Using iterative linear solvers in non-linear continuation algorithm

Reijo KOUHIA
Laboratory of Structural Mechanics
Helsinki University of Technology
P.O. Box 2100, 02015 HUT, Finland

ABSTRACT

Some techniques to solve non-linear algebraic equations in finite element analysis are considered. Especially the use of preconditioned iterative linear equation solvers in path-following algorithms is described and the choice of an accelerator iteration and the preconditioner are discussed. Some preliminary results are presented from structural analyses.

1 INTRODUCTION

Solutions of non-linear structural problems under quasi-static loading conditions are usually obtained by using an arc-length type continuation procedure. In these algorithms an additional constraint equation is augmented to the equilibrium equation system. This constraint destroys the symmetry and the banded form of the tangent stiffness matrix. In order to utilize the specific storage format of the tangent matrix, the solution of the constrained system is usually obtained by using the block factorization scheme, where the tangent matrix is factorized and the system is solved with two right-hand side vectors.

In large problems the decomposition time and the storage requirements will be prohibitively high when Gaussian elimination type factorizations are used. Special sparse matrix techniques have been developed which try to minimize the fill in during the decomposition. However, these techniques need reordering of the unknowns and thus are not well parallelizable and vectorizable. Iterative methods seems to be ideal for modern vector and parallel computers to solve systems of linear equations. For large problems they require much less storage than the direct solvers and computing times are also in many cases reduced.

There are at least two main factors which have contributed to the slow spread of iterative linear solvers in non-linear structural problems. Firstly, the computational cost of the incremental and iterative methods is usually so high that the size of the discretized non-linear problems has to be much smaller than it is possible to do in linear analysis. Therefore storage requirements are not so critical issue. In a practical non-linear analysis the decomposition time is usually comparable to the time needeed to form the global matrices and internal force vectors. Secondly, the stiffness matrix might be ill conditioned and not necessarily positive definite in certain parts of the solution path. In addition, some important control parameters of the continuation process, like the determinant or the lowest eigenvalue of the tangent stiffness matrix, are not easily accessible when iterative solvers are used.

The block factorization strategy is not feasible if the solution of the linear system is obtained with an iterative solver. Usually the solution is carried out directly to the augmented unsymmetric system. Krenk and Hededal [1] have recently introduced an orthogonal or a dual orthogonal residual method where this block factorization type of solution is not needed. In this paper these procedures are coped with iterative linear equation solvers and the performance is compared to that of usual practice. Some example problems both in non-linear heat conduction and solid mechanics are solved.

2 CONTINUATION ALGORITHM

Discretization of the quasi-static equilibrium equations expressing the balance between external and internal forces results in an equation of the form:

$$\boldsymbol{f}(\boldsymbol{q}, \lambda) \equiv \lambda \boldsymbol{p}_r - \boldsymbol{r} = \boldsymbol{0}, \tag{1}$$

where $\boldsymbol{p}$ is the external load vector. If the finite element method is used in the discretization process, the internal force vector $\boldsymbol{r}$ follows from the assembly operation of the element contributions

$$\boldsymbol{r}^{(e)} = \int_{V^{(e)}} \boldsymbol{B}^T \boldsymbol{s} dV.$$

The vector $\boldsymbol{s}$ contains the stress components. The strain-displacement matrix $\boldsymbol{B}$ is defined by

$$\delta \boldsymbol{e} = \boldsymbol{B} \delta \boldsymbol{q},$$

where the column vector $\boldsymbol{e}$ contains the strain components.

Usually the applied loading is assumed to depend linearily on a single parameter, i.e. the load parameter λ, such that $\boldsymbol{p} = \lambda \boldsymbol{p}_r$, where $\boldsymbol{p}_r$ is the reference load vector.

Solution of the equation system (1) forms a one-dimensional equilibrium curve in an $N + 1$ dimensional displacement-load parameter space, where N is the dimension of the state space, i.e. the number of dof's in the vector $\boldsymbol{q}$. Procedures to trace the one dimensional equilibrium path defined by equation (1) are called continuation or path following methods. They are incremental or step-wise algorithms. A typical continuation step includes the predictor and the corrector phases.

To traverse a solution path a proper parametrization is needed. Simple load control is the oldest type of parametrization. It is usually the most efficient one in regular parts of a path, and the adaptation of an iterative linear equation solver in it is straightforward. However, near the so called limit points, where the structure loses its load carrying capacity (at least locally), it might break down. At the limit point the tangent stiffness matrix is singular and the load parameter is decreasing after such a point. A remedy is to change the control from the load parameter to some of the displacement components. Selecting the controlling displacement (or component from the scaled vector containing both displacements and the load parameter) to be the largest one from the last converged increment, results in a simple and reliable continuation procedure [2]. Non-dimensionalizing of the variables is an essential point of this method. Nevertheless, it is recommendable for all other procedures, too.

A usual setting of a continuation process is to augment the discrete equilibrium equations with a single constraint equation c in the following form:

$$\boldsymbol{g}(\boldsymbol{q},\lambda) = \left\{ \begin{array}{rcl} \boldsymbol{f}(\boldsymbol{q},\lambda) & = & \boldsymbol{0} \\ c(\boldsymbol{q},\lambda) & = & 0 \end{array} \right. . \tag{2}$$

This kind of procedures are also commonly called arc-length methods. A large class of constraint equations can be written in the form

$$c(\boldsymbol{q},\lambda) = \boldsymbol{t}^T \boldsymbol{C} \boldsymbol{n} - c_0 = 0$$

where $\boldsymbol{t}$ and $\boldsymbol{n}$ are $N+1$ dimensional vectors and c_0 is a scalar. For explicit expressons of the vectors $\boldsymbol{t}$ and $\boldsymbol{n}$ see ref. [3]. The weighting matrix $\boldsymbol{C}$ can be partitioned as $\mathrm{diag}(\boldsymbol{W},\alpha^2)$, where $\boldsymbol{W}$ is a positive definite or semidefinite diagonal matrix corresponding to displacements and α is a scaling factor.

