A solver combining reduced basis and convergence acceleration with applications to non-linear elasticity

Jean-Marc Cadou, Michel Potier-Ferry

A solver combining reduced basis and convergence acceleration with applications to non-linear elasticity

J. M. Cadou[1] and M. Potier-Ferry[2]

[1]*Laboratoire Ingénierie des Matériaux de Bretagne, Université Européenne de Bretagne, Université de Bretagne Sud, Rue de Saint Maudé, B.P. 92116, 56321 Lorient Cedex, France*
[2]*Laboratoire de Physique et Mécanique des Matériaux, Université Paul Verlaine, Ile du Saulcy, 57045 Metz, France*

An iterative solver is proposed to solve the family of linear equations arising from the numerical computation of non-linear problems. This solver relies on two quantities coming from previous steps of the computations: the preconditioning matrix is a matrix that has been factorized at an earlier step and previously computed vectors yield a reduced basis. The principle is to define an increment in two sub-steps. In the first sub-step, only the projection of the unknown on a reduced subspace is incremented and the projection of the equation on the reduced subspace is satisfied exactly. In the second sub-step, the full equation is solved approximately with the help of the preconditioner. Last, the convergence of the sequences is accelerated by a well-known method, the modified minimal polynomial extrapolation. This algorithm assessed by classical benchmarks coming from shell buckling analysis. Finally, its insertion in path following techniques is discussed. This leads to non-linear solvers with few matrix factorizations and few iterations.

1. INTRODUCTION

The algorithms to solve non-linear problems transform a non-linear system into to a family of linear ones. In general, these linear systems cannot be solved simultaneously because the right-hand side vectors depend on the solutions at previous steps. At least two classes of non-linear algorithms can be distinguished. In the first class, the matrix and the right-hand side are updated at each iteration, as for instance in the Newton–Raphson method. In the second group that includes the modified Newton method (MNM) and the asymptotic numerical method (ANM, [1]), the same matrix holds good for several linear equations. Iterative solvers are convenient for the first class, but direct solvers are more efficient when a matrix is associated with multiple right-hand sides, because the important computation time due to matrix factorization is shared between several linear systems.

Although an iterative method is not a natural choice for linear problems with multiple right-hand sides, there are many papers dealing with this question, see, for example, Reference [2]. When

the vectors are known since the beginning, all the systems can be solved simultaneously with the so-called block solvers BL-*. The block conjugate gradient method can then be applied [3] if the matrix is symmetric, as it is generally the case in structural mechanics. For non-symmetric matrices, a consequent bibliography can be found in [4, 5].

Unfortunately in the case of a non-linear problem, the right-hand side vectors are build up recursively and block solvers are not adapted. Another idea is to account for the results of previous linear systems in the numerical treatment of the current system, for instance by using the previously computed vectors. Such techniques have been introduced under various names: deflated conjugate gradient [6], augmented conjugate gradient [7] or subspace projection extrapolation [8]. Comparisons between these methods have been done within a computational electromagnetic framework [9]. However, these methods have several drawbacks when they are applied with a cheap preconditioning technique such as ILU or Incomplete Cholesky. Indeed a too large number of stored vectors leads to numerical instabilities and many re-used vectors are not necessarily representative of the problem to be solved.

The aim of this paper is to present and discuss two linear iterative solvers that are convenient in the numerical solutions of non-linear problems. These solvers rely on previously computed quantities. Among linear problems, a few are solved by a direct method and most of them are solved iteratively, the last factorized matrix being the preconditioner. The basic idea is more or less the same as in the MNM, but in the present algorithm, a preconditioning matrix is used to solve linear equations and not non-linear ones as in MNM. Further this procedure is coupled with a reduced basis technique, the basis being made of previously computed vectors, as in [6–9].

A non-linear solver has been presented recently [10]. It is based on a homotopy transformation, a perturbation technique and convergence acceleration by Padé approximants (HPP). It improves the high-order Newton algorithm and the high-order iterative algorithms introduced in [11, 12]. Roughly, one applies the consistent tangent matrix on the reduced subspace and the chosen preconditioner in a correction phase on the whole space. The linear version of this HPP algorithm is one of the two linear solvers evaluated in this paper, see Section 2.3. But it is a bit intricate, because one has to define the homotopy transformation and the recurrence formulae to compute the series. That is why we propose a second linear solver that is an iterative variant of the previous one: it includes the same reduced basis, the same type of reduced problem and the same preconditioner. Convergence acceleration is a family of very efficient techniques to improve the performance of iterative algorithms [13, 14]. Two acceleration techniques will be used in this paper: the Padé approximants introduced in [15] to accelerate Taylor series and mainly the modified minimal polynomial extrapolation (MMPE, [16]) that is suitable to improve the convergence of sequences and that has about the same efficiency as Padé approximants [16–18], see Section 2.2. The resulting algorithm is relatively simple and combines, first a preconditioning by a consistent tangent operator emanating from a previous step, second a projection on a reduced basis, last an acceleration by MMPE.

The paper is organized as follows. In part 2, we state or recall the considered algorithms: the new iterative solver, the linear version of the algorithm from [10], the acceleration by MMPE and the Newton–Raphson algorithm with arc-length control. In part 3, two classical benchmarks from non-linear shell analysis permit us to assess the performance of the proposed methods and to compare their efficiency. They will be compared with the preconditioned conjugated gradient associated with various preconditioners. Last, in part 4, the robustness of the proposed procedure is evaluated by considering two path following problems involving the solution of many linear systems.

