Contemporary Mathematics
Volume 180, 1994

Cascadic Conjugate Gradient Methods for Elliptic Partial Differential Equations: Algorithm and Numerical Results

PETER DEUFLHARD

ABSTRACT. Cascadic conjugate gradient methods for the numerical solution of elliptic partial differential equations consist of Galerkin finite element methods as an outer iteration and (possibly preconditioned) conjugate gradient methods as an inner iteration. Both iterations are known to minimize the energy norm of the arising iteration errors. The present paper derives a unified framework in which to study the relative merits of different preconditioners versus the case of no preconditioning. Surprisingly, in the numerical experiments the cascadic conjugate gradient method *without any preconditioning* (to be called CCG method) turns out to be not only simplest but also fastest. It appears that the cascade principle in itself already realizes some kind of preconditioning. A theoretical explanation of the observed iteration pattern will be given elsewhere.

Introduction

This paper deals with *cascadic preconditioned conjugate gradient methods* — hereinafter called CPCG methods — for the solution of general elliptic boundary value problems for partial differential equations. Any such method is based on the so-called *cascade principle* which involves the cascade-like numerical solution of a sequence of linear systems of equations associated with a sequence of finite element spaces on successively finer grids. In this setting, the *coarse* grid linear system (up to moderate size) is assumed to be solved *directly* — say, by a (sparse) elimination technique. *Finer* grid systems are solved iteratively by *preconditioned conjugate methods* — hereinafter called PCG methods. Starting values for the PCG iteration on a given discretization level are just the (approximate) finite element solutions of the previous level. The successive finite element spaces are constructed *adaptively* based on local energy error estimators. Within each discretization level the PCG termination criterion aims at keeping the iteration error below the expected discretization error.

1991 *Mathematics Subject Classification.* Primary 65F10, 65N30.

This paper is in final form and no version of it will be submitted for publication elsewhere.

In former realizations of this concept [9, 5, 10, 12, 13], the whole iteration control mechanism was based on some cheap but not very satisfactory *energy error norm approximation*, which led to a close coupling between the local error estimators and the PCG termination criterion. That approximation has been recently replaced by another more satisfactory one, which is also cheap – compare [7] or the more thorough discussion in [8]. In Section 1, the new energy error control is applied to the (nested) CPCG iteration as a whole. The sequence of Galerkin approximations on successively finer grids is interpreted as an *outer iteration*, which minimizes the energy error norms over a sequence of finite element spaces. Arbitrary PCG iterations, which minimize the energy error norms over a sequence of Krylov spaces, are interpreted as *inner iterations*. An efficient strategy for the matching of inner versus outer energy error norms is developed. In Section 2, the proposed CPCG method is illustrated by numerical experiments on the Laplace equation. Comparison runs with the hierarchical basis preconditioner due to YSERENTANT [15], the multilevel preconditioner due to XU [14], BRAMBLE, PASCIAK and XU [6], and no preconditioning are presented. Two different CPCG modes are exemplified, an *adaptive* mode — including an adaptive mesh refinement strategy in the spirit of [1] based on the edge oriented error estimator of [9] — and a *uniform* mode. A convergence analysis explaining the surprising numerical findings of Section 2 will be given in a forthcoming paper.

1. Energy Error Control in Cascadic Preconditioned CG Methods

Consider an elliptic PDE problem given in the weak formulation

$$a(u,v) = \langle f, v\rangle, \qquad v \in H, \tag{1}$$

wherein H is the appropriate Hilbert space, $u \in H$ is the solution to be computed, $a(\cdot,\cdot)$ is a symmetric H-elliptic bilinear form with

$$\|\cdot\|_A = a(\cdot,\cdot)^{\frac{1}{2}} \tag{2}$$

the associated energy norm, and $\langle\cdot,\cdot\rangle$ the L_2 inner product. Consider further a Galerkin method for a sequence of nested finite element spaces $\mathcal{S}_0 \subset \mathcal{S}_1 \subset \ldots \mathcal{S}_i \subset H$. This generates a sequence of linear systems

$$A_j u_j = b_j, \qquad j = 0, 1, \ldots, i. \tag{3}$$

All matrices A_j are symmetric positive definite so that PCG methods can be applied. Let $n_j = \dim \mathcal{S}_j$. For ease of writing we will not distinguish between the solutions u_j of the Galerkin equations within subspace $\mathcal{S}_j$ and the exact solutions u_j of the corresponding systems of linear equations (3). The meaning will be clear from the context.

