right-hand side vector is $\mathbf{V}^{\mathrm{T}}\mathbf{b}$. This transformation is actually a rotation such that the eigenvectors become parallel to the space coordinate axes. If the original system is solved using the IRM–CG or the CG and steps are then transformed (rotated to that specific position), the results are equal as if the methods had been directly applied to a diagonal system. Therefore, work with a diagonal or original matrix is equivalent (except for the rounding errors if floating-point arithmetic is used).

It should be mentioned that this transformation is based on the eigensolution, which is more 'expensive' than the solution of the corresponding system. Therefore, such a strategy (mostly) makes no sense for real-life engineering problems; it could only be used for clever benchmarks and for tests used to analyse various numerical methods.

Furthermore, for such experiments (with either exact or floating-point arithmetic) the above mentioned transformations of the original matrix are not needed. Put simply, an appropriate diagonal matrix is formed directly. Here 'appropriate' means that the matrix spectrum, condition number, or eigenvalue distribution are easily controlled and that their effects on solver performance could be clearly recognised. Moreover, even for large systems, memory and time requirements are not very demanding and change of the residual spectrum [2, 3] over steps is easy to follow. Of course, purpose of this approach is not to efficiently solve a system, but to get insight into the behaviour of iterative methods. Solution to a diagonal system is easily obtained directly, because equations are mutually independent.

8. Conclusion

With the rapid development of computer algebra systems, exact arithmetic has become a very useful tool for the convergence assessment of numerical solution methods. Herein, a simple equivalent to the CG named the IRM-CG was tested. The method is not based on conjugacy; therefore, it has several advantages over the classical CG.

Its good behaviour is confirmed by analysing three structural engineering examples, which are exactly and numerically solved using both methods. Algorithms are realized using Wolfram Mathematica. Because the methods are equivalent, the exact residual curves always collide and for relatively small condition number they are pretty close. However, for large (and still acceptable) condition number, the results differ in favour of the IRM-CG. Therefore, the method is considered to be very stable, and it should be useful for not well-behaved problems, especially if stronger convergence criterion is adopted. Additionally, even if the CG is preferred as the solution method, it can be restarted using the IRM-CG, which may be called "refresh" instead of "restart".

Two more things should be mentioned. First, the stiffness matrix and the load vector from the finite element analysis model may always

be rationalised. In such cases, a slightly different model is obtained, but it can be exactly solved. This strategy greatly increases the number of problems that can be attacked by this approach. Second, if a large system of equations is considered, three ideas may be exploited: arbitrary precision arithmetic, the inverse solution method and the controlled diagonal system. All can help to overcome the computing demands of the exact arithmetic if larger systems are analysed.

Finally, we start considering a parallel implementation and nonlinear approach to algorithms 1 and 2. Such strategies are extremely useful for large linear and nonlinear models because number of operations and solution time increase rapidly with the number of unknowns. Additionally, the property of conjugacy, which underlies many iterative methods and is valid only for linear models, is not absolutely necessary here. Therefore, we expect that the IRM and the IRM-CG can be very efficient in the nonlinear environment (including optimisation) where conjugacy is not even defined.

Using this methodology, together with the increased computing power, it is possible to establish exact or very precise and more realistic benchmarks for algorithm performance testing.

Declarations

Author contribution statementa

J. Dvornik, A. J. Lazarevic: Analyzed and interpreted the data; Wrote the paper.

D. Lazarevic: Conceived and designed the experiments; Performed the experiments.

M. Uros: Contributed reagents, materials, analysis tools or data.

Funding statement

This work was fully supported by the Croatian Science Foundation under the project IP–2014–09–2899.

Competing interest statement

The authors declare no conflict of interest.

Additional information

No additional information is available for this paper.