2. ALGORITHMS

One deals with the numerical computation of non-linear systems of partial differential equations that depend on a scalar parameter λ. After discretization, this system can be written in a generic form:

$$R(U,\lambda)=0, \quad U\in\mathbb{R}^N, \ \ \lambda\in\mathbb{R} \tag{1}$$

Most of the algorithms to compute non-linear equations as (1) lead to the computation of a family of linear systems in the form:

$$K^i U^i = F^i \quad \text{with } U^i \in \mathbb{R}^N, \ F^i \in \mathbb{R}^N, \ K^i \in \mathbb{R}^{N*N} \tag{2}$$

In this paper, two algorithms are discussed to solve such a family of linear systems. These algorithms will be based on a preconditioning matrix that is a full tangent matrix coming from a previous step of the computation. We intend to keep the same preconditioner to solve several linear problems. Since the linear problems are computed by an iterative method, the consistent tangent matrix can be considered without additional cost. That is why we shall choose the Newton–Raphson method to transform the non-near problem (1) into a family of linear problems such as (2): for completeness this non-linear algorithm is recalled in Section 2.4. In the parts 3 and 4, the proposed algorithms will be applied in thin shell analysis that is known to provide ill-conditioned matrices. The chosen class of preconditioning matrices is able to remain very efficient in such a case.

2.1. A linear iterative solver involving a reduced subspace technique

We now describe an algorithm to solve one linear problem. For simplicity, the indices are omitted and (2) is rewritten as:

$$KU = F \quad \text{with } U \in \mathbb{R}^N, \ F \in \mathbb{R}^N, \ K \in \mathbb{R}^{N*N} \tag{3}$$

An increment from a trial solution U should satisfy the following incremental equation:

$$K\Delta U = F - KU \tag{4}$$

The principle of the proposed linear solver is to define an increment in two sub-steps. In the first sub-step, only the projection of the unknown on a reduced subspace is incremented and the projection of the incremental equation (4) on the reduced subspace is satisfied exactly. In the second sub-step, the incremental equation (4) is satisfied approximately by replacing the consistent matrix K by a given preconditioning matrix K^*. Hence the proposed algorithm relies on the choice of a preconditioning matrix and of a reduced subspace. Let $\mathscr{E}$ be a given subspace generated by n vectors E^i, where n is much smaller than N. We suppose that this basis $(E^i)_{i=1,n}$ is orthonormalized according to a chosen scalar product. From this subspace, one can define a prolongation operator in such a way that a vector of the whole space, $X \in \mathbb{R}^N$ is associated with any vector q of the reduced subspace $\mathbb{R}^n$:

$$X = \mathbb{P}q = \sum_{l=1}^{n} q_l E^l \tag{5}$$

If U is the starting vector of the iteration, the first sub-increment $\Delta U = \mathbb{P}\Delta q$ satisfies a linear problem on the subspace $\mathscr{E}$

$$k\Delta q = f - CU \tag{6}$$

where k and C are, respectively, a reduced and a coupling matrix and f is a reduced load. All these quantities are defined with the following expressions:

$$\begin{aligned} k &= {}^t\mathbb{P}K\mathbb{P}, \quad k \in \mathbb{R}^{n*n} \\ C &= {}^t\mathbb{P}K, \quad C \in \mathbb{R}^{n*N} \\ f &= {}^t\mathbb{P}F, \quad f \in \mathbb{R}^n \end{aligned} \tag{7}$$

Finally, by considering the initial solution $U_0 = 0$, we define the iterative method by the following recurrence formulae:

$$\begin{aligned} k\Delta q &= f - CU_{i-1} \\ K^* V &= F - K(\mathbb{P}\Delta q + U_{i-1}) \\ U_i &= U_{i-1} + \mathbb{P}\Delta q + V \end{aligned} \tag{8}$$

2.2. Convergence acceleration

During the iterations, a sequence of vectors denoted by s has been constructed:

$$s = \{U_1, \ldots, U_i, \ldots, U_I\} \tag{9}$$

The previous sequence can slowly converge, or sometimes can be divergent, especially when the preconditioner is not very close to the consistent matrix K. There are several techniques to improve the convergence of sequences: see, the Reference [19] where a detailed review of these acceleration techniques is presented. Here, we only consider a vector extrapolation method and more precisely the MMPE [16]. Within this method, the iterate U_I is replaced by the following 'extrapolated' vector:

$$T_I = U_0 + \sum_{i=1}^{I} a_i (U_i - U_{i-1}) = U_0 + \sum_{i=1}^{I} a_i \Delta U_i \tag{10}$$

where the real coefficients a_i are computed by solving a small linear system resulting from an orthogonality condition

$$[M]\{a\} = \{b\} \tag{11}$$

where the matrix $[M]$ and the right-hand side $\{b\}$ are defined by:

$$M_{ij} = (\Delta U_{j+1} - \Delta U_j) \cdot Y_i \quad \text{and} \quad b_i = -\Delta U_1 \cdot Y_i \quad \text{with } i, j = 1, \ldots, I-1 \tag{12}$$

Within MMPE, the vectors Y_i are arbitrary and linearly independent. In our numerical tests, these vectors are the vectors ΔU_i.

In practice this acceleration procedure works well, but only if the sequence s is not too large and it has been advised to apply MMPE to a limited number of vectors such as $I = 5, 10$ or 15 [18]. So in this case, once the extrapolated vector has been built with the help of expression (10) one defines a new linear problem:

$$K X = F - K T_I \tag{13}$$

Next the new unknown vector X is computed by the same iterative procedure. In the numerical tests, this procedure will be denoted as 'restart algorithm'.

Finally, the first presented iterative method combines the recurrence formulae (8) and possibly the convergence acceleration method (10). In the application, it will be referred as 'proposed method'.

Other acceleration techniques can be deduced from Padé approximants. In the practice, the results from MMPE and Padé approximants are very close [18]. It has been established that, in some specific cases, the Padé approximants introduced in [15] lead to the same mathematical results as MMPE [17].

2.3. A variant based on homotopy and perturbation

In this section, we propose a second linear solver, which is based on homotopy technique, perturbation method and Padé approximants (HPP). This solver is the linear version of an iterative high-order corrector recently introduced in Reference [10]. For the linear problem (3), the method of Reference [10] to define a high-order iterative corrector leads to the following system of equations:

$$\begin{aligned} \Lambda - C_{\text{pen}}({}^t\mathbb{P}U - q) &= 0 \\ kq + CV - \Lambda &= f \\ K(\mathbb{P}q + V) + \mathbb{P}\Lambda &= F \\ U &= \mathbb{P}q + V \end{aligned} \tag{14}$$

where Λ is a Lagrange multiplier, $\Lambda \in R^n$, C_{pen} is a pseudo-penalization matrix. As in Reference [10], this matrix is defined with the following expression:

$$C_{\text{pen}} = \alpha \,\text{Diag}(k) \tag{15}$$

where α is a small parameter and is chosen equal to 10^{-20} [10] in all the numerical tests presented in this paper. The reduced matrix k, the coupling matrix C and the reduced load f are the same as in Section 2.1 and are defined by expressions (7). System (14) is then modified by introducing an artificial parameter ε

$$\begin{aligned} \Lambda - \varepsilon C_{\text{pen}}({}^t\mathbb{P}W - q) &= 0 \\ kq + \varepsilon CV - \Lambda &= f \\ (1-\varepsilon)K^*V + K(\mathbb{P}q + \varepsilon V) + \mathbb{P}\Lambda &= F \\ W &= \mathbb{P}q + V \end{aligned} \tag{16}$$

where ε is a real parameter ($0 \leqslant \varepsilon \leqslant 1$). The unknowns (Λ, q, V, W) are then sought in the form of a truncated integro-power series of the parameter ε. For example, the asymptotic expansion for vector W is:

$$W(\varepsilon) = W_0 + \varepsilon W_1 + \varepsilon^2 W_2 + \cdots + \varepsilon^p W_p \tag{17}$$

This series is introduced in Equation (16) and equating like power of ε a sequence of a linear problem is obtained:

Order 0 *in* ε

$$\begin{aligned} \Lambda_0 &= 0 \\ kq_0 &= f \\ K^*V_0 &= F - K\mathbb{P}q_0 \\ W_0 &= \mathbb{P}q_0 + V_0 \end{aligned} \tag{18}$$

Order p *in* ε

$$\begin{aligned} \Lambda_p &= C_{\text{pen}}({}^t\mathbb{P}W_{p-1} - q_{p-1}) \\ kq_p &= \Lambda_p - CV_{p-1} \\ K^*V_{p-1} &= K^*V_{p-1} - K(\mathbb{P}q_p + V_{p-1}) - \mathbb{P}\Lambda_p \\ W_p &= \mathbb{P}q_p + V_p \end{aligned} \tag{19}$$

Finally, when all the terms $(\Lambda_i, W_i, q_i, V_i)$ are computed, one replaces the asymptotic expansions (17) by equivalent rational fractions, called Padé approximants. The method to deduce the Padé approximants is exactly the same as in many previous works (see, for example, [15]) and this permits to accelerate the convergence of the polynomial approximation (17).

In this form, the algorithm is not iterative because the series (17) and the Padé approximants are explicit functions of ε. The sought solution could be obtained by replacing $\varepsilon = 1$. Nevertheless, if the radius of convergence is larger than 1, the obtained value $U = W(\varepsilon = 1)$ can be not satisfactory. The perturbation process can be applied again and this defines an iterative process, see [11, 12]. In this paper, we shall not need iterations in the application of this algorithm.

One remarks that the recurrence formulae (19) are rather similar to formula (8) and to the iterative process of Section 2.1, except for the Lagrange multiplier and the pseudo-penalization that had been introduced in [10] to stabilize the result of the perturbation process. We no longer discuss this algorithm that is only the linear version of the one presented in [10]. It is recalled here for comparison with the main algorithm of this paper that is the one of Section 2.1. In the following application, it will be referred as 'HPP algorithm'.

There is a wide recent literature about the association of homotopy and perturbation techniques to solve differential equations or partial differential equations, generally to get analytical solutions and with few terms in the series [20–22]. Homotopy perturbation methods can also be applied in a numerical framework and with many terms in the series.

2.4. Applications to non-linear problems

We recall the classical iterative Newton–Raphson method, associated here with an arc-length control. The generic non-linear Equation (1) is solved in several steps, each step including a prediction phase and several corrections. In the prediction phase, the increment $(\Delta U_{\text{pred}}, \Delta\lambda_{\text{pred}})$ is characterized by the following equations:

$$\begin{aligned} \left.\frac{\partial R}{\partial U}\right|_0 \Delta U_{\text{pred}} + \left.\frac{\partial R}{\partial \lambda}\right|_0 \Delta\lambda_{\text{pred}} = 0 \\ \|\Delta U_{\text{pred}}\|^2 + (\Delta\lambda_{\text{pred}})^2 = s^2 \end{aligned} \tag{20}$$

The second one is the arc-length control and s is the given arc-length. In the first one, $\left.\frac{\partial R}{\partial U}\right|_0$ is the Jacobian matrix at the starting point of the step. If (U_i, λ_i) is the result of the iteration i, the correction $(\Delta U_i, \Delta\lambda_i)$ is characterized by:

$$\begin{aligned} \left.\frac{\partial R}{\partial U}\right|_i \Delta U_i + \left.\frac{\partial R}{\partial \lambda}\right|_i \Delta\lambda_i = -R|_i \\ \Delta U_i \cdot \Delta U_{\text{pred}} + \Delta\lambda_i \Delta\lambda_{\text{pred}} = 0 \end{aligned} \tag{21}$$

The second one is the so-called normal plane condition. As it is well known, this non-linear process requires the solution of $N_{\text{iter}} + 1$ linear problems, all the matrices being different.