1.1. Galerkin method as outer iteration. At discretization level j, let

$$\epsilon_j = \|u - u_j\|_A^2, \qquad \delta_j = \|u_{j+1} - u_j\|_A^2$$

denote the various discretization error norms. Then, for *nested* spaces S_j, the *orthogonality relation*

$$\epsilon_{j+1} = \epsilon_j - \delta_j, \tag{4}$$

is well-known to hold. For an appropriate sequence of subspaces of the Hilbert space H we have convergence $u_j \to u$ and therefore $\epsilon_\infty = 0$, so that the recursion (4) can be solved to yield

$$\epsilon_j = \sum_{l=j}^{\infty} \delta_l, \qquad j = 0, 1, \ldots . \tag{5}$$

As for the convergence of the FE method, i.e. the outer iteration, we will naturally require the theoretical termination criterion

$$\epsilon_i \leq TOL\, \epsilon_0, \tag{6}$$

with some error tolerance parameter TOL to be prescribed by the user and some index i to count for the finest actually computed discretization level. Upon proceeding as in [7, 8], we will try to replace the not implementable termination criterion (6) by a *sufficient and implementable* termination criterion. As for the above right-hand side, we will replace ϵ_0 by the lower bound

$$\epsilon_0^{(i)} = \|u_{i+1} - u_0\|_A^2 = \epsilon_0 - \epsilon_{i+1} = \sum_{j=0}^{i} \delta_j \leq \epsilon_0. \tag{7}$$

As for the left-hand side of (6), let Θ denote a contraction factor understood to satisfy

$$\epsilon_{j+1} \leq \Theta \epsilon_j, \qquad \Theta < 1 \qquad j > j_0$$

for some threshold index j_0. Then equation (4) implies that

$$\epsilon_j \leq \bar{\epsilon}_j = \frac{\delta_j}{1-\Theta}. \tag{8}$$

For illustration purposes, consider the Laplace equation on some polygonal domain Ω in d-dimensional space with $d = 2$ or $d = 3$. This means that H is now some space H_0^1 wherein the subscript 0 indicates the fact that we assume Dirichlet boundary conditions on a sufficient part of the boundary $\partial\Omega$. In this case we know that *uniform* mesh refinement and *linear* finite elements in the regular case lead to

$$\Theta \doteq \frac{1}{4}, \qquad \bar{\epsilon}_j \doteq \frac{4}{3}\delta_j. \tag{9}$$

In the case of *adaptive* meshes — assuming *energy error equidistribution* and once more *linear* finite elements — we expect a comparable estimated contraction factor

$$\Theta \doteq \bar{\Theta}_j = \left(\frac{n_{j-1}}{n_j}\right)^{2/d}, \qquad j > j_0, \tag{10}$$

on the basis of theoretical results of [2]. The latter choice is used in the numerical experiments below. Further improvements of this factor should be possible in close combination with the adaptive mesh refinement strategy based on local extrapolation [1]. Such a device is presently under investigation.

Summarizing, we end up with the *sufficient* condition

$$\bar{\epsilon}_i \leq TOL\, \epsilon_0^{(i)} \tag{11}$$

for termination at finite element level i. The implementation of this criterion requires the iterative quantities δ_j, which — as we will see below — can be obtained cheaply from the PCG iteration.

1.2. PCG method as inner iteration. In previous versions of cascade type algorithms — such as [9, 5, 10] — the criterion (11) has been used in connection with the edge-oriented discretization error estimator due to [9]. In what follows, an alternative technique based on considerations from [7, 8] will be worked out.

With the outer iterates u_j from the Galerkin method, we now need two indices $u_{j,k}$ for the PCG iterates. On levels $j = 1, 2, \dots, i$, the iteration index k formally runs within $k = 0, \dots, n_j$. On the coarse grid level $j = 0$, direct linear equation solving supplies some u_0 assumed to be exact. On finer levels $j > 0$, the cascade principle realizes

$$u_{j,0} = u_{j-1}, \qquad u_j = u_{j,n} \quad \text{for } n = n_j.$$

It is known from [7, 8] that the iterative energy error contributions

$$\delta_{j,k} = \|u_{j,k+1} - u_{j,k}\|_A^2$$

are cheaply available on levels $j > 0$. Moreover, since

$$\|u_j - u_{j-1}\|_A^2 = \|u_{j,n} - u_{j,0}\|_A^2 = \sum_{k=0}^{n_j-1} \|u_{j,k+1} - u_{j,k}\|_A^2,$$

we can express the iterative discretization errors as

$$\delta_{j-1} = \|u_j - u_{j-1}\|_A^2 = \sum_{k=0}^{n_j-1} \delta_{j,k}. \tag{12}$$