Using the Newton-Raphson linearization on the extended equation system (2) results in

$$\left\{ \begin{array}{rcrcl} \dfrac{\partial \boldsymbol{f}}{\partial \boldsymbol{q}}\delta \boldsymbol{q} + \dfrac{\partial \boldsymbol{f}}{\partial \lambda}\delta\lambda + \boldsymbol{f}(\boldsymbol{q},\lambda) & = & -\boldsymbol{K}\delta\boldsymbol{q} + \boldsymbol{p}_r\delta\lambda + \boldsymbol{f} & = & \boldsymbol{0} \\ \dfrac{\partial c}{\partial \boldsymbol{q}}\delta \boldsymbol{q} + \dfrac{\partial c}{\partial \lambda}\delta\lambda + c(\boldsymbol{q},\lambda) & = & \boldsymbol{c}^T\delta\boldsymbol{q} + e\delta\lambda + c & = & 0 \end{array} \right. . \tag{3}$$

Usually in structural analyses the tangent stiffness matrix $\boldsymbol{K}$ is symmetric. Therefore, in order to utilize the specific sparsity pattern and symmetry of the tangent stiffness matrix, the solution of the augmented equations (3) is usually performed by using the following three phase block elimination method, also known as bordering algorithm [2], [4], [5]:

1. solve $\boldsymbol{K}\delta\boldsymbol{q}_f = \boldsymbol{f}$ and $\boldsymbol{K}\boldsymbol{q}_p = \boldsymbol{p}_r$,
2. compute $\delta\lambda = -(c + \boldsymbol{c}^T\delta\boldsymbol{q}_f)/(e + \boldsymbol{c}^T\boldsymbol{q}_p)$,
3. update $\delta\boldsymbol{q} = \delta\boldsymbol{q}_f + \delta\lambda\boldsymbol{q}_p$.

In this format the solution of the linear equation system at phase 1 is performed by means of direct solvers. If iterative solvers are used, the nonsymmetric sparse format of the equation system (3):

$$\boldsymbol{H}\delta\boldsymbol{y} = \boldsymbol{h}, \quad \boldsymbol{H} = \begin{bmatrix} \boldsymbol{K} & -\boldsymbol{p}_r \\ \boldsymbol{c}^T & e \end{bmatrix}, \quad \delta\boldsymbol{y} = \left\{ \begin{array}{c} \delta\boldsymbol{q} \\ \delta\lambda \end{array} \right\}, \quad \boldsymbol{h} = \left\{ \begin{array}{c} \boldsymbol{f} \\ -c \end{array} \right\}, \tag{4}$$

seems to be more appropriate, see refs. [6], [7]. Chan and Saad [8] have studied different preconditioning techniques in the non-linear elliptic second order problem

$\Delta u + \lambda \exp(u) = 0$ on unit square with zero Dirichlet boundary conditions, discretized by a standard five point finite difference formula. One of their conclusions is that, when a preconditioning is available, it seems best to work directly with an iterative method on the unsymmetric form (4).

Usually the question of the "best choice" of the constraint equation is overemphasized in the engineering literature, for example see discussion in ref. [9]. However, the specific form of the constraint equation is a relevant topic in the present context. A procedure which fits well together with the iterative linear equation solvers, is the orthogonal residual procedure by Krenk and Hededal [1], which does not require the block factorization process and thus only one solution of linear equation is needed per iteration.

As argued by Krenk and Hededal, the magnitude of the displacement increment is optimal when an orthogonality condition

$$\Delta \boldsymbol{q}^T \boldsymbol{f} = 0$$

is satisfied. This linear condition is used to determine the current load parameter λ. The algoritm can be described briefly as:

1. compute: $\boldsymbol{r}_i = \boldsymbol{r}(\boldsymbol{q}_0 + \Delta \boldsymbol{q}_i)$, $\Delta \boldsymbol{r}_i = \boldsymbol{r}_i - \lambda_0 \boldsymbol{p}_r$, $\Delta\lambda_{i+1} = \Delta \boldsymbol{q}_i^T \Delta \boldsymbol{r}_i / \Delta \boldsymbol{q}_i^T \boldsymbol{p}_r$,
2. solve: $\boldsymbol{K} \delta \boldsymbol{q}_{i+1} = \boldsymbol{f}_{i+1} = (\lambda_0 + \Delta\lambda_{i+1}) \boldsymbol{p}_r - \boldsymbol{r}_i$,
3. update: $\Delta \boldsymbol{q}_{i+1} = \Delta \boldsymbol{q}_i + \delta \boldsymbol{q}_{i+1}$.

λ_0 and $\boldsymbol{q}_0$ denote the load level and the displacement vector at the beginning of current increment. However, even if the algorithm seems to be ideally suited for the use of an iterative linear equation solver, it has some drawbacks observed in numerical experiments. Since the size of the increment is not restricted during the iteration, the algorithm seems to have some tendency of increasing the size of the displacement increment near limit points.

3 PRECONDITIONED ITERATIVE METHODS

3.1 Krylov subspace methods

In the sequel, a generic linear equation system will be denoted by

$$\boldsymbol{A}\boldsymbol{x} = \boldsymbol{b},$$

where the coefficient matrix $\boldsymbol{A}$ can be symmetric or unsymmetric. An equivalent preconditioned system is

$$\boldsymbol{M}_1^{-1} \boldsymbol{A} \boldsymbol{M}_2^{-1} \boldsymbol{y} = \boldsymbol{M}_1^{-1} \boldsymbol{b},$$

where $\boldsymbol{M} = \boldsymbol{M}_1 \boldsymbol{M}_2$ is the preconditioning matrix and $\boldsymbol{M}_1, \boldsymbol{M}_2$ are the left- and right preconditioning matrices, respectively. In practice this split form is not always needed. It is usually possible to rewrite the iterative method in a way that only a computational step: solve $\boldsymbol{u}$ from $\boldsymbol{Mu} = \boldsymbol{v}$, is necessary, so the preconditioner applies in its entirety. The question of preconditioning will be discussed briefly in the next section.

Krylov subspace methods seem to be among the most important iterative techniques available for solving large linear systems [10], [11], [12]. These techniques are based on projections onto Krylov subspaces, which are subspaces spanned by vectors which are obtained recursively by multiplying the previous residual with the matrix: i.e.

$$\mathcal{K}_m(\boldsymbol{A}, \boldsymbol{r}_0) = \operatorname{span}\left\{\boldsymbol{r}_0, \boldsymbol{A}\boldsymbol{r}_0, \boldsymbol{A}^2\boldsymbol{r}_0, ..., \boldsymbol{A}^{m-1}\boldsymbol{r}_0\right\},$$

where $\boldsymbol{r}_0 = \boldsymbol{b} - \boldsymbol{A}\boldsymbol{x}_0$. Approximate solution of the system is found from a m-dimensional subspace $\boldsymbol{x}_0 + \mathcal{K}_m$ by imposing the Petrov-Galerkin condition requiring the residual to be orthogonal to another m-dimensional subspace $\mathcal{L}_m$.

The most wellknown Krylov subspace method is the preconditioned conjugate gradient (PCG) method for symmetric positive definite (SPD) matrices. There are many different implementations of the PCG-iteration, but the following algorithm is perhaps the most common: construct $\boldsymbol{M}$, initialize $\boldsymbol{r}_0 = \boldsymbol{b} - \boldsymbol{A}\boldsymbol{x}_0$, solve $\boldsymbol{M}\boldsymbol{d}_0 = \boldsymbol{r}_0$, compute $\tau_0 = \boldsymbol{r}_0^T\boldsymbol{d}_0$ and iterate $i = 0, 1, 2, ...$ until convergence:

1. compute: $\boldsymbol{s} = \boldsymbol{A}\boldsymbol{d}_i, \quad \alpha_i = \tau_i / \boldsymbol{d}_i^T\boldsymbol{s}$,
2. update: $\boldsymbol{x}_{i+1} = \boldsymbol{x}_i + \alpha_i\boldsymbol{d}_i, \quad \boldsymbol{r}_{i+1} = \boldsymbol{r}_i - \alpha_i\boldsymbol{s}$,
3. solve: $\boldsymbol{Mz} = \boldsymbol{r}_{i+1}$ and compute $\tau_{i+1} = \boldsymbol{r}_{i+1}^T\boldsymbol{z}, \quad \beta_i = \tau_{i+1}/\tau_i$,
4. update $\boldsymbol{d}_{i+1} = \boldsymbol{z} + \beta_i\boldsymbol{d}_i$.

It is a Galerkin (orthogonal projection) type Krylov subspace method, i.e. $\mathcal{L}_m = \mathcal{K}_m$. One iterate of the PCG method requires one matrix-vector product, five[1] level-1-operations and one solution of linear equations $\boldsymbol{Mz} = \boldsymbol{r}$.

If the matrix $\boldsymbol{A}$ is symmetric but indefinite the PCG-algorithm can become unstable and even break down. Paige and Saunders [13] were the first to devise stable algorithms for symmetric indefinite systems. These two algorithms called SymmLQ and MinRes are based on Lanczos tridiagonalization, which exists also in indefinite case.

For unsymmetric matrices the situation is much more complex. The CG method for symmetric and positive definite systems has two important properties. It is based on three term recurrence, and it minimizes the error with respect to the energy norm. Unfortunately these two properties can only be fulfilled for nonsymmetric CG-type schemes for a very limited class of matrices, namely the shifted and rotated Hermitean matrices. In this paper only those algorithms are considered which retain the short recurrencies thus being more favourable with respect to memory requirements. Biconjugate gradient (BCG) type algorithms are based on the Lanzcos biorthogonalization algorithm which builds a pair of biorthogonal bases for the two subspaces $\mathcal{K}_m(\boldsymbol{A}, \boldsymbol{r}_0)$ and $\mathcal{K}_m(\boldsymbol{A}^T, \tilde{\boldsymbol{r}}_0)$. In the numerical examples the following BCG-type methods are used in this study: biconjugate gradient squared (CGS), biconjugate gradient stabilized (Bi-CGStab), quasi-minimal residual (QMR) and transpose free QMR (TFQMR). For a unified general description of all these methods with numerous references see ref. [14].

[1]PCG requires an additional norm evaluation if the convergence is checked from the residual $\boldsymbol{r}$.

3.2 Preconditioning

It is well known that the performance of iterative solvers depends on the eigenvalue distribution and on the possible non-normality of the coefficient matrix. These problems can be avoided, at some extent, by employing a preconditioner. It seems to be generally agreed that the choice of the preconditioner is even more critical than the choice of the type of the Krylov subspace iteration [15].

There are two major conflicting requirements in the development of a preconditioned iteration, namely the construction[2] and use of a preconditioner should be cheap and its resemblance with matrix $\boldsymbol{A}$ should be as close as possible. The most general preconditioning strategies can be grouped into classes:

1. preconditioners based on classical iterations like Jacobi, SSOR,
2. incomplete sparse LU-decompositions (ILU or IC for symmetric matrices),
3. polynomial preconditioners,
4. explicit sparse approximate inverse preconditioners,
5. multigrid or multilevel preconditioners.

Incomplete factorization is perhaps the most wellknown strategy. There are many variants of ILU-decompositions differing, for instance on the way how the nonzero pattern of the preconditioner is defined. The simplest strategy is to have the same nonzero pattern for the $\boldsymbol{L}$ and $\boldsymbol{U}$ factors as $\boldsymbol{A}$. This incomplete factorization known as ILU(0) is easy and inexpensive to compute, but often leads to a crude approximation resulting in many iterations in the accelerator to converge. Several alternative ILU factorizations have been developed in which the fill-in is determined by either using the concept of level of fill or by a treshold strategy where the nonzero pattern of the preconditioner is determined dynamically neglecting small elements in the factorization.

Meijerik and Van der Vorst [16] proved existence of the ILU factorization for arbitrary fill patterns if the coefficient matrix is a M-matrix[3]. This is often the case, e.g. matrices arising from discretizations of the heat equation. However, matrices arising from problems in structural mechanics usually do not have this property. In order to circumvent this problem an additional reduction step has been introduced, where an M-matrix is determined from the stiffness matrix and the incomplete factorization scheme is applied to this [17].

Mathematical analysis reveals that for second-order elliptic boundary value problems the ILU(0) approach is asymptotically no better than the unpreconditioned iteration. More precisely, the condition number of the ILU preconditioned operator is of the same order as matrix $\boldsymbol{A}$. Several variants of the basic ILU have been presented in the literature e.g. MILU, RILU and DRILU (modified, relaxed and dynamically relaxed) [17]. However, in real engineering problems these modifield versions does not perform

[2]If the preconditioner is to be used many times more effort can be paid to its construction.

[3]A matrix is a M-matrix if its off-diagonal elements are nonpositive and all the elements of the inverse are positive.

any better than the basic ILU. Ajiz and Jennings [18] proposed the corrected IC factorization (CIC), [4] which guarantees a positive definite preconditioner if the matrix itself is SPD.

It should be remembered that the effectiveness of a preconditioning strategy is highly problem and architecture dependent. For instance, incomplete factorizations are difficult to implement on high-performance computers, due to the sequential nature of the triangular solves. On the other hand, sparse approximate inverse preconditioning needs only matrix-vector products, which are relatively easy to vectorize and parallelize, but they are usually not as robust as ILU-factorization based strategies [15].

For second-order elliptic PDE's discretized by low order finite elements many of the listed preconditioning techniques can be used. However, for finite element models of thin-shells only the corrected incomplete factorization allowing some degree of fill-in [18], [19] or a multilevel preconditioner [20] seems to be the only reasonable choices.

For a certain type of a preconditioning technique, the computational complexity can be reduced. Construction of a preconditioning matrix $\boldsymbol{M}$ in a form

$$\boldsymbol{M} = (\tilde{\boldsymbol{D}} + \boldsymbol{E})\hat{\boldsymbol{D}}(\tilde{\boldsymbol{D}} + \boldsymbol{F}), \tag{5}$$

where $\tilde{\boldsymbol{D}}$, $\hat{\boldsymbol{D}}$ are diagonal matrices and $\boldsymbol{E}$ and $\boldsymbol{F}$ are the strictly lower and upper parts of $\boldsymbol{A} = \mathrm{diag}(\boldsymbol{A}) + \boldsymbol{E} + \boldsymbol{F}$, allows implementation of the preconditioned CG or Bi-CG-type methods in which the computational labor is comparable to the unpreconditioned case. This strategy is due to Eisenstat [21], and it is commonly called as the Eisenstat trick, see also refs. [10],[12]. Unfortunately the usefulness of this stratgy is somewhat limited. For a very sparse matrices, such as resulting from a low order FE discretizations of the diffusion equation, the triangular solution including short rows is the main bottleneck in a typical supercomputer implementation. Also quality of the split-preconditioners (5), which can be used in the Eisenstat trick is not good enough in shell problems.

4 EXAMPLES

In this section some example problems are solved and the performances of the iterative methods are compared to a direct solution procedure. The direct solver used is a slightly modified version from ref. [22], pages 327-329. All computations have been preformed on Digital Alpha Server 8400 [5] using double precision representation for real numbers. The program is written in Fortran 77 and the level 3 optimization flag is used in the compilation for most, including all linear algebra routines.

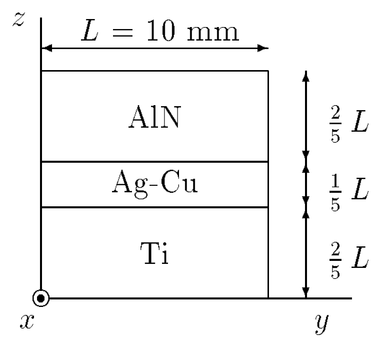
Convergence of the iteration is checked by the relative and absolute residual error and declared if

$$\|\boldsymbol{r}_i\|_2/\|\boldsymbol{b}\|_2 < RTOL \quad \text{or} \quad \|\boldsymbol{r}_i\| < ATOL,$$

except computations where the Eisenstat trick is used. In that case the absolute criteria is used and the measure is the weighted Euclidean norm with the preconditioner as the weight matrix.

[4]The name corrected incomplete Cholesky is adopted from ref. [19].

[5]Center for Scientific Computing, Espoo, Finland.



	Ti	Ag-Cu	AlN
E/GPa	118.6	94.4	315
ν	0.36	0.36	0.25
$10^6\alpha$	6.441	17.995	2.85

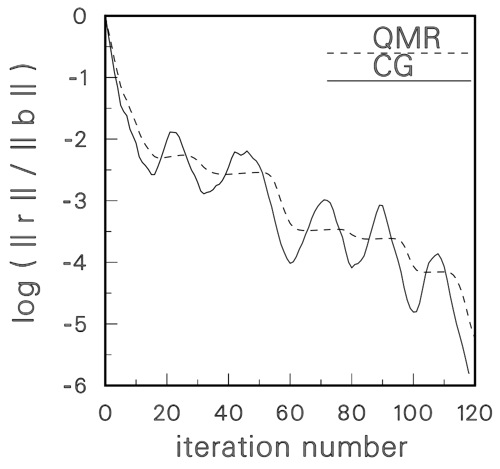


Figure 1: Three material solid block: (a) geometry and material data, (b) convergence behaviour of CG and symmetric QMR iteration with IC(0) preconditioner $L/h = 20$ (h is the sidelength of an element).

4.1 Three material elastic block

An elastic block composed by three material layers and occupying the region (in cartesian coordinates) $0 < (x, y, z) < L$ is considered. The material interfaces are horizontal layers parallel to the xy-plane, and having positions $z = \frac{2}{5}L$ and $z = \frac{3}{5}L$. The stack models a ceramic (AlN) to metal (Ti) joint brazed together with Ag-Cu filler alloy. Constitutive parameters used for these materials are shown in fig. 1.

Uniform meshes with eight node trilinear brick elements are used. The only loading is the temperature change defined by $\Delta T = \Delta T_0 xyz/L^3$. Minimal constraints which prevent the rigid body motion are imposed. Convergence tolerancies used are $ATOL = RTOL = 10^{-6}$.

Some data of the stiffness matrix is shown in table 1 and a comparison of the performance of the preconditioned CG iteration with a direct in-core skyline solver is recorded in table 2. The SSOR preconditioning is implemented by using the Eisenstat trick. In this case the convergence is measured in the weighted norm $\|\boldsymbol{x}\| = (\boldsymbol{x}^T \boldsymbol{M} \boldsymbol{x})^{1/2}$ and thus the tabulated values are not comparable to those of IC(0) preconditioning. Compressed row storage format is used to store the nonzero elements of the matrix (also for the IC(0) preconditioner).

Convergence behaviour for the conjugate gradient method exhibits some oscillations which are absent if the material characteristics are uniform. The symmetric QMR iteration [23] with coupled two term recurrence shows much smoother convergence than the CG method, see fig. 1.

As expected, the solution times for the direct solvers become quickly untolerably high due to the large bandwidth which grows like $B \sim N^{2/3}$. For large 3-D problems iterative solvers are the only possible way to get the solution at reasonable cost.

Table 1: Three material block, stiffness matrix characteristics.

L/h	N	B_{rms}	M	NZ	$2NZ/M$
5	642	117	70552	18615	0.528
10	3987	380	1455352	135915	0.187
20	27777	1355	36805402	1035165	0.056
40	206757	5105	1043093302	8075265	0.015

N number of unknowns
B_{rms} root-mean square bandwidth
M number of elements under envelope
NZ number of nonzero elements

Table 2: Three material block, solution times

	direct solver		CG-IC(0)				CG-SSOR Eisenstat[†]			
L/h	F time	B time	iter	P time	I time	ratio	ω	iter	I time	ratio
5	0.11	0.01	25	0.02	0.07	1	0.8	22	0.04	3
10	8.3	0.2	57	0.2	1.3	6	1.2	41	0.51	17
20	1660.	6.	120	1.6	39.8	40	1.4	77	15.0	111
40	97 hours*	140.*	257	12.7	697.	490*	1.7	195	324.	1080*

F time = Factorization time in seconds
B time = Backsubstitution + load vector reduction time
P time = Preconditioner construction time
I time = Iteration time
ratio = (F+B)/(P+I)
* = estimated value, † = convergence measured in weighted norm

4.2 Pinched cylinder

A well known shell element test is the pinched cylinder, see e.g. ref. [24]. Length of the shell equals to its diameter ($L = 2R$) and the Poisson's ratio is 0.3. Performance of the conjugate gradient method is studied with respect to the relative thickness and some relevant parameters in the finite element model. As expexted, the problem gets harder when the thickness to radius ratio, i.e. the characteristic thickness gets smaller. Here results of only the cases $t/R = 10^{-2}$ and 10^{-3} are reported.

The shell elements are facet type 3-node triangular or 4-node quadrilateral elements, with drilling rotations using the Hughes-Brezzi formulation [25]. The plate bending part of the element is based on the stabilized MITC theory [26]. In the MITC formulation the stabilization parameter has been 0.4 for both triangular and quadrilateral elements. One octant of the shell is discretized by uniform 30×30 mesh resulting in 5489 unknowns. Strict convergence tolerance is used $RTOL = 10^{-9}$ and only the relative criteria is active.

Value of the regularizing penalty parameter γ used in the formulation of Hughes and Brezzi has a notable effect on the convergence of the conjugate gradient iteration.

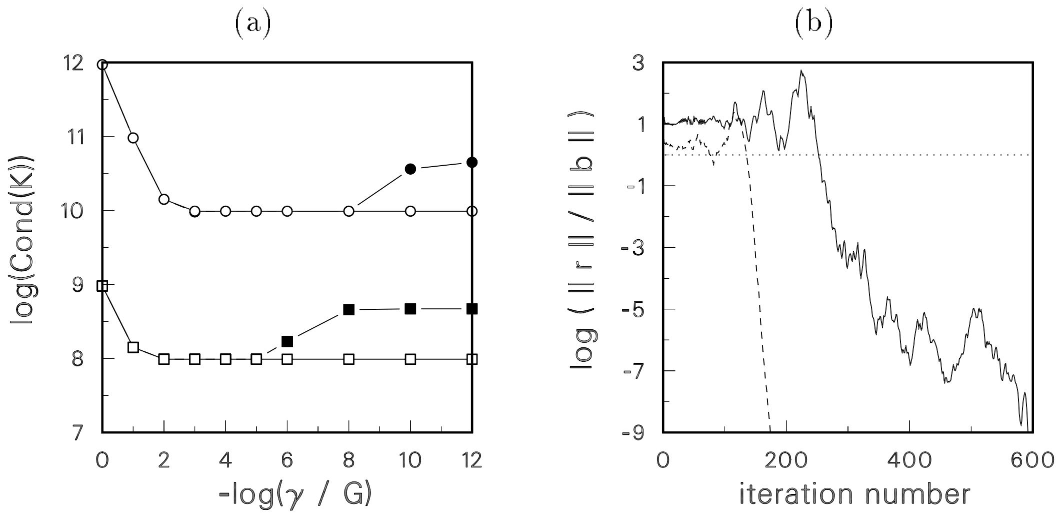


Figure 2: (a) Effect of the regularizing parameter γ on the spectral condition number of the stiffness matrix. Solid markers correspond to stiffness matrices without Allman displacement field. Upper two curves $t/R = 10^{-3}$ and lower ones $t/R = 10^{-2}$. (b) Convergence plot of the CG-IC(0) iteration: $\gamma = G$ (without Allman field), solid line $t/R = 10^{-3}$ and dashed line $t/R = 10^{-2}$. Quadrilateral 30×30 mesh.

It affects the spectral condition number of the stiffness matrix, see fig. 2a. Due to the ill-conditioning of the thinner shell problem the convergence of the PCG iteration slows down at the level of 10^{-4} in the relative error, see fig. 2b. Adding Allman type displacement field [27], [28] to the in-plane interpolation has also an effect on the convergence, however, there seems to be no definite trend on that dependency. The number of iterations needed to convergence are shown in table 3.

The IC factorization does not exist for all values of the γ parameter. A simple remedy is to use the shifting strategy of Manteuffel [29] where the matrix $\boldsymbol{A} + \rho\mathrm{diag}(\boldsymbol{A})$ is factorized instead of $\boldsymbol{A}$. However, the quality of such a preconditioner is not very good as can be seen from table 4.

4.3 Non-linear analysis of cylindrical panel

A shallow cylindrical shell subjected to a central point load on the convex side is a common test problem of path-following algorithms, see e.g. ref. [30]. The longitudinal boundaries are immovable, whereas the curved edges are completely free. The problem data are: radius $R = 2540$ mm, length of the straight hinged edge $L = 508$ mm, Young's modulus $E = 3.10275$ GPa, Poisson's ratio $\nu = 0.3$ and $\theta = 0.1$ rad. Two values for the thickness are used: $t = 12.7$ mm ($R/t = 200$) and $t = 6.35$ mm ($R/t = 400$). Uniform 32×32-mesh with DKT-elements is used in the simulations resulting in 6175 dof for a quadrant of the panel. Drilling rotations are included by the Hughes-Brezzi formulation and the γ-parameter has the value $\gamma = 0.026G$. Allman-type interpolation is also included.

First, some comparisons with the IC(0) and the CIC(ψ) preconditioners for the

Table 3: Influence of the regularizing parameter γ on the convergence of the CG-IC(0) iteration, 30×30 mesh with MITC elements. A = Allman type amendment for the in-plane interpolation.

(a) $t/R = 10^{-2}$

γ/G	Q-A	T-A	Q	T
1.0	-	-	175	293
0.5	-	-	153	283
0.25	-	306	142	273
0.2	379	301	138	269
0.1	147	288	129	259
0.01	120	262	117	2337
0.001	118	249	115	1201

(b) $t/R = 10^{-3}$

γ/G	Q-A	T-A	Q	T
1.0	-	-	593	-
0.5	-	-	511	-
0.25	-	-	502	-
0.2	-	-	508	-
0.1	356	-	397	1197
0.01	307	768	330	743
0.001	287	674	284	636

Table 4: Influence of the shift ρ on the convergence, 30×30 mesh with Allman type interpolation, $\gamma = G$, quadrilateral MITC elements with $t/R = 10^{-2}$.

shift	0.035	0.040	0.045	0.050	0.055	0.060	0.065	0.075	0.100
iterations	2017	1265	947	825	797	805	825	875	1000

linear problem are performed. Also a pure treshold version of the IC preconditioner is tested, where the diagonal corrections are omitted. This preconditioner is abbreviated as IC(ψ). Behaviour of the CIC(ψ) and IC(ψ) methods are tested with different drop tolerancies ψ and the preconditioner sizes are also recorded in table 5. Convergence tolerancies have been $RTOL = ATOL = 10^{-5}$. It should be noted that the IC(0) factorization needs a small shift ($\rho = 0.005$) as well as the IC(ψ) method, where the optimal shift depend on the drop tolerance ψ. This is an annoying feature, since there is no known method to determine the optimal or near optimal shift a priori.

The computing times for almost all cases shown in table 5 are higher than the solution time needed for the direct solution, worst case almost by factor 10. However, there are some potential of using CIC or IC with a low drop-tolerance in the non-linear analysis, if the preconditioner need not to be computed at every time when the stiffness matrix is formed.

Five continuation strategies are compared. Symmetric formulations use the orthogonal residual method with direct or iterative linear equation solver. For consistently linearized elliptical constraint, the symmetric formulation uses only direct solver and with the augmented (4) nonsymmetric forms both direct and iterative solvers are used. Only the full Newton-Raphson strategy is used in the computations, even it is not necessary for all parts of the continuation paths. Load-deflection curves are shown in fig. 3 and the iteration characteristics from the computations of the thicker shell are recorded in table 6. Same conclusions can also be drawn from the thinner case.

The results in the linear case for the Jacobi and IC(0) strategies for preconditioning can be directly generalized to the non-linear analysis. Only the strategy, where the

Table 5: Comparison of the shifted IC(0), Jacobi and CIC(ψ) and IC(ψ) preconditioners on the pinched cylindrical panel. P = preconditioner evaluation time, I = iteration time, t-r = (P+I)-times/direct solution time, m-r = memory ratio: $NZ(\boldsymbol{M})/NZ(\boldsymbol{A})$. Solution time of the direct solver is 2.6 s.

$R/t = 200$

method	shift	iter	P	I	t-r	$NZ(\boldsymbol{M})$	m-r
Jacobi	-	1935	0.0	18.7	7.2	6175	0.05
IC(0)	$5 \cdot 10^{-3}$	184	0.1	4.3	1.7	127095	1.00
CIC(1.0)	-	1995	0.3	22.7	8.8	6175	0.05
CIC(10^{-1})	-	1079	0.4	14.2	5.6	18036	0.15
CIC(10^{-2})	-	299	0.7	6.2	2.7	93349	0.74
CIC(10^{-3})	-	118	1.3	3.9	2.0	200689	1.6
CIC(10^{-4})	-	51	2.7	3.2	2.3	391606	3.1
CIC(10^{-5})	-	23	5.1	3.0	3.1	713691	5.6
CIC(10^{-6})	-	9	7.0	1.8	3.4	1010602	8.0
IC(10^{-1})	$2 \cdot 10^{-2}$	388	0.4	5.6	2.3	40010	0.31
IC(10^{-2})	$5 \cdot 10^{-3}$	78	0.7	1.7	0.9	102486	0.81
IC(10^{-3})	10^{-3}	39	1.4	1.4	1.1	226936	1.8
IC(10^{-4})	10^{-4}	19	2.9	1.5	1.7	455261	3.6

$R/t = 400$

method	shift	iter	P	I	t-r	$NZ(\boldsymbol{M})$	m-r
Jacobi	-	1924	0.0	18.7	7.2	6175	0.05
IC(0)	$5 \cdot 10^{-3}$	181	0.1	4.3	1.7	127095	1.00
CIC(1.0)	-	1957	0.3	23.0	9.0	6175	0.05
CIC(10^{-1})	-	1023	0.4	13.7	5.4	18496	0.15
CIC(10^{-2})	-	304	0.7	6.1	2.6	88230	0.69
CIC(10^{-3})	-	110	1.3	3.7	1.5	195314	1.5
CIC(10^{-4})	-	46	2.7	3.1	2.2	386823	3.0
CIC(10^{-5})	-	21	5.1	2.7	3.0	718339	5.7
CIC(10^{-6})	-	8	7.3	1.7	3.5	1030444	8.1
IC(10^{-1})	$2 \cdot 10^{-2}$	386	0.5	6.3	2.6	38905	0.31
IC(10^{-2})	$5 \cdot 10^{-3}$	76	0.7	1.6	0.9	98030	0.77
IC(10^{-3})	10^{-3}	38	1.4	1.3	1.0	215934	1.7
IC(10^{-4})	10^{-4}	18	2.8	1.3	1.6	434653	3.4

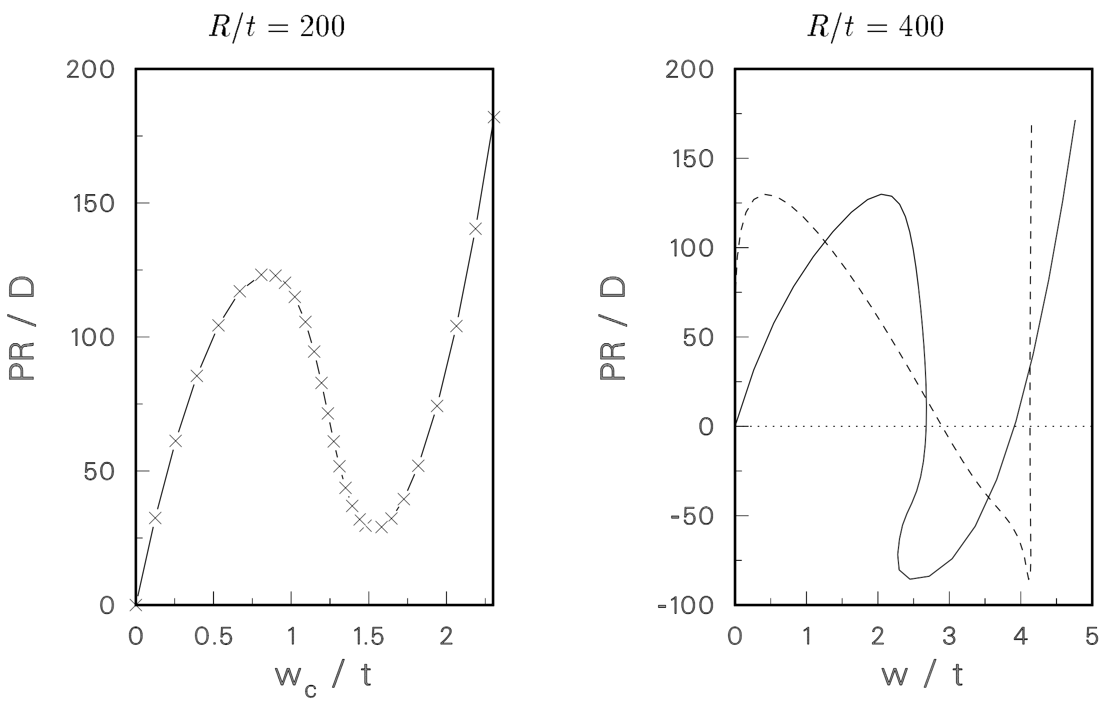


Figure 3: Hinged cylindrical panel, load deflection paths of the load point (solid line) and the free edge (dashed line).

Table 6: Iteration characteristics from the non-linear analysis of the pinched cylinder ($R/t = 200$); OR = Orthogonal Residual, E = Elliptic constraint.

constraint	solver	steps	smu	pu	CG-it	P/F	I/B	T
OR	direct	27	111	-	-	290	15	530
OR	CG-IC(0)	27	111	1	23462	0.1	497	711
OR	CG-IC(0)	27	111	27	20384	3	423	641
OR	CG-CIC(10^{-6})	27	111	1	4725	6	827	1047
OR	CG-CIC(10^{-6})	27	111	27	1037	184	182	580
OR	CG-IC(10^{-3})	27	111	1	9151	1.3	350	567
OR	CG-IC(10^{-3})	27	111	27	4316	35	167	417
OR	CG-IC(10^{-3})	27	111	111	4312	153	168	534
E block f.	direct	46	254	-	-	630	69	1227
E augm.	direct	46	254	-	-	1476	41	2098
E augm.	Bi-CGStab-ILU(0)	46	254	46	73573	8.6	6287	6842

smu = stiffness matrix updates, pu = precondtitoner updates
CG-it = total number of CG-iterations
P/F = total (incomplete or full) factorization time
I/B = total CG-iteration or backsubstitution time
T = total run time

treshold IC preconditioner (with drop tolerance $\psi = 10^{-3}$ and shift $\rho = 10^{-3}$) is updated at the beginning of each increment, gives faster solution time than using the direct solver.

Using the augmented nonsymmetric equation system requires much more computing time than the block factorization with direct linear solver. All the tested nonsymmetric iterations CGS, BI-CGSTAB, QMR and TFQMR performed almost identically. Since only the no-fill ILU preconditioner is used in the nonsymmetric case, the figures in table 6 should not be compared to the symmetric iterations with treshold IC preconditioning.

It should be mentioned that a mixed strategy, where the stiffness matrix is updated when necessary, would be in favour of direct solvers.

5 CONCLUDING REMARKS

At present iterative methods for linear systems of equations have reached the level of robustness that they are included in almost every valued commercial FE-code. In despite of the excellent preformance of preconditioned Krylov subspace methods in heat transfer and stress analysis of solid bodies, they cannot be regarded as robust as direct solvers. Especially, for non-linear shell analysis the results with incomplete factorization preconditioners are still far from being satisfactory. Possibly the multilevel approach will change the affirmed state of affairs. Finally, it should be emphasized that the performance of the various procedures is highly problem and computer architecture dependent, for which reason fair comparison of different approaches is difficult.

REFERENCES

[1] S. Krenk and O. Hededal. A dual orthogonality procedure for non-linear finite element equations. *Computer Methods in Applied Mechanics and Engineering*, 123:95–107, 1995.

[2] W.C. Rheinboldt. *Numerical Analysis of Parametrized Nonlinear Equations.* Wiley, 1986.

[3] R. Kouhia and M. Mikkola. Strategies for structural stability analyses. In N.E. Wiberg, editor, *Advances in Finite Element Technology*, pages 254–278, 1995.

[4] H.B. Keller. The bordering algorithm and path following near singular points of higher nullity. *SIAM Journal on Scientific and Statistical Computing*, 4:573–582, 1983.

[5] K.H. Schweizerhof and P. Wriggers. Consistent linearization for path following methods in non-linear FE analysis. *Computer Methods in Applied Mechanics and Engineering*, 59:261–279, 1986.

[6] E.L. Allgower and K. Georg. *Numerical Continuation Methods - An Introduction.* Springer-Verlag, 1990.

[7] E. Barragy and C.F. Carey. A partitioning scheme and iterative solution for sparse bordered systems. *Computer Methods in Applied Mechanics and Engineering*, 70:321–327, 1988.

[8] T.F. Chan and Y. Saad. Iterative methods for solving bordered systems with applications to continuation methods. *SIAM Journal on Scientific and Statistical Computing*, 6(2):438–451, 1985.

[9] E. Riks. On formulatins of path-following techniques for structural stability analysis. paper presented at the European Conf. on New Advances in Computational Structural Mechanics, Giens, France, 1991.

[10] H. Voss. Iterative methods for linear systems of equations. University of Jyväskylä, Department of Mathematics, lecture notes 27, 1993.

[11] O. Axelsson. *Iterative Solution Methods.* Cambridge University Press, 1994.

[12] Y. Saad. *Iterative Methods for Sparse Linear Systems.* PWS Publishing, 1996.

[13] C.C. Paige and M.A. Saunders. Solution of sparse indefinite systems of linear equations. *SIAM Journal on Numerical Analysis*, 12:617–629, 1975.

[14] R. Barrett, M. Berry, T. Chan, J. Demmel, J. Donato, J. Dongarra, V. Eijkhout, R. Pozo, C. Romine, and H. van der Vorst. *Templates for the Solution of Linear Systems: Building Blocks for iterative methods.* SIAM, 1994.

[15] M. Benzi and M. Tůma. Numerical experiments with two approximate inverse preconditioners, 1997. CERFACS TR/PA/97/11.

[16] J.A. Meijerink and H.A. van der Vorst. An iterative solution method for linear systems of which the coefficient matrix is a symmetric M-matrix. *Mathematics of Computation*, 31:148–162, 1977.

[17] P. Saint-Georges, G. Warzee, R. Beauwens, and Y. Notay. High-performance PCG solvers for FEM structural analysis. *International Journal for Numerical Methods in Engineering*, 39:1313–1340, 1996.

[18] M.A. Ajiz and A. Jennings. A robust incomplete Cholesky conjugate gradient algorithm. *International Journal for Numerical Methods in Engineering*, 20:949–966, 1984.

[19] P. Saint-Georges, G. Warzee, Y. Notay, and R. Beauwens. Fast iterative solvers for finite element analysis in general and shell analysis in particular. In B.H.V. Topping, editor, *Advances in Finite Element Technology*, pages 273–282, Edinburgh, 1996. Civil-Comp Press.

[20] V.E. Bulgakov. The use of the multi-level iterative aggregation method in 3-D finite element analysis of solid, truss, frame and shell structures. *Computers and Structures*, 63(5):927–938, 1997.

[21] S.C. Eisenstat. Efficient implementation of a class of preconditioned conjugate gradient methods. *SIAM Journal on Scientific and Statistical Computing*, 2:1–4, 1981.

[22] G. Dhatt and G. Touzot. *Une Présentation de la Méthode des Éléments Finis.* Maloine S.A. Éditeur, 1984. 2^{e} édition.

[23] R.W. Freund and N.M. Nachtigal. An implementation of the QMR method based on coupled two-term recurrences. *SIAM Journal on Scientific Computing*, 15(2):313–337, 1994.

[24] H. Hakula, Y.Leino, and J. Pitkäranta. Scale resolution, locking and high-order finite element modelling of shells. *Computer Methods in Applied Mechanics and Engineering*, 133:157–182, 1996.

[25] T.J.R. Hughes and F. Brezzi. On drilling degrees of freedom. *Computer Methods in Applied Mechanics and Engineering*, 72:105–121, 1989.

[26] M. Lyly, R. Stenberg, and T. Vihinen. A stable bilinear element for the Reissner-Mindlin plate model. *Computer Methods in Applied Mechanics and Engineering*, 110:343–357, 1993.

[27] D.J. Allman. A compatible triangular element including vertex rotations for plane elasticity analysis. *Computers and Structures*, 19:1–8, 1984.

[28] A. Ibrahimbegovic, R.L. Taylor, and E.L. Wilson. A robust quadrilateral membrane finite element with drilling degrees of freedom. *International Journal for Numerical Methods in Engineering*, 30:445–457, 1990.

[29] T.A. Manteuffel. An incomple factorization technique for positive definite linear systems. *Mathematics of Computation*, 34:473–497, 1980.

[30] G. Horrigmoe and P.G. Bergan. Nonlinear analysis of free-form shells by flat finite elements. *Computer Methods in Applied Mechanics and Engineering*, 16:11–35, 1978.