In this paper, these linear problems will be solved by the iterative methods of Section 2.1 or Section 2.3. The preconditioning matrix will be the Jacobian matrix of a previous step and the same matrix will be used for many linear problems as (20) or (21). Several strategies will be discussed to manage the preconditioning matrix: for instance in the next part, the prediction problem (20) will be solved by a direct method and this factorized matrix will be the preconditioner for all the iterations of the same step. Hence this leads to an algorithm with one matrix factorization per step.

3. NUMERICAL RESULTS

3.1. Two numerical tests

The previous linear solvers are now applied to linear problems coming from a predictor–corrector algorithm, the Newton–Raphson method. Two main numerical tests are considered and described in Figures 1(a) and (b). The non-linear response curves are plotted in Figures 2(a) and (b). The first example is a cylindrical shell with two diametrically opposed rectangular cut-outs [23]. The corresponding geometric and material characteristics are given in Figure 1. Owing to the symmetries, only one octant of the structure is considered and the mesh involves about 1600 classical triangular DKT18 elements [24] (5190 d.o.f.). The second example is an open cylinder pulled out by two diametrically opposite point loads [25]. The geometric and material characteristics are given in Figure 1(b). The finite element for this second example is an eight nodes quadrilateral element that is presented in Reference [26]. An additional variable accounts for the strain variations throughout the thickness (EAS concept). Owing to the symmetries, only one octant of the cylinder is modelled by 900 elements (30×30). The corresponding number of d.o.f. is then 17 000.

The non-linear response curves have been obtained by the Newton–Raphson method. They are pictured in Figures 2(a) and (b). The prescribed step length of this incremental/iterative method is chosen in such a way that at each step, 2–3 corrections are necessary to get the desired accuracy. For these first tests, the algorithm has been designed in such a way that only one matrix is

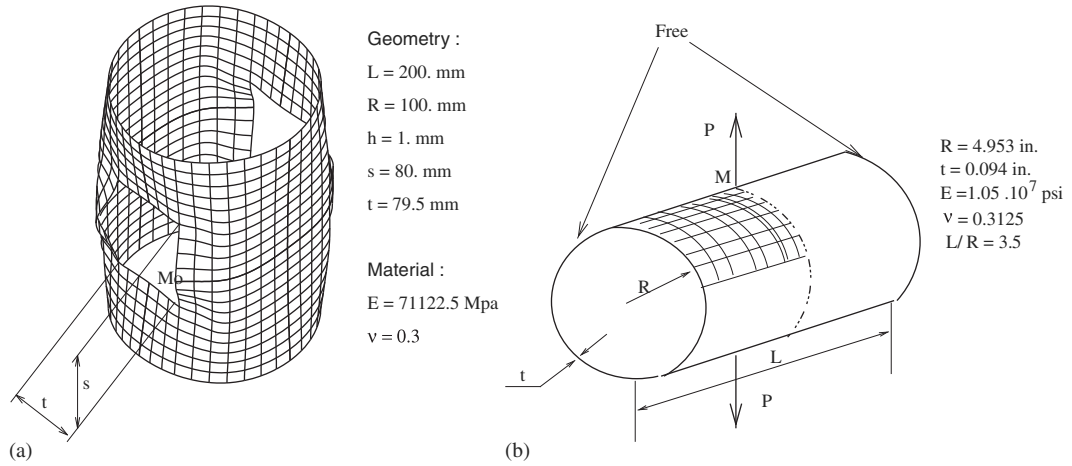

Figure 1. Geometrically non-linear shell problems. Tests 1 and 2: (a) cut-out cylinder and (b) pull-out of an open cylinder.

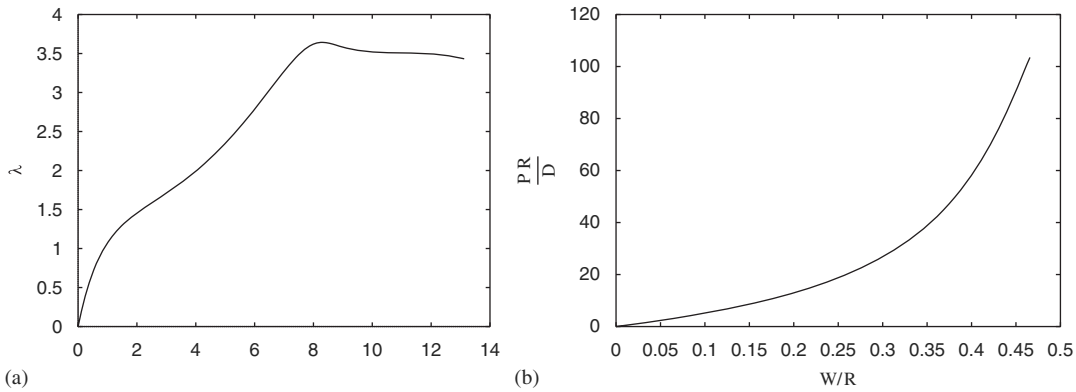

Figure 2. Load–displacement curves at the loaded point of the geometrically non-linear shell problems: (a) reference curve of the cut-out cylinder and (b) reference curve of the pull-out cylinder.

triangulated at each step. The prediction step is carried out by using a direct solver. The corrections are achieved by the iterative linear solvers presented in Part 2, the preconditioning matrix K^* being the triangulated matrix of the prediction step. The prolongation operator $\mathbb{P}$ is built with the already computed solutions U, coming from the previous prediction–correction steps. This operator is then upgraded at each prediction–correction step. Consequently, the reduced basis size 'n' increases with the number of prediction–correction steps.

In all the numerical tests presented in this study, we consider that the linear solvers have converged when the following criterion is satisfied:

$$\frac{\|KU - F\|}{\|F\|} < \eta_{\text{lin}} \tag{22}$$

where $\|\bullet\|$ is the Euclidian norm and η_{lin} is a small parameter ($10^{-8} \leqslant \eta_{\text{lin}} \leqslant 10^{-4}$).

3.2. Evaluation of the proposed solver

The proposed iterative solver is first evaluated by studying its convergence within the first corrective step of a Newton–Raphson algorithm, the preconditioning matrix and the prolongation operator being defined in Section 3.1. The numerical results will be compared with those obtained with the preconditioned conjugate gradient method, the preconditioning matrix K^* being the same. In the

next Figures 3–6, the convergence is represented by curves relating the logarithm of the residual of the linear problem versus the iteration number. The results obtained with the 'proposed method' and with the 'preconditioned conjugate gradient' method will be referred as 'PM' and 'PCG', respectively.

A first example is presented in Figure 3, the physical model being the cut-out cylinder. In this case, we have also considered the linear solver based on the HPP method (Equation (19), denoted by 'linear solver' from Reference [10] in Figure 3). First, one observes that the number of iterations is very small: one can get a residual of 10^{-5} after only 4 iterations. This is due to the choice of a good preconditioner and also the efficiency of the proposed algorithm. Second remark: the proposed method converges more rapidly than the preconditioned conjugate method, with the same preconditioning matrix. Third, the proposed method and the HPP solver of the Section 2.3 lead to the same convergence curve: this is in accordance with the results of Reference [17], where one has established the mathematical equivalence between a sequence accelerated by MMPE and the 'homotopy–perturbation–Padé' method. Consequently, the HPP will be no longer considered in the following numerical tests.

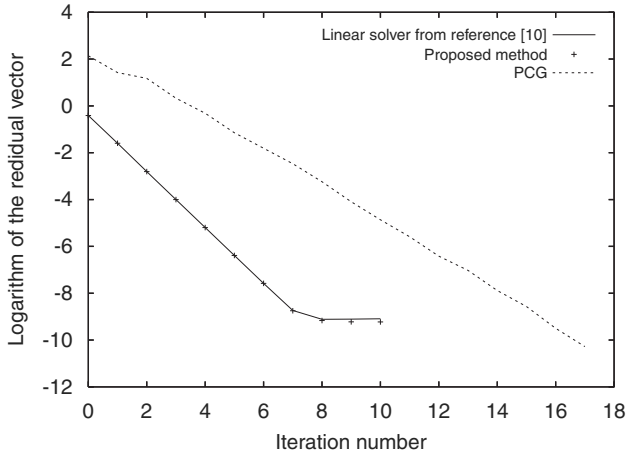

Figure 3. Convergence curve. Comparison between the proposed solver (Equation (8)), the linear solver coming from Reference [10] and the preconditioned conjugate gradient (PCG). The size of the basis is $n = 20$. Cut-out cylinder.

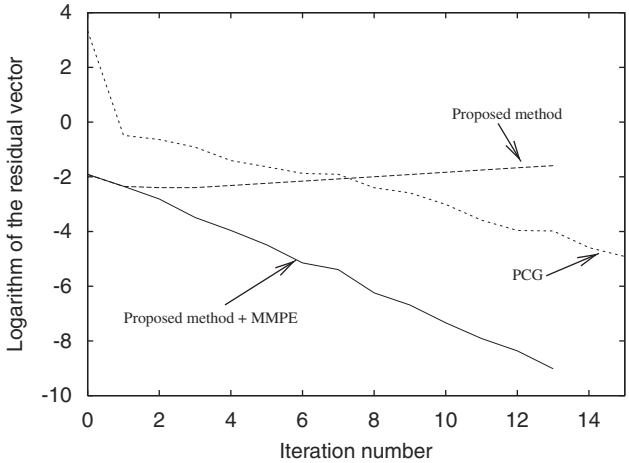

Figure 4. Convergence curve. Comparison between the extrapolated vector (expression 10) and the initial sequence (expression 9) with $n = 40$. Cut-out cylinder.

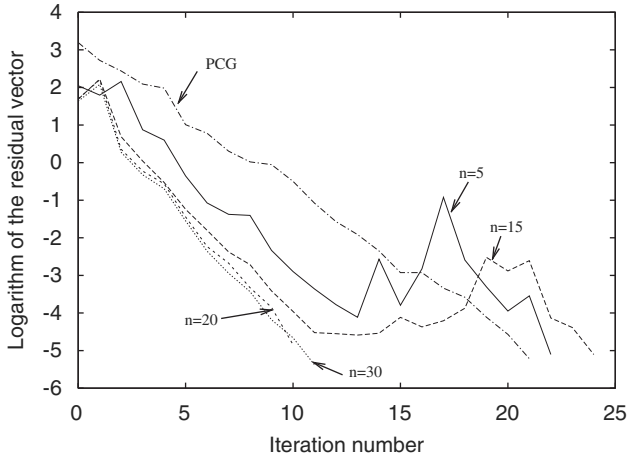

Figure 5. Convergence curves for several sizes of the basis. Cut-out cylinder.

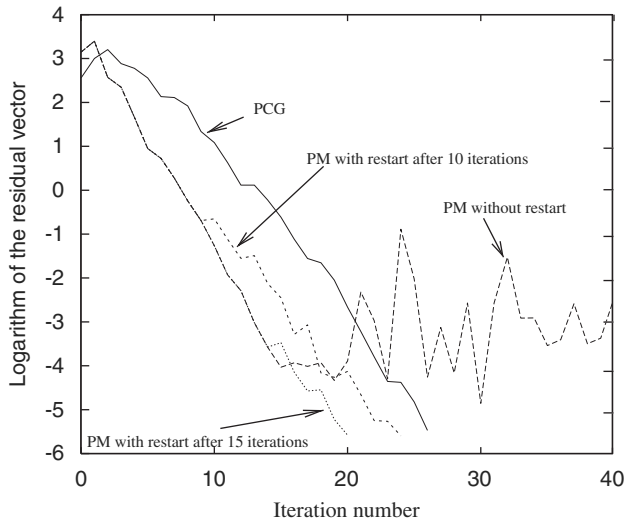

Figure 6. Comparison between the proposed method+MMPE with or without restart. The size of the reduced basis is $n=5$. Cut-out cylinder.