In words: *The (exact) PCG iteration on discretization level j supplies the energy norm of the iterative discretization error of the preceding level $j-1$.*

In actual computation, things are slightly more complicated, since instead of the above exact PCG iterates $u_{j,k}$ we have *perturbed* iterates $\tilde{u}_{j,k}$ obtained from truncated PCG iterations, which yield perturbed Galerkin approximations $\tilde{u}_j$. As in the exact case, we start from the direct solution $\tilde{u}_0 = u_0$. On finer levels $j > 0$, however, we continue according to

$$\tilde{u}_{j,0} = \tilde{u}_{j-1}, \qquad \tilde{u}_j = \tilde{u}_{j,m+1} \quad \text{for some truncation index } m = m_j. \tag{13}$$

With the cheaply available quantities

$$\tilde{\delta}_{j,k} = \|\tilde{u}_{j,k+1} - \tilde{u}_{j,k}\|_A^2$$

instead of the $\delta_{j,k}$ we obtain the analog of result (12) now in the form

$$\tilde{\delta}_{j-1} = \|\tilde{u}_j - \tilde{u}_{j-1}\|_A^2 = \sum_{k=0}^{m_j} \tilde{\delta}_{j,k}. \tag{14}$$

The associated discretization errors

$$\tilde{\epsilon}_j = \|u - \tilde{u}_j\|_A^2 \tag{15}$$

can be seen to satisfy

$$\tilde{\epsilon}_j = \|u - \tilde{u}_j\|_A^2 \geq \min_{v \in S_j} \|u - v\|_A^2 = \|u - u_j\|_A^2 = \epsilon_j. \tag{16}$$

Throughout this paper the FE spaces are (technically) assumed not to depend on the sequence of truncation indices of the PCG iteration — an assumption which is only realistic if utmost care is taken in the realization of the whole scheme including the adaptive mesh refinement strategy. As in (4) the orthogonality relation

$$\tilde{\epsilon}_{j+1} = \tilde{\epsilon}_j - \tilde{\delta}_j \tag{17}$$

holds — this time not as a consequence of the Galerkin minimization property, but due to the cascade property (13) and the orthogonality of the PCG iterative corrections within the Krylov spaces also in the perturbed case. As in the exact case, we want to replace the unavailable term $\tilde{\epsilon}_0$ by computationally available terms of the form

$$\tilde{\epsilon}_0^{(i)} = \sum_{j=0}^{i} \tilde{\delta}_j. \tag{18}$$

With $\tilde{\epsilon}_0 = \epsilon_0$ — due to the direct solution on the coarse grid — and (16) we obtain

$$\tilde{\epsilon}_0^{(i)} = \tilde{\epsilon}_0 - \tilde{\epsilon}_{i+1} = \epsilon_0 - \tilde{\epsilon}_{i+1} \leq \epsilon_0 - \epsilon_{i+1} = \epsilon_0^{(i)}.$$

This means that — in view of sufficiency — the right-hand side term $\epsilon_0^{(i)}$ in the termination criterion (11) can be replaced by $\tilde{\epsilon}_0^{(i)}$.

We now turn to the approximation of the left-hand side of (11), which means that we have to consider an approximation of the quantity ϵ_j at the recursive levels j. Unfortunately, this quantity cannot be bounded on either side by its associated estimate $\tilde{\epsilon}_j$, since the starting values $\tilde{u}_{j,0}$ for the perturbed PCG iteration differ from the exact starting values $u_{j,0}$ on the finer grids $j > 0$. Moreover, at level j only the discretization error $\tilde{\delta}_{j-1}$ for the previous level is actually available. In this situation, we recur to (10) and (15) to obtain

$$\tilde{\epsilon}_j \doteq \bar{\Theta}_j \tilde{\epsilon}_{j-1}, \qquad \tilde{\epsilon}_{j-1} \leq \frac{\tilde{\delta}_{j-1}}{1 - \bar{\Theta}_j}. \tag{19}$$

Accordingly, we approximate the termination criterion (11) by the criterion

$$\hat{\epsilon}_i = \frac{\bar{\Theta}_i \tilde{\delta}_{i-1}}{1 - \bar{\Theta}_i} \leq TOL\, \tilde{\epsilon}_0^{(i)} \tag{20}$$

to be satisfied at final level i. This is now the implementable *termination criterion for the outer iteration* to be used for the determination of the actually needed discretization level i according to the user prescribed relative accuracy TOL. (The terms $\hat{\epsilon}_j$ will be returned to the user as *recursive discretization error estimate* on level j.)

We are still left with the decision of how to control the inner PCG iterations in such a way that the iterative discretization error estimates $\hat{\epsilon}_j$ are sufficiently reliable. In view of [**7**, **8**] we will require a condition of the kind

$$\text{(21)} \qquad \frac{\tilde{\delta}_{j,m}}{1-\bar{\Theta}_{j,m}} \leq \hat{\rho}_j \bar{\Theta}_j \tilde{\delta}_{j-1} \qquad \text{for some truncation index } m = m_j .$$

Herein the estimate of the contraction factors $\bar{\Theta}_{j,m}$ may cause difficulties in actual computation and therefore require some additional heuristics. The safety factors $\hat{\rho}_j$ need to be chosen as internal default parameters. In former realizations [**9**, **5**], the choice $\hat{\rho}_j = \text{const}$ had been made, which led to a condition most stringent on the *final* level i. Recall, however, the global error relation

$$\text{(22)} \qquad \tilde{\epsilon}_i = \|u - \tilde{u}_i\|_A^2 = \epsilon_0 - \sum_{j=1}^{i} \sum_{k=0}^{m_j} \tilde{\delta}_{j,k} .$$

This relation seems to indicate that the final level restriction from (21) should already be observed on coarser levels to avoid unnecessary more costly iterations on the finer levels. Therefore we suggest to replace (21) by

$$\text{(23)} \qquad \frac{\tilde{\delta}_{j,m}}{1-\bar{\Theta}_{j,m}} \leq \hat{\rho}\, TOL\, \tilde{\epsilon}_0^{(j)} \qquad \text{for some truncation index } m = m_j .$$

This criterion is more stringent on *coarser* levels. Numerical experiments (with $\hat{\rho} = \frac{1}{16}$ throughout) strongly confirmed the expectation that criterion (23) saves costly iterations on finer levels compared to the criterion (21). This is now the desired implementable *termination criterion for the inner iteration.*

With the two termination criteria (20) for the Galerkin outer iteration and (23) for the inner PCG iteration, the whole CPCG iteration error control is now complete. Note that it applies independent of any special choice of preconditioner or even without any preconditioner.

For the convenience of the reader, we now summarize the whole CCG iteration (without preconditioning) in the form of a *pseudocode* — see Table 1. The notation for the CG iteration follows the one given in [**7**].

TABLE 1. **Pseudo-code: cascadic conjugate gradient method**

PROCEDURE CCG (OUTER ITERATION):

```
input $TOL$                                          // prescribed tolerance
$j = 0$
while $(j \leq maxLevel)$
{
        assemble linear system A,b ;
        CG(A,u,b);                                   // conjugate gradient iteration
        calculate $\hat{\epsilon}_j, \tilde{\epsilon}_0^{(j)}$
        if $\left(\hat{\epsilon}_j \leq TOL\ \tilde{\epsilon}_0^{(j)}\right)$ end;       // termination criterion (20)
        refine mesh;
        $j = j + 1$;
}
```

PROCEDURE CG(A,x,b) (INNER ITERATION):

```
input $j, TOL, \hat{\rho}, \tilde{\epsilon}_0^{(j)}$
$p_0 = r_0 = b - Ax_0$,
$\sigma_0 = (r_0, r_0)$,
$k = 0$
while $(k \leq maxIter)$
{
        $\alpha_k = \frac{\langle Ap_k, p_k \rangle}{\sigma_k}$,
        $x_{k+1} = x_k + \frac{1}{\alpha_k} p_k$,
        $\tilde{\delta}_{j,k} = \sigma_k / \alpha_k$;
        calculate $\bar{\Theta}_{j,k}$, $\tilde{\delta}_{j-1}$, ;
        if $\left(\frac{\tilde{\delta}_{j,k}}{1 - \bar{\Theta}_{j,k}} \leq \hat{\rho} TOL \tilde{\epsilon}_0^{(j)}\right)$ return;     // termination criterion (23)
        $r_{k+1} = r_k - \frac{1}{\alpha_k} Ap_k$,
        $\sigma_{k+1} = \langle r_{k+1}, r_{k+1} \rangle$,   $\beta_{k+1} = \frac{\sigma_{k+1}}{\sigma_k}$,
        $p_{k+1} = r_{k+1} + \beta_{k+1} p_k$;
        $k = k + 1$;
}
```

2. Numerical Experiments

Up to now, numerical tests were only made for the *Laplace equation with linear finite elements*, both in 2–D and in 3–D. The picture in 3–D appeared to be essentially the same as in 2–D (though slightly less reliable in the adaptive case). As computing times and storage requirements for the 3–D test runs blew up considerably, the subsequent illustration is restricted to the 2–D case. Out of the three test examples given in [7] only one will be considered here for reasons of restricted space.

Peak Problem. Given the PDE

$$-\Delta u = f \,, \tag{24}$$

Dirichlet boundary conditions are imposed such that

$$u = (x+1)(x-1)(y+1)(y-1)e^{-100(x^2+y^2)}$$

is the solution.

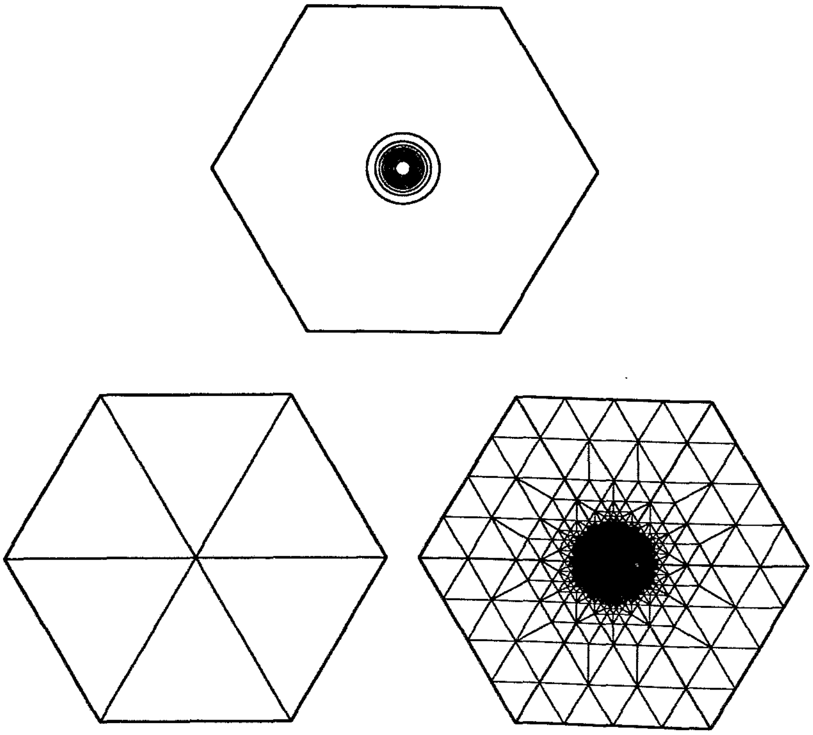

FIGURE 1. Solution of peak problem on level $j = 7$, $n = 1975$ nodes (adaptive mode) and grids on levels $j = 0$ and $j = 7$.

The comparative performance of the following three variants of the CPCG algorithm are presented:

— CPCG–HB: algorithm with *hierarchical basis* preconditioner [15]

— CPCG–BPX: algorithm with *multilevel* BPX preconditioner [14] , [6]

— CCG: algorithm *without any* preconditioner

In the simple case of the Laplace equation, the HB preconditioner is known to give rise to an $O(j^2)$-bound on the condition number in 2–D — see [15], whereas the BPX preconditioner is known to lead to $O(1)$ independent of the space dimension — see [11]. In 2–D, the expected numerical efficiency will be nearly the same for both preconditioners. Based on the subtle condition number estimates of Xu [14], the expectation for the algorithm without any preconditioning would be that it might be asymptotically disastrous in rather uniform grids (which exhibit a *geometric* increase of the number of nodes) and not too bad in highly non–uniform grids (with an *arithmetic* increase of the number of nodes).

In actual computation, the explicit formulation of the FE problem (numerical quadrature for evaluation of inner products), will dominate the whole computing time. In order to make the differences between the three algorithmic variants visible, the subsequent comparison runs will mostly quote the pure *iteration times* and the *number of required iterations.* Since the solutions of all examples above are explicitly known, the directly computed iterative errors and the errors estimated from the CPCG iterations could be compared: the discrepancies were marginal on lower levels and tolerable (in most cases) on the finest levels. For this reason, only the estimated accuracies are documented here — which is the realistic case. The accuracy is measured in terms of the improvement factor from initial to final energy error norms; since, in the best case, we can expect an iterative improvement of one bit of accuracy for the Laplace equation with linear finite elements, all Figures below will use binary digits. Numerical experiments were run on a SUN Sparc Workstation 10/41 using the g++ C compiler.

2.1. Adaptive Mode. In this section we will arrange comparative results for the three algorithmic variants running in the adaptive mode, which means that a *refinement strategy* is applied to generate a sequence of *possibly highly non–uniform meshes.* Any such mesh refinement strategy will naturally aim at equidistributing the energy error. In the earlier version [9] of the cascade principle a mean value strategy due to BANK [3] has been used. This strategy, however, sometimes produced unsatisfactory meshes in critical examples. Therefore, the more advanced versions [10] and [5] realized a mesh refinement technique in the spirit of BABUŠKA and RHEINBOLDT [1]. It is based on *local extrapolation* of energy error contributions from the edges as obtained by the *edge oriented discretization error indicator* due to [9]. The subsequent numerical experiments are run with a slightly modified heuristic – for details see [7].

In Fig. 2 the comparative results for the three codes are represented graphically in terms of *iteration computing times*. The surprise is that the code CCG without any preconditioning is fastest. The comparison in terms of *number of iterations* is given in Fig. 3. Obviously, the asymptotic behavior of both CCG and CPCG–BPX is the same. Between start and end all three variants show some iteration number bump, coming from the global accuracy requirement (23), which is more stringent on coarser levels. The bump is largest for CCG and smallest for BPX. The HB variant ranges in between. The different picture in terms of computing times is explained by the fact that each CG iteration with BPX preconditioning (even in a rather efficient implementation — see e.g. [4]) costs a rough factor of 3 more than each pure CG iteration.

Remark 1. It should be mentioned that the above Fig. 3 does *not* contradict Fig. 10, p. 3198 in [5], wherein the effect of BPX preconditioning versus no preconditioning has been exemplified as well. There, however, the iteration has been continued *far below the discretization error*. In this setting, the number of iterations without preconditioning drifted off far above the number of iterations with BPX preconditioning.

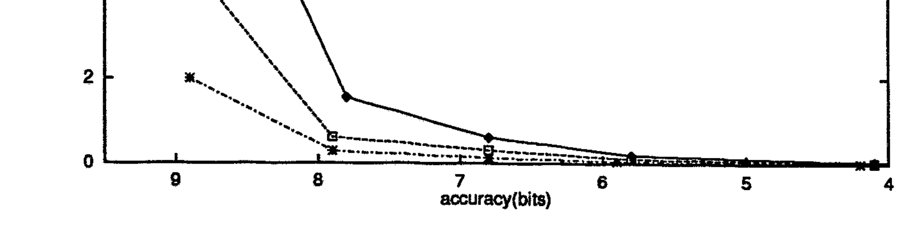

FIGURE 2. Comparative iteration times, adaptive mode.

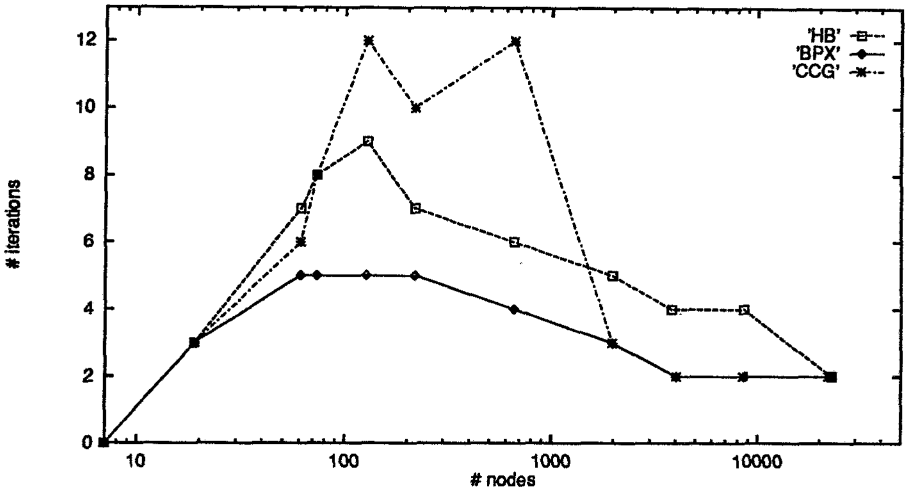

FIGURE 3. Comparative iteration number patterns, adaptive mode.

Remark 2. For the sake of completeness, we should mention that part of the above orthogonality relations require the successive FE spaces to be *nested* — a condition, which is not satisfied whenever so-called *green edges* (cf. [3]) are dissolved from one level to the next. We have corrected the above formulas so affected in terms of the energy error differences thus introduced. However, the effect was so minor that this modification was ultimately omitted.

2.2. Uniform Mode. We now illustrate the three algorithmic variants of the CPCG method in the non-adaptive or uniform mode. In this mode, *uniform mesh refinement* is performed without making actual use of any discretization error estimator or indicator. The associated considerable amount of overall computing time is therefore saved. We exemplify this mode only for the peak problem above, which would certainly require a highly non-uniform mesh (compare Fig. 1).

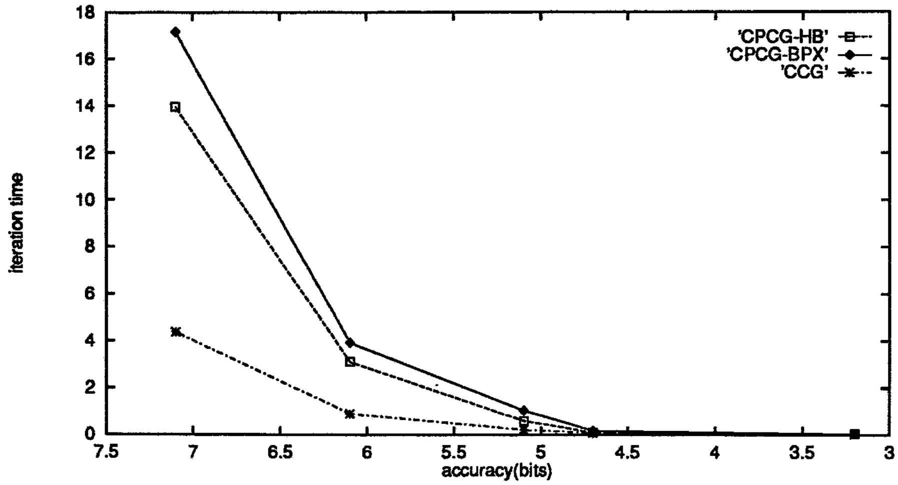

FIGURE 4. Comparative iteration times, uniform mode.

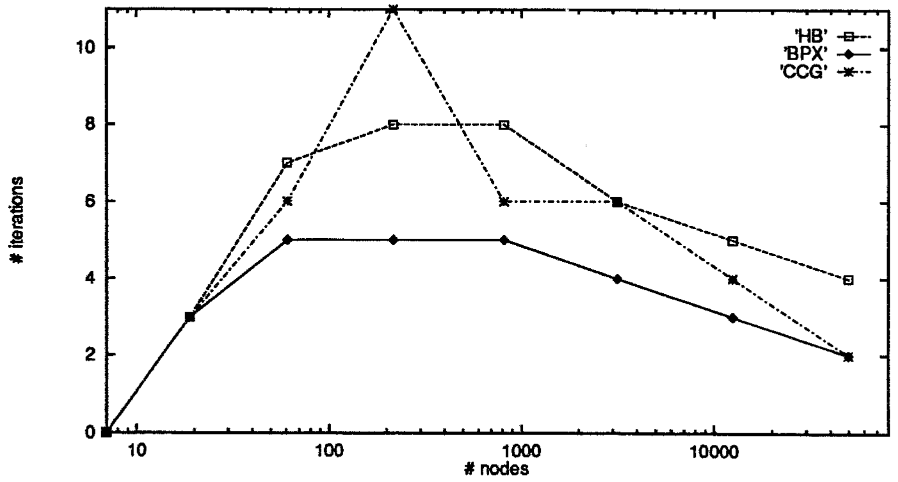

FIGURE 5. Comparative iteration number patterns, uniform mode.

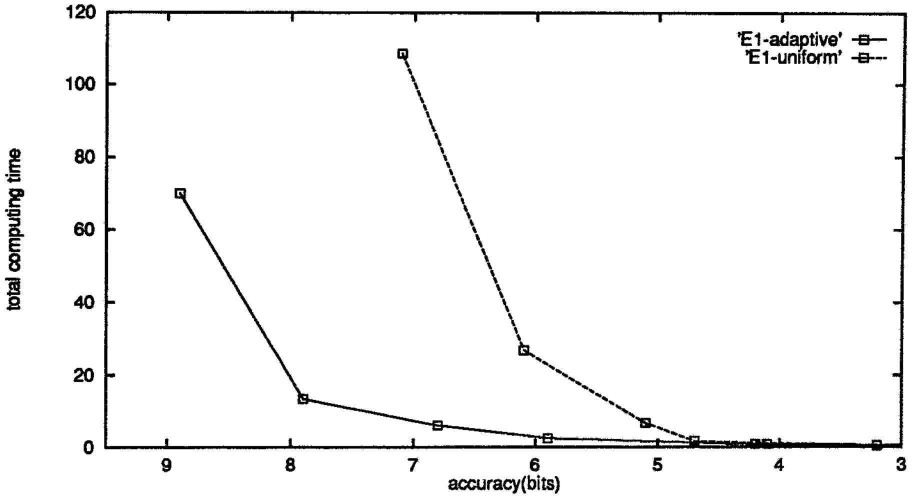

FIGURE 6. Uniform versus adaptive mode, peak problem.

Fig. 4 shows the comparative iteration times and Fig. 5 the comparative number of iterations. The effects in the uniform mode are obviously the same as in the adaptive mode before. Once more, the CCG variant without preconditioning is clearly superior to the two preconditioners HB and BPX. Note that the CCG variant is much simpler to implement and does not need any analytic pre–investigations, which typically involve a high technical level of sophistication.

The above experiments should not leave the impression that adaptivity does not pay off. For this reason, Fig. 6 compares the total amount of computing time (full Galerkin approximation) for the adaptive and the uniform CCG mode as a function of the achieved discretization error accuracy. As can be seen, storage and computing time limitations lead to rather stringent accuracy limitations for the uniform mode. This factor is even more limiting in 3–D!

Conclusion

The present paper derives a simple but efficient strategy to control the discretization errors of the Galerkin FEM in combination with the iteration errors of the PCG method in terms of the energy norm. The relative merits of different preconditioners versus the case of no preconditioning have been compared. It appeared that the cascadic conjugate gradient method without any preconditioning (called CCG herein) was not only simplest but also fastest compared to the HB and BPX preconditioned case. In the 2–D comparison runs, HB was second and BPX was third, whereas in the 3–D runs (not documented herein) the two preconditioners interchanged their role – as expected from theory. The asymptotic behavior of the CCG method turned out to be the same as the one of the CPCG method with BPX preconditioning. Moreover, the effects were the same both for the adaptive and the non–adaptive mode — and therefore independent of any possible energy error equidistribution.

Summarizing, the numerical results seem to indicate that the cascade principle in itself already realizes some kind of preconditioning. A theoretical study of this feature is in progress.

Acknowledgements. The author gratefully acknowledges invaluable computational assistance from R. Roitzsch and B. Erdmann and helpful comments and careful reading of the manuscript by R. Beck. He also wishes to thank E. C. Körnig for her quick and patient help with the revision of the manuscript.

REFERENCES

1. I. Babuška, W.C. Rheinboldt: *Estimates for Adaptive Finite Element Computations.* SIAM J. Numer. Anal. **15** (1978), pp. 736–754.
2. I. Babuška, R.B. Kellogg, J. Pitkaränta: *Direct and Inverse Error Estimates for Finite Elements with Mesh Refinements.* Numer. Math. **33** (1979), pp. 447–471.
3. R.E. Bank: *A–posteriori Error Estimates, Adaptive Local Mesh Refinement and Multigrid Iteration.* In: W. Hackbusch, U. Trottenberg (eds.): Multigrid Methods II. Springer Verlag, Berlin, Heidelberg, New York: Lect. Notes Math. Vol. **1228** (1986), pp. 7–23.
4. F. Bornemann: *An Adaptive Multilevel Approach to Parabolic Equations in Two Space Dimensions.* Dissertation, FU Berlin (1991).
5. F. Bornemann, B. Erdmann, R. Kornhuber: *Adaptive Multilevel–Methods in Three Space Dimensions.* Int. J. Numer. Methods Eng. **36** (1993), pp. 3187–3203.
6. J.H. Bramble, J.E. Pasciak, J. Xu: *Parallel Multilevel Preconