Abstract

The conjugate gradient method (CG), an iterative method of solving linear systems, has been optimized in conjunction with two preconditioners, the incomplete Cholesky decomposition (IC) and a truncated Neumann series. In both cases, the preconditioning technique generally used has been adapted and fine-tuned to the application described. The first (IC) involves a generalization of the conventional technique, that is it covers all spectral patterns across the lower triangular matrix L. Secondly, the truncated Neumann series takes account of an innovation borrowed from IC. The implementation of the above procedures in finite element method (FEM) analysis is described, with the emphasis on how the innovations lead to results relevant to the application of these models in the optimization of system solving. Finally, there follow some remarks on the implementation of these techniques in parallel computing environments.

1 Introduction

The increasing use of simulation of complex structural models by the finite element method in the field of structural engineering – more specifically in numerical applications – has called for both the manipulation of large amounts of data, intrinsic to this method, and also a search for reduced response times to solve the linear system generated.

Despite the impact that the microelectronic sector has had on the development of computer components, particularly on more compact memory systems and ever faster processors^1, these powerful machines alone are not always capable of dealing adequately with a variety of structural models, whether it be due to insufficient memory or to excessively long response times.

The use of interconnected computer systems to solve specific problems, however, has become widespread over the last 20 years, e.g. [2, 3]. It should be pointed out that these systems, commonly known as parallel processing analysis systems, require the readaptation of traditional numerical procedures such as the resolution of linear systems.

A result of the parallel computing architecture, then, is that the scientific community is seeking to use methods of solving linear systems that have already been dropped because of their specific deficiencies but that can now be adapted to the new computational model. Among these, the growing use of iterative methods stands out against that of the traditional direct methods for linear system resolution.

Direct methods used to be commonly employed, as they are stable and sturdy and the order and number of operations to be carried out is, *a priori*, known, and they are thus well suited to FEM applications. On the other hand, these methods have the disadvantage that the operation among the different lines of the stiffness matrix are extremely interdependent while, in this architecture, the more independent a numerical method, the more efficient its use, [3, 4, 5, 6, 7].

Iterative methods have become widely applied since they allow the system to be solved without requiring the use of several interconnected computers. Recently, therefore, it has become commonplace to solve linear systems by means of iterative methods, e.g. [4], such as Gauss-Seidel, Jacobi, Quasi-Newton (BFGS) and the Conjugated Gradient Method (CG method). This paper focuses on the CG method owing to its versatility and easy implementation in both sequential and parallel environments, and because the method is currently the most widely employed in parallel processing analyses, [3, 4].

The major problem that is characteristic of the CG method is its numerical instability in certain configurations of the linear system, specifically of the conditioning of the matrix. If the system is well-conditioned, the method converges rapidly; otherwise, convergence is unsatisfactory, which makes the method inadequate to solve the system.

Preconditioning techniques, which are simply optimizers that reduce the instability of the method itself, have been employed to solve this problem. In the literature, it is common to find the CG method associated with the use of an adequate preconditioning technique in order to give it an advantage over the traditional direct methods.

Thus, the iterative CG method is presented here, optimized by means of preconditioning techniques. The two techniques employed are Incomplete Cholesky Decomposition (IC) and a truncated Neumann series. Both these preconditioning techniques are improvements over those in the literature, see [8, 9, 10, 11, 12, 13].

The first technique, Incomplete Cholesky Decomposition (IC), is developed considering a generalization of the conventional technique, while the second, the truncated Neumann series, involves an innovation taken from the IC technique.

Examples of these techniques are shown for *sequential processing* applied to structural examples of a sheet and plate, demonstrating that these innovations offer satisfactory results for the use on models of such structures.

To conclude, comments are made on the use of these techniques developed on the potential implementation of these techniques in parallel processing, highlighting the advantages of each improved technique in the new computational environment.

2 Conjugate Gradient Method

Consider the linear system

$$[K].\{U\} = \{F\} \tag{1}$$

where [K] is the stiffness matrix and {U} and {F} are displacement and force vectors, respectively. The conjugate gradient (CG) method, [3, 4, 9, 15], which is based on the strategy of seeking the path of *steepest descent*, may be applied to [K] as long as this matrix is symmetric and positive definite. In the strategy of steepest descent, a vector $\{U_0\}$ is chosen as starting value and, by means of successive approximations $\{U_1\}$, $\{U_2\}$ .. $\{U_n\}$, the value that satisfies (1) is found.

The minimization rate, represented by the scalar α, is given by:

$$\alpha_{CRITIC} = \frac{\{r_p^T\}\cdot\{r_p\}}{\{r_p^T\}\cdot[K]\cdot\{r_p\}} \tag{2}$$

where $\{r_p\}$ is the residual vector, given by:

$$\{r_p\} = \{F\} - [K]\cdot\{U_0\} \tag{3}$$

The *(p+1)*the approximation of vector {U} is then:

$$\{U_{p+1}\} = \{U_p\} + \alpha_{critic}\cdot\{r_p\} \tag{4}$$

The pseudocode is shown in Figure 1.

```
→ j = 0,
{U0} = {0}
 {r0} = {F}
→ While $\|\{r_j\}\| < \varepsilon$
        → j =j + 1
        → $\alpha_j = \frac{\{r_{j-1}^T\}\cdot\{r_{j-1}\}}{\{r_{j-1}^T\}\cdot[K]\cdot\{r_{j-1}\}}$
        → $\{U_j\} = \{U_{j-1}\} + \alpha_j \{r_{j-1}\}$
        → $\{r_j\} = \{F\} - [K]\{U_j\}$
→ Endwhile
```

Figure 1 : Pseudocode for the method of steepest descent.

A potential problem with the steepest descent strategy is slow convergence, since nothing prevents a particular step repeating the direction of an earlier one, resulting in an oscillation about a point that is not the desired solution.

The CG method improves on this situation in that the search directions $\{p_0\}$, $\{p_1\}$,..,$\{p_{n-1}\}$ are orthogonal to those previously calculated. In each of these directions is found one of the coordinates of {U}, so that after *n* steps, where *n* is the dimension of {U}, the procedure ends and the required solution is found.

The vector {U} evolves in accordance with:

$$\{u_{j+1}\} = \{u_j\} + \alpha_j\{p_j\} \tag{5}$$

The following condition guarantees orthogonality between the directions of $\{p_i\}$ and $\{p_{i+1}\}$, such that they are K-orthogonal:

$$\{p_i^T\}\cdot[K]\cdot\{p_j\} = \{0\}\text{, para i} \neq \text{j} \tag{6}$$

Pseudocode for the CG method is presented in Figure 2.

Theorems (1) and (2), proofs of which are given, respectively, in refs. [4] and [8], complete the convergence requirements of the method.

Theorem 1: If search directions $\{p_0\}$, $\{p_1\}$,..,$\{p_{n-1}\}$ are K-conjugated (K-orthogonal) and scalar α_j is chosen such that: $\alpha_j = -\frac{\{p_j^T\}\cdot[K]\cdot\{r_j\}}{\{p_j^T\}\cdot[K]\cdot\{p_j\}}$; then the procedure terminates in, at most, *n* steps, where *n* is the dimension of the square matrix [K] of the system in (1).

Theorem 2: If [K] is symmetric and positive definite, the CG algorithm will produce a sequence of vectors $\{U_0\}$, $\{U_1\}$,..,$\{U_j\}$..., with the following property:

$$\left\|\{U\} - \{U_j\}\right\|_K \le 2\cdot\left\|\{U\} - \{U_0\}\right\|_K \cdot \left(\frac{\sqrt{\kappa}-1}{\sqrt{\kappa}+1}\right)^{\kappa} \tag{7}$$

where $\kappa=\lambda_{max}/\lambda_{min}$ and λ_{max} and λ_{min} are, respectively, the highest and lowest eigenvalues of [K] and **κ** is its condition number.

In (7), two factors are seen to influence convergence in the CG method: the initial estimate $\{U_0\}$ and the condition number of the matrix, **κ**. In the literature, the value of $\{U_0\}$ is said to be of little relevance to the efficiency of CG. Meanwhile, in refs. [10] have reported an optimization scheme in which a starting value for vector $\{U_0\}$ is produced. In recent years, researchers in the artificial intelligence (AI) field have paid more attention to this item, the strategy being to create data-banks of deformations and/or stresses in given structural models and thence apply special algorithms (known as meta-algorithms) to optimize this approach, see [5].

Note, in (7), that variation of the condition number will have a more significant effect on convergence than that of $\{U_0\}$. Thus, the greater the spread of the eigenvalues of [K], the slower is the convergence of the CG method. In fact, the biggest obstacle to using this method is precisely the requirement that the condition number of [K] be close to unity.

In the last five years, a lot of research has gone into mathematical techniques to produce condition numbers as close to one as possible. The idea on which this work is based is that, instead of the system [K] · {U} = {F}, another system of the type [M]·[K]·{U} = [M]·{F} is solved. [M] is chosen to make the product ([M]·[K]) as near as possible to the identity matrix, since this has a condition number of 1. [M] is known as a *preconditioner*, which is just a convergence accelerator for the method.

```
→ $p_0 = r_0 = \{F\} - [K]\,\{U_0\}$
→ j = 1
→ **While**
        → $\alpha_j = \dfrac{\{r_j^T\}\cdot\{r_j\}}{\{p_j^T\}\cdot[K]\cdot\{p_j\}}$
        → $\{U_{j+1}\} = \{U_j\} + \alpha_j\,\{p_j\}$
        → $\{r_{j+1}\} = \{r_j\} - \alpha_j\,[K]\,\{p_j\}$
        → $\beta_j = \dfrac{\{r_{j+1}^T\}\cdot\{r_{j+1}\}}{\{r_j^T\}\cdot\{r_j\}}$
        → $\{p_{j+1}\} = \{r_{j+1}\} + \beta_{j+1}\,\{p_j\}$
        → j =j + 1
→ **Endwhile**
```

Figure 2 : Pseudocode for the CG Method

3 Preconditioners

One way of accelerating the convergence of the CG method is to try and reduce the condition number (**κ**) of the stiffness matrix [K]. This may be achieved by transforming the system shown in equation (1) into an equivalent, but better conditioned, one.

Such *preconditioning* converts (1) into:

$$[\tilde{K}]\cdot[\tilde{U}] = [\tilde{F}] \qquad (8)$$

in which the preconditioned matrix $[\tilde{K}]$ must have the same characteristics as the stiffness matrix [K], i.e. it must be symmetric and positive-definite, so as to guarantee equivalence of the systems and fulfill the requirements of the CG method. Apart from this, $[\tilde{K}]$ has to be a good approximation to [K], but with a structure that permits easy matrix inversion, since in practice the object of preconditioning is to obtain an approximation to $[K]^{-1}$ efficiently. Thus, the initial system, seen in (1), is pre-multiplied by the inverse of the preconditioning matrix:

$$([M]^{-1}\cdot[K])\cdot\{U\} = [M]^{-1}\cdot\{F\} \qquad (9)$$

so that as $[M]^{-1}$ approaches $[K]^{-1}$, **κ** $[\tilde{K}]$ approaches 1 and the following relation holds:

$$\kappa(\tilde{K}) << \kappa(K) \qquad (10)$$

$$\lim_{[M]^{-1}\to[K]^{-1}} \kappa(\tilde{K}) = 1$$

In the limit, when the inverse of [K] is reached, the CG method becomes direct.

However, obtaining an exact inverse of [K] would entail much more work for the computer than solving the system without the aid of a preconditioner. Hence, the study of these preconditioners is aimed at finding a particular matrix, as nearly an inverse of [K] as possible, yet requiring less computation for its construction while ensuring convergence of the equivalent system.

3.1 Algorithm for the CG method with preconditioning

Pseudocode for the pure CG method, as written in figure 2, may be adapted to take into account the effect of preconditioning. This may be done by applying the preconditioner to the residual vector $\{r_p\}$, rather than pre-multiplying equation (1) as in (9). The preconditioned vector is then:

$$\{\tilde{r}_p\} = [M]^{-1}\cdot\{r_p\} \qquad (11)$$

This procedure is also known as the Implicit CG Method.

3.2 Incomplete Cholesky decomposition

The incomplete Cholesky decomposition (or factorization), IC, was first used in conjunction with the CG method by [8]. In refs. [11] is demonstrated that the [K] matrices generated by FEM, despite their not having M-matrix properties, can be treated by IC as in [8], resulting in good convergence in problems analyzed by FEM.

3.3 Assembling the IC preconditioner

The idea of the IC method is to obtain the preconditioning matrix by factorizing (Cholesky decomposing) [K] as follows:

$$[K] = [L]\cdot[L^T] \qquad (12)$$

where [L] is a lower triangular matrix. This procedure is no more than the familiar LU-decomposition, [16].

When [K] is factorized into [L], the latter is not as sparse, in outside the field of elements defined by the bandwidth, as [K]. This means [L] must be stored in full, causing a serious "bottleneck" in the numerical solution of the problem, due to the increase in memory overheads and number of data to process.

To remedy this bottleneck, the IC method performs the [L]-factorization such that elements in certain appropriate positions are neglected thus neither stored nor processed, these positions being picked so as to maintain the original sparsity of [K] in the triangular matrix. Hence the name Incomplete Cholesky Decomposition.

The actual elements of the triangular matrix that are made zero may be chosen arbitrarily while the non-zero positions (i,j) belong to a set P described as follows:

$$P \subset P_N \equiv \{ (i , j) \mid i \neq j,\ 1 \leq i \leq n,\ 1 \leq j \leq i \} \qquad (13)$$

where P_N is the set of all matrix positions. Thus, a field of positions to be filled in matrix [L] is chosen. For example, let this field comprise set P^1, in which (i,j) indicates positions of non-zero elements:

$$P^1 \equiv \{ (i , j) \mid |i - j| \neq 0, 1, 2 \} \qquad (14)$$

Such a field is the one sketched in Figure 3.

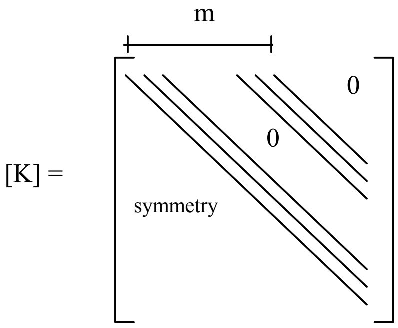


Figure 3: Field of element values in [K] included in [L]

In [8] is pointed out that the non-neglected elements of [K] may not be adjacent, since in the great majority of problems arising in solid mechanics, involving either finite difference or FEM analyses, the interactions observed among the matrix parameters are not restricted to the vicinity of the leading diagonal. Typically, positions next to both that diagonal and the farthest non-zero one are filled, while the work cited makes it clear that the usual practice is to consider only the positions in set P.

As an example, in figure 4, the first 3 diagonals and the farthest 3 non-zero diagonals of matrix [K] are utilized, *m* being the half bandwidth. Thus, P is given by:

$$P^3 \equiv \{ (I , j) \mid |i - j| \neq 0, 1, 2, m-3, m-2, m-1 \} \qquad (15)$$

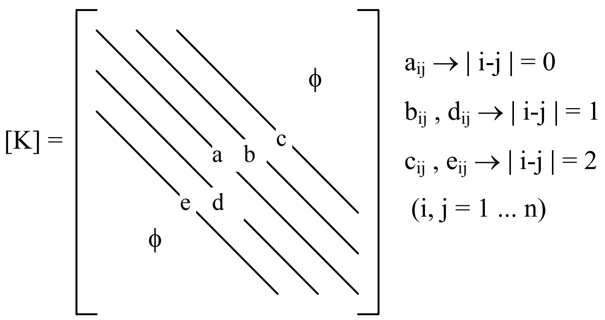

[K] =
ϕ
a
b
c
e
d
ϕ
$a_{ij} \rightarrow | i-j | = 0$
b_{ij} , $d_{ij} \rightarrow | i-j | = 1$
c_{ij} , $e_{ij} \rightarrow | i-j | = 2$
(i, j = 1 ... n)

Figure 4: Field P^3 applied over [K]

The method of conjugate gradients, used together with preconditioning by incomplete Cholesky decomposition, is frequently abbreviated to ICCG (η_b), where η_b is the bandwidth, here the number of diagonals included in P apart from the 3 that belong to set P^1, in (14). Thus, for the field in figure 4, the notation is ICCG(3).

3.4 Types of spectra over [L]

For the current investigation, an algorithm was developed in which the spectrum of influence over the non-zeroed positions in matrix [L] is itself an input variable. The ranges of influence of the diagonals, starting both at the leading diagonal and at the most distant non-zero diagonal, are thus parameters to be chosen, as in figure 5.

The motive behind this extension of the technique was to increase the accuracy of the calculated inverse of [K], by broadening the spectrum to include degrees of freedom that have secondary, yet significant, influence over the leading diagonals.

Given that the values of m_1 and m_2 may not be the same, the abbreviation used by [8] cannot now be employed. Therefore a new abbreviation, specific to the present method, has been created to include the definition of m_1 and m_2. Consider, for example, a field on set P, such that $m_1 = 4$ and $m_2 = 3$. Then, the set P would be labeled $P_{m_1}^{m_2}$ and given by:

$$P_4^3 \equiv \{ (i , j) \mid |i - j| \neq 0, 1, 2,3,m-3,m-2,m-1 \} \qquad (16)$$

The notation for the combination of the CG method with this general-interaction preconditioner is: ICCG(m_1,m_2); the last example is thus ICCG(4,3).

3.5 Polynomial preconditioner

A further classic method of finding an approximation to the inverse of [K] is polynomial preconditioning, which makes use of an expansion in powers of a matrix. Given that [K] must be symmetric and positive-definite, it mat be rewritten as follows:

$$[K] = [D_S] + [\bar{K}] \qquad (17)$$

where $[D_S]$ is a matrix containing only the leading diagonal elements of [K], while $[\bar{K}]$ is the matrix [K] with all leading diagonal elements set to zero.

Separating the factor $[D_S]$ and performing matrix manipulation on the terms in (17), the inverse of [K] is given by:

$$[K]^{-1} = ([I] + [D_S]^{-1} \cdot [\bar{K}])^{-1} \cdot [D_S]^{-1} \qquad (18)$$

[I] being the identity matrix.

If [K] is strictly diagonally dominant, the first part of the right side of this equation may be expressed as a power series:

$$([I]+[D_S]^{-1}\cdot[\bar{K}])^{-1}=\sum_{J=0}^{\infty}(-1)^J\cdot([D_S]^{-1}[\bar{K}]) \qquad (19)$$

Hence, the required inversion is given by:

$$[K]^{-1}=\{\sum_{J=0}^{\infty}\{(-1)^J\cdot([D_S]^{-1}[\bar{K}])^J\}\}[D_S]^{-1} \qquad (20)$$

According to published accounts, it is normal practice to truncate this infinite series to a polynomial of at most five terms. Higher terms are frequently observed to introduce instability into the calculation, as the resulting matrix becomes less sparse.

An easily implemented preconditioner is the widely-known POLY(0), also called the Jacobi preconditioner. It consists of the elements of the leading diagonal alone, each term being the inverse of the corresponding value in [K]:

$$[K]^{-1}\cong diag(1/k_{ii}) \qquad i=1,n \qquad (21)$$

3.6 Incomplete polynomial preconditioner

In this study, as an extension of the idea of incomplete factorization, the latter was coupled to the polynomial series technique. With each truncation of the series at a given power, there was an associated *truncation of the number of columns* of the matrix being multiplied by itself. The series approximation is then constructed by taking only the positions nearest to the leading diagonal into account, as these are the elements whose values are most significant in the calculation of $[K]^{-1}$.

In view of the possibility of varying the spectrum of influence over the product of the matrix with itself, a new notation had to be created, to specify both the type of polynomial series and the number of diagonals (m_1) taken into account, commencing at the leading diagonal. Adding m_1 to the previous notation, it becomes POLY(n_T,m_1).

4 Applications

Shown below are some examples comparing the two preconditioning methods, in which the intrinsic parameters of each method have been varied. In other words, variations are given of the spectra of the influence of m_1 and m_2 in the Incomplete Cholesky Conjugate Gradient (ICCG) and the influence of the order of the number of powers considered in the (n_T) series and the number of polynomial (m_1) columns. It is explained that, when all the polynomial columns are considered, the polynomial (n_T, m) form is indicated, where m represents the length of the total matrix stored in the upper half of a square matrix.

The results presented here were obtained using a 64-Mbyte PENTIUM II 300 microcomputer and sequential programming.

4.1 Simply supported Square Plate

The model used here to verify the two methods originates from a FEM problem applied to a bidimensional plate structure with DKT elements, see [17, 18]. The geometrical characteristics of the material and the type of loading are shown in figure 5. Also shown are the results for the system without the use of preconditioners, considering the identity matrix as a preconditioner.

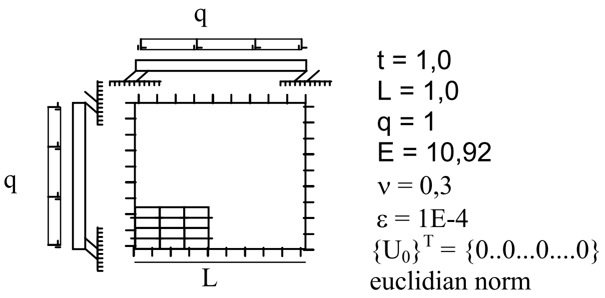


Figure 5 – Design of the plate, orientation and its parameters

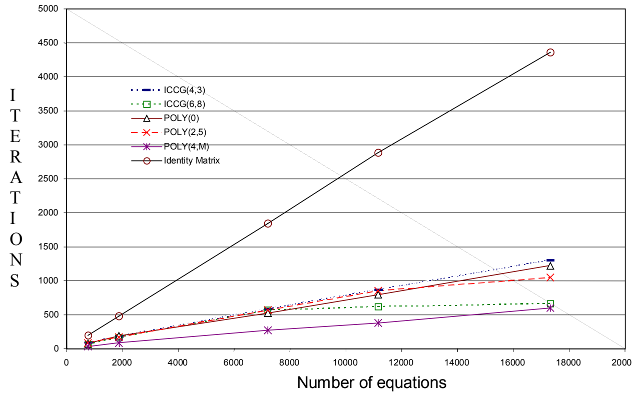


Figure 6 – Number of iterations *versus* Number of equations to the plate

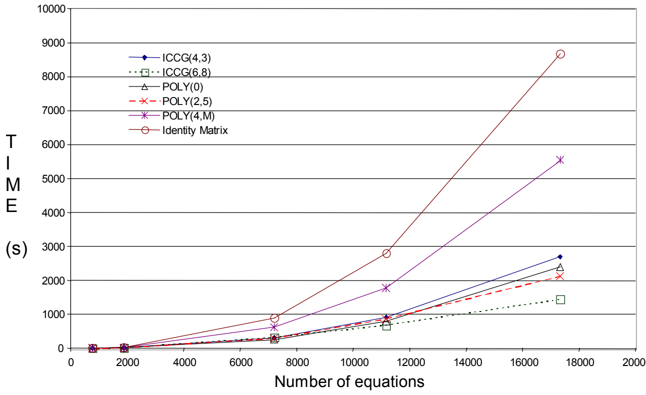
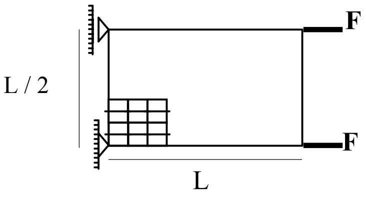


Figure 7 –Time *versus* Number of equations to the plate

4.2 Rectangular sheet with parallel sides

This example shows the results of convergence of a flat structure made of plate elements, both in terms of the number of iterations and of time. The finite element used to discretize the model is the CST, see [19, 20]. Thus, figure 8 illustrates the discretized structure, the orientation of the elements and the characteristics adopted.



$E = 1{,}0$
$\nu = 0{,}25$
$L = 4{,}0$
$F = t = 1{,}0$
$\varepsilon = 1E\text{-}4$
$\{U_0\}^T = \{0..0..0\}$
euclidian norm

Figure 8 – Design of the sheet, orientation and its parameters

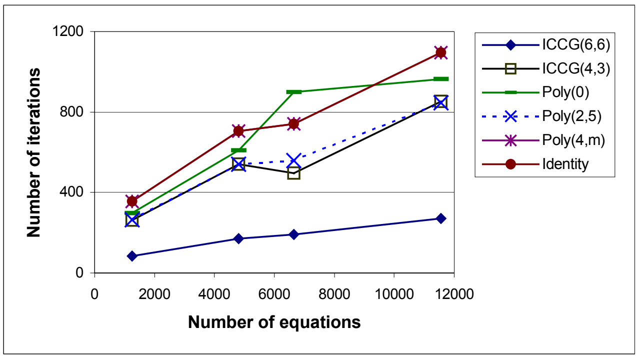


Figure 9 – Number of iterations *versus* Number of equations to the sheet

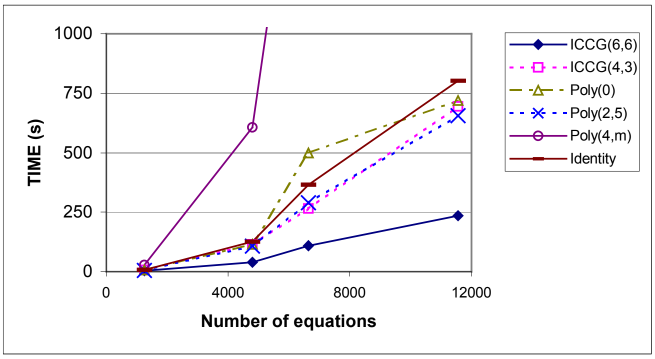


Figure 10 –Time *versus* Number of equations to the sheet

5 Conclusions

The value of using preconditioners to accelerate linear system solving by CG methods has been demonstrated in the results of tests on the model problems given in the previous section.

In terms of convergence time, the best overall result was given by the incomplete Cholesky decomposition (ICCG). Applied to the bending *plate* model, ICCG(6,8) and ICCG(6,6) converged in similar times. However, with the *plate*, ICCG(6,8) led to a non-positive definite matrix, while ICCG(6,6) did not and thus became the method of choice. Comparing the ICCG parameters employed here with those commonly used in the literature, ICCG(3,3) and (3,0), the latter were found to be less efficient, while having the advantage of easier implementation.

The incomplete polynomial proved to be an adequate preconditioner when coupled to the CG method for the analysis of structures. The conventional polynomial generates a more accurate value for the inverse of [K], but requires a large number of floating-point operations, whereas the incomplete polynomial method takes more iterations to converge, but gains in terms of time, because it gives emphasis on near leading diagonal values, which are in fact more relevant in the calculation of the inverse. Thus, for systems in which this diagonal dominates, the incomplete polynomial is efficient; in particular, POLY(2,5) was considered most successful in tests on the bending *plate* and plate intension models. POLY(0) did not prove so efficient in use, but its implementation was simpler.

The final remarks concern the parallel processing of structure analysis by the above techniques, in distributed memory computers (multicomputers). It is already clear that the choice of preconditioning technique for a particular structure is a hard one, and it must get significantly more complex when multicomputing is involved. Even so, certain advantages can be expected in using these techniques to solve linear systems in a parallel architecture. Thus, the generalized incomplete Cholesky decomposition is more efficient than the conventional method, in terms of number of iterations, while the incomplete polynomial method necessitates carrying out a matrix-vector multiplication, an operation well suited to parallel computing, with the added advantage that the number of multiplications can be truncated without reducing the effectiveness. Concluding, we believe the innovations offered by these techniques, when implemented in multicomputers, will play a leading role in linear system solving by FEM.

6 Acknowledgments

The authors wish to express their gratitude to FAPESP (São Paulo State Research Aid Foundation), which financed this research and without whose help the work could not have been done.

7 Bibliography

[1] A. K Noor. New Computing systems and future high-performance computing environment and their impact on structural analysis and design. *Computers & Structures,* 64, n. 1-4, p. 1-30 (1997).

[2] M. N Rezende. *Parallel Processing in structural analysis. In* São Carlos. 112p. PhD thesis – São Carlos, São Paulo University, 1995, (In Portuguese).

[3] V. S Almeida. *An adaptation of the MEF to the analysis in multicomputers: application in some structural models.* In São Carlos. 126p. Graduation work – Engineering school of São Carlos, São Paulo University, 1999, (In Portuguese).

[4] J. A Cuminato, M. Jr Meneguette. *Discretization of partial differential equation: Finite Differential Techniques*. Institute of Science and Mathematics of São Carlos, São Paulo University, 1998, (In Portuguese).

[6] K. H. Law. A parallel finite element solution method. *Computers & Structures,*. v. 23, n. 6, p. 845-858 (1986).

[8] J. A Meijerink and H. van der Vorst. An iterative solution method for linear systems of wich the

coefficient matrixis a symmetric M-matrix. *Mat. Comp.*, v.31 (137), p.148-162 (1977).

[10] L. A. Jr Schmt e Y. C. Lai. Structural optimization based on preconditioned conjugate gradient analysis methods. International Journal for numerical methods in engineering, v. 37, p. 943-964 (1994).

[12] P. F Dubois, A. Greenbaum e G. H. Rodrigue. Aproximating the Inverse of a Matrix for Use in Iterative Algorithms on Vector Processors. *Computing*, v.22, p. 257-268 (1977).

[14] D. G. Luenberger. *Optimization by vector space methods.* John Wiley, New York (1969).

[16] K. J Bathe. *Finite element procedures in engineering analysis.* Englewood Cliffs, Prentice-Hall (1982).

[18] C Jeyachandrabose, J. Kirkhope and C. R. Babu. An alternative explicit formulation for the DKT plate-bending element. *International Journal for Numerical Methods in Engineering*, v. 21, p. 1289-1293 (1985).

[20] M. J Turner, R. W Clough and H. C. Martin. Stiffness and Deflection Analysis of Complex Structures. *Journal of the aeronautical sciences*, v.23, n.9, p. 805-823 (1956).

[5] A. del Grosso and G. Righetti. Finite element techniques and artificial intelligence on parallel machines. *Computers & Structures*, v. 30, n.04, p. 999-1007 (1988).

[7] S. Bitzarakis, M. Papadrakakis and A. Kotsopulos. Parallel solution techniques in computational structural mechanics. *Comput. Methods Appl. Mech. Engr.*, v. 148, p. 75-104 (1997).

[9] M. Cimerman. Resolution of Linear Systems using Preconditioners Iterative Methods – *application in structural engineering problems.* São Paulo. 111p. Graduation work – São Paulo University, 1996, (In Portuguese).

[11] T. A Manteuffel. An incomplete factorization technique for positive definitive linear systems. *Mat. Comp.*, v.34 (150), p.473-497 (1980).

[13] O. G. Johnson, C. A. Micchelli and G. Paul. Polynomial preconditioners for conjugate gradient calculations. *SIAM J. Num. Anal.*, v.20, p. 362-376 (1983).

[15] G. H Golub and Van C. F. LOAN. *Matrix Computation.* Johns Hopkins Press, Baltimore. (1984).

[17] J. L Batoz K. J.Bathe and L. W. HO. A study of three–node triangular plate bending elements. *International Journal for numerical methods in engineering*, v. 15, p. 1771-1812 (1980).

[19] W. Savassi. Introdution to the FEM*: linear structural analysis*. São Carlos. Engineering school of São Carlos, São Paulo University, 1996, (In Portuguese).