Another example is presented in Figure 4, where the sequence (9) diverges, despite of the great size of the reduced problem ($n=40$). The initial sequence has been replaced by the extrapolated vector (10), which leads to a sequence that converges more quickly than the PCG. Hence, it is very interesting to use a convergence acceleration technique that has moreover a negligible computational cost [18].

The influence of the size of the basis ($5 \leqslant n \leqslant 30$) is discussed in Figure 5. In this case, the accuracy is not improved beyond 20 vectors in the basis. A similar result has also been noted in other tests with a larger number of unknowns. This can be explained by the fact that the gap between the operator K^*, the prediction matrix and the consistent correction matrix K is small. In this case, this leads to a good preconditioning matrix and the influence of the reduced problem on the convergence is less important.

This figure also illustrates one of the characteristics of the MMPE. Indeed, the accelerated sequence can become unstable or the convergence is not improved when the number of vectors composing the sequence is high. In this case, a solution is to apply the restart algorithm as defined in Section 2.2. In Figure 6, one can see that without using the restart algorithm the proposed

Table I. Average number of iterations for various linear solvers.

Cut-out cylinder				
	Average number of iterations			
Newton–Raphson Step	PM	PCG	PCG IC[0]	PCG IC[1]
4	6	11	170	110
5	6	11	176	111
6	7	12	178	113
7	8	17	184	121
8	9	20	193	124
9	10	21	198	130
10	9	21	198	135

The proposed method (PM) is compared with the conjugated gradient associated with three preconditioners: the same as in PM (PCG) and the incomplete Crout triangulations of level 0 and 1 (IC[0] and IC[1]). Required accuracy $\eta_{\text{lin}} = 10^{-4}$ for the linear solver and $\eta_{\text{nl}} = 10^{-3}$ for the non-linear solver. Each Newton–Raphson step includes the prediction and 2 or 3 iterations. In PM, the size of the reduced basis varies between 15 and 37.

Table II. Average number of iterations for the proposed method and the conjugated gradient associated with the same preconditioner.

Pull-out cylinder		
	Average number of iterations	
Newton–Raphson Step	PM	PCG
4–6	11	46
7–12	6	27
13–15	3	12
16–23	7	20

The required accuracies are the same as in Table I. Each Newton–Raphson step includes the prediction and 2 or 3 iterations. The size of the reduced basis varies between 17 and 50.

method does not reach the desired accuracy ($\approx 10^{-5}$). Whereas, when the restart algorithm is used after 10 or 15 iterations, the proposed linear solver converges.

3.3. Two full path following computations

Let us now apply the proposed iterative solver for the whole computation of non-linear solution branches in order to demonstrate its robustness. We consider the two tests described in Section 3.1: cut-out cylinder and pull-out open cylinder pulled by two diametrically opposite sides. The prediction step is done with a direct method and the iterative solver is used in the correction phase. The accuracy on the linear and non-linear problems are, respectively, chosen equal to 10^{-4} and 10^{-3}. The prediction step length is chosen in such a way that 2 or 3 iterations per step are needed. This leads to 10 steps to solve the cut-out cylinder problem and 23 for the open cylinder (the sought response curve is in the range $0 < PR/D < 100$).

Owing to the efficiency of the preconditioning matrix, the proposed method (PM) and the preconditioned conjugate gradient (PCG) converge in any case. The average number of iterations for these linear solvers is reported in Tables I and II. With the proposed method, the number of iterations is very small, generally lower than 10. The conjugate gradient associated with the same preconditioning matrix (PM) is more expensive: the number of iterations is twice larger in the first example and four times larger in the second one that has a larger number of degrees of freedom.

For comparison, we have also evaluated the conjugate gradient associated with an incomplete Crout factorizations [27] of the matrix of the prediction step. In Table I, this preconditioner is referred to as IC[m], where m is the 'level' of the incomplete factorization. As it is well known,

this technique requires much less memory and much less computation time per iteration than the previous one. The drawbacks are also known and corroborated by our numerical results. Indeed the number of iterations is much higher than with a complete factorization: in the first test (cut-out cylinder), the number of iterations is about 200 with IC[0] and 100 with IC[1]. In the second test (pull-out cylinder), the algorithm diverges. Likely this could be removed by introducing an appropriate stabilization parameter, but this confirms the lack of robustness of incomplete factorization techniques with ill-conditioned problems as in shell analysis.

4. PREDICTION–CORRECTION STRATEGIES

Various prediction–correction strategies can be established from the linear iterative solver presented in Part 2. As recalled in Section 2.4, a Newton–Raphson step leads to $N_{iter}+1$ linear problems. These linear problems will be solved, either by a direct method, either by the iterative solver. In the two first strategies that are discussed in this part, only one matrix per step is triangulated and the corresponding linear problem is solved by the direct method, while all the other problems are solved iteratively, the preconditioner being the last triangulated matrix. More precisely, the prediction problem is solved by the direct method in the first strategy and the linear problem of the first iteration in the second strategy. They will be referred as 'one matrix algorithms', because they require only one matrix triangulation per step. By comparison, the Newton–Raphson method requires classically $N_{iter}+1$ matrix triangulations or, at least, two in the so-called 'MNM'. With the third strategy, one tries to establish a matrix-less algorithm. More precisely, the direct solver is no longer used at each step, but only when the number of iterations increases too much. In the following numerical experiments, we limit ourselves to 10 iterations before applying the convergence acceleration process (i.e. $I=10$ in formula (10)) and to 2 cycles '10 iterations + MMPE'. In other words, a new matrix is factorized as soon as the number of iterations is beyond 20. These three strategies are sketched in Table III.

The efficiency of these strategies is assessed by two examples. The first one is the open cylinder (Figure 1). The second one is another classical benchmark: a hemispherical shell with a hole and loaded by pinching forces (F=1.) [28]. The geometric and material descriptions are given in Figure 7. The response curves of the two loaded points are plotted in Figure 7. The mesh is the same as the one used in [26] (108 eight-node quadrilateral elements with 2166 d.o.f.). In our computations, 30 prediction–correction steps are needed to obtain the non-linear solution branch (Figure 7). The corresponding total number of linear problems to be solved is equal to 139. In Reference [26], this branch had been obtained by the commercial code Abaqus with a cost of 104 tangent matrix triangulations, which is relatively close to our results. For the pull-out cylinder, the number of prediction–correction steps is 23 and 86 linear problems are to be solved. Because the solver is based on a reduced basis, the first three steps are computed by classical direct method, which permits to compute at least 10 basis vectors. The demanded accuracies are the same as in Section 3.3 : 10^{-3} for the non-linear iterations and 10^{-4} for the linear ones.

The numerical costs depends on the number of iterations and of matrix factorizations. These quantities are reported in Tables IV and V, for the two tests and for the three computational strategies. First, one observes the strategy 2 is much more efficient than the other 'one matrix algorithm' (strategy 1), the number of iterations being about half. This can be explained by the fact that the matrix of the first Newton iteration is a better preconditioner (for the next iterations) than

Table III. Description of the three prediction–correction strategies. DS and PM design, respectively, a direct solver and the proposed linear solver.

	Strategy 1	Strategy 2	Strategy 3
Prediction	DS	PM	DS or PM
Correction	PM	Iteration 1 : DS Iteration >1 : PM	DS or PM

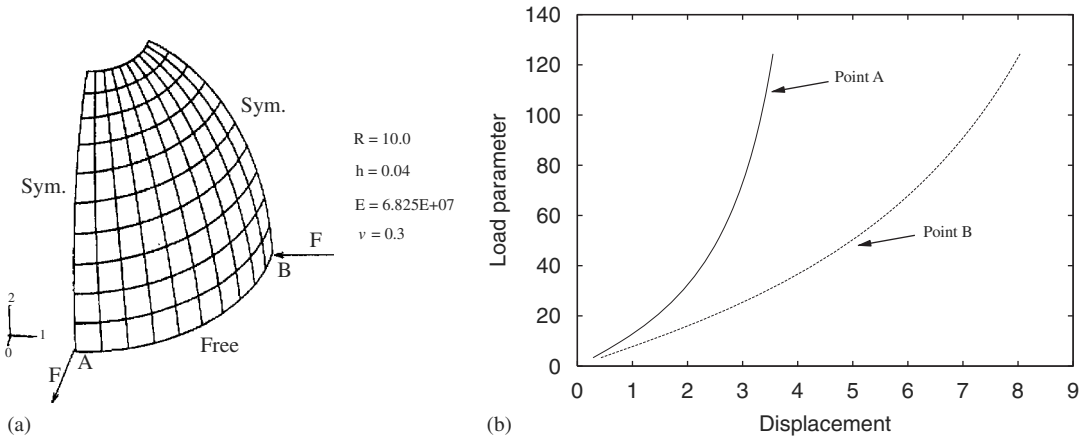

Figure 7. Example of the pinched hemisphere: (a) geometry and material descriptions and (b) displacement of points A and B versus the load parameter. The curve is obtained with 30 steps of the Newton–Raphson method.

Table IV. Evaluation of the three strategies to update the preconditioner.

Pinched hemisphere						
	Average number of iterations per step			Number of matrices		
Steps	Strategy 1	Strategy 2	Strategy 3	Strategy 1	Strategy 2	Strategy 3
4–8	24	8	14	5	5	4
9–13	10	4	13	5	5	2
14–30	6	2.5	12	17	17	4

Table V. Comparison of the three strategies to update the preconditioner. Average number of iterations per step.

Pull-out cylinder			
	Strategy 1	Strategy 2	Strategy 3
Iterations per step	7	2.75	13
Matrices	20	20	4

the prediction matrix. Another consequence: the prediction problem (20) is also an ‘easy’ problem that is generally solved with less than 5 iterations. As for the ‘matrix-less algorithm’, it permits to reduce hugely the number of matrix factorizations: for instance, in the pull-out cylinder problem, only 4 factorizations are necessary, to be compared with 20 for the ‘one matrix algorithms’ and 76 for the direct solver. The decrease of the number of matrices does not lead to a consequent increase of iterations. More surprisingly, in some cases (Table IV) there are even less iterations than with the strategy 1. This can also be explained by the fact that in strategy 3, the triangulated matrix comes always from the Newton’s correction phase, and not from the prediction step, as in strategy 1.

Hence an algorithm based on sparse matrix factorizations, as strategy 3, seems the most attractive technique, especially for large-scale problems since the cost of the matrices is much larger than the cost of one iteration.

5. CONCLUSION

In this paper, a new linear iterative solver has been proposed, that is designed to solve the numerous linear equations arising in the solution of a non-linear system. It associates a reduced basis technique, a full preconditioning matrix and an accelerating convergence method, the modified minimal polynomial extrapolation (MMPE). It converges more rapidly than the preconditioned conjugate gradient method and as rapidly as a recently introduced algorithm [10] that relies on homotopy transformation, perturbation technique and Padé approximants. The process is robust and leads to convergence with a small number of iterations.

The reliability and the efficiency of the procedure have been proved by several shell buckling non-linear computations within a Newton–Raphson framework. When it is applied to solve non-linear problems, one gets algorithms that need one matrix factorization per step or one factorization for several Newton–Raphson steps.

This strategy could also be applied in the asymptotic numerical method (ANM) [1] to compute several steps of the continuation methods with a single matrix factorization. Note that, within ANM, it is very easy to build up a relevant reduced subspace including a rather large number of vectors. There are also possible applications in the computation of non-linear dynamics problems with an implicit scheme.

In this paper, the number of iterations and the speed of convergence have permitted us to discuss the efficiency of the algorithm. Of course, a direct measure of the computation time would be more convincing, but only for tests with many degrees of freedom on parallel computers and with the help of modern direct solvers as MUltifrontal Massively Parallel sparse direct Solver (MUMPS [29]). We are now working in this way.

REFERENCES

1. Cochelin B, Damil N, Potier-Ferry M. Asymptotic-numerical method and Padé approximants for non-linear elastic structures. *International Journal for Numerical Methods in Engineering* 1994; **37**:1187–1213.
2. Gutknecht MH. See the web site: http://www.sam.math.ethz.ch/mhg/.
3. O'Leary D. The block conjugate gradient algorithm and related methods. *Linear Algebra and its Applications* 1980; **29**:293–322.
4. Toutounian F, Karimi S. Global least squares method (GL-LSQR) for solving general linear systems with several right-hand sides. *Applied Mathematics and Computation* 2006; **178**:452–460.
5. Bouyouli R, Jbilou K, Messaoudi A, Sadok H. On block minimal residual methods. *Applied Mathematics Letters* 2007; **20**(3):284–289.
6. Saad Y, Yeung M, Erhel J, Guyomarc'h F. A deflated version of the conjugate gradient algorithm. *SIAM Journal on Scientific Computing* 2000; **21**:1909–1926.
7. Risler F, Rey C. On the use of Ritz vectors for the solution to nonlinear elasticity problems by decomposition methods. *Contemporary Mathematics* 1998; **218**:334–340.
8. Clemens M, Wilke M, Schumann R, Weiland T. Subspace projection extrapolation scheme for transient field simulations. *IEEE Transactions on Magnetics* 2004; **40**(2):934–937.
9. Clemens M, Helias M, Steinmetz T, Wimmer G. Multiple right hand side techniques for numerical simulation of quasistatic electric and magnetic fields. *Journal of Computational and Applied Mathematics* 2008; **215**(2):328–338.
10. Cadou JM, Damil N, Potier-Ferry M, Braikat B. Projections technique to improve high-order iterative correctors. *Finite Elements in Analysis and Design* 2004; **41**:285–309.
11. Damil N, Potier-Ferry M, Najah A, Chari R, Lahmam H. An iterative method based upon Padé approximants. *Communications in Numerical Methods in Engineering* 1999; **15**:701–708.
12. Mallil E, Lahmam H, Damil N, Potier-Ferry M. An iterative process based on homotopy and perturbation techniques. *Computer Methods in Applied Mechanics and Engineering* 2000; **190**:1845–1858.
13. Brezinski C, Van Iseghem J. Padé approximants. In *Handbook of Numerical Analysis*, Ciarlet PG, Lions JL (eds), vol. 3. North-Holland: Amsterdam, 1994.
14. Baker GA, Graves-Morris P. Padé approximants. *Encyclopedia of Mathematics and its Applications* (2nd edn). Cambridge University Press: Cambridge, 1996.
15. Najah A, Cochelin B, Damil N, Potier-Ferry M. A critical review of asymptotic numerical methods. *Archives of Computational Methods in Engineering* 1998; **5**:3–22.
16. Jbilou K, Sadok H. Vector extrapolation methods, applications and numerical comparison. *Journal of Computational and Applied Mathematics* 2000; **122**:149–165.
17. Damil N, Cadou JM, Potier-Ferry M. Mathematical and numerical connections between polynomial extrapolations and Padé approximants. *Communications in Numerical Methods in Engineering* 2004; **20**(9):699–707.

18. Cadou JM, Duigou L, Damil N, Potier-Ferry M. Convergence acceleration of iterative algorithms. Applications to thin shell analysis and Navier–Stokes equations. *Computational Mechanics* 2009; **43**(2):253–264.
19. Brezinski C. Convergence acceleration during the 20th century. *Journal of Computational and Applied Mathematics* 2000; **122**:1–21.
20. Andrianov IV, Awrejcewicz J. Asymptotic solution of the theory of shells boundary value problem. *Mathematical Problems in Engineering* 2007; DOI: 10.1155/2007/82348.
21. He JH. Comparison of homotopy perturbation method and homotopy analysis method. *Applied Mathematics and Computation* 2004; **156**:527–539.
22. Liao S. Beyond perturbation. *Introduction to the Homotopy Analysis Method*. Chapman & Hall/CRC: Boca Raton, U.S.A., 2004.
23. Riks E. Some computational aspects of the stability analysis of nonlinear structures. *Computer Methods in Applied Mechanics and Engineering* 1984; **47**(3):219–259.
24. Batoz JL, Dhatt G. *Modélisation des structures par éléments finis*. Editions Hermès, Paris, 1990.
25. Sze KY, Liu XH, Lo SH. Popular benchmark problems for geometric nonlinear analysis of shells. *Finite Elements in Analysis and Design* 2004; **40**(11):1551–1569.
26. Zahrouni H, Cochelin B, Potier-Ferry M. Computing finite rotations of shells by an asymptotic-numerical method. *Computer Methods in Applied Mechanics and Engineering* 1999; **175**:71–85.
27. Joly P, Vidrascu M. *Quelques méthodes classiques de résolution de systèmes linéaires*, *Collection—Didactique*. INRIA, 1994.
28. Büchter N, Ramm E. Shell theory versus degeneration—a comparison in large rotation finite element analysis. *International Journal for Numerical Methods in Engineering* 1992; **34**:39–59.
29. MUltifrontal Massively Parallel sparse direct Solver (MUMPS). Available from: http://graal.ens-lyon.fr/MUMPS/index.html.