References

[1] J. Dvornik, Generalization of the CG method applied to linear and nonlinear problems, Comput. Struct. 10 (1/2) (1979) 217–223.
[2] J. Dvornik, D. Lazarevic, Iterated Ritz Method for solving systems of linear algebraic equations, Gradevinar 69 (7) (2017) 521–535.
[3] J. Dvornik, D. Lazarević, The iterated Ritz method: basis, implementation and further development, Coupled Syst. Mech. 6 (7) (2018) 755–774.
[4] J. Dvornik, D. Lazarević, A. Jaguljnjak Lazarević, M. Demšić, Non-recursive equivalent of the conjugate gradient method without the need to restart, Adv. Civ. Eng. 2019 (2019) 1–5.
[5] O.C. Zienkiewicz, R.L. Taylor, J.Z. Zhu, The Finite Element Method: its Basis and Fundamentals, Elsevier Butterworth-Heinemann, Oxford, 2006.
[6] O.C. Zienkiewicz, R.L. Taylor, The Finite Element Method for Solid and Structural Mechanics, Elsevier Butterworth-Heinemann, Oxford, 2006.
[7] E. Oñate, *Structural Analysis with the Finite Element Method. Linear Statics. Volume 1. Basis and Solids*, International Centre for Numerical Methods in Engineering (CIMNE), Springer, Barcelona, 2009.
[8] E. Oñate, Structural Analysis with the Finite Element Method. Linear statics. Volume 2. Beams, plates and shells, International Centre for Numerical Methods in Engineering (CIMNE), Springer, Barcelona, 2013.
[9] Y.H. Dai, L.Z. Liao, D. Li, On restart procedures for the conjugate gradient method, Numer. Algorithms 35 (2/4) (2004) 249–260.
[10] H.A. Van der Vorst, K. Dekker, Conjugate gradient type methods and preconditioning, J. Comput. Appl. Math. 24 (1/2) (1988) 73–87.
[11] M. Benzi, C.D. Meyer, M. Tůma, A sparse approximate inverse preconditioner for the conjugate gradient method, SIAM J. Sci. Comput. 17 (5) (1996) 1135–1149.
[12] F.H. Pereira, S.L. Sérgio Lopes Verardi, S.I. Nabeta, A fast algebraic multigrid preconditioned conjugate gradient solver, Appl. Math. Comput. 179 (1) (2006) 344–351.
[13] E. vańt Wout, M.B. van Gijzen, A. Ditzel, A. van der Ploeg, C. Vuik, The Deflated Relaxed Incomplete Cholesky CG method for use in a real-time ship simulator, Procedia Comput. Sci. 1 (1) (2010) 249–257.
[14] I.A.R. Moghrabi, A new preconditioned conjugate gradient method for optimization, IAENG Int. J. Appl. Math. 49 (1) (2019) 29–36.
[15] A. Li, Improving AOR iterative methods for irreducible L-matrices, Eng. Lett. 19 (1) (2011) 46–49.
[16] H.A. van der Vorst, Iterative Krylov Methods for Large Linear Systems, Cambridge University Press, Cambridge, 2009.
[17] M.R. Hestens, E. Stiefel, Methods of conjugate gradients for solving linear systems, J. Res. Natl. Bur. Stand. 49 (6) (1952) 409–436.
[18] Y. Shang, Subspace confinement for switched linear systems, Forum Math. 29 (3) (2017) 693–699.
[19] M.L. Overton, Numerical Computing with IEEE Floating Point Arithmetic, Society for Industrial and Applied Mathematics, Philadelphia, 2001.
[20] F.S. Acton, Real Computing Made Real. Preventing Errors in Scientific and Engineering Calculations, Dover Publications, Inc., New York, 2005.
[21] S. Wolfram, An Elementary Introduction to the Wolfram Language, Wolfram Media, Inc., Champaign, 2015.
[22] Wolfram Research, Inc., Mathematica, Version 11.3, 2018. Champaign.
[23] A. Jaguljnjak Lazarević, J. Dvornik, L. Frgić, Utjecaj pogreške zaokruživanja na točnost proračuna konstrukcije, Gradevinar 63 (11) (2011) 911–921.
[24] N.J. Higham, Accuracy and Stability of Numerical Algorithms, Society for Industrial and Applied Mathematics, Philadelphia, 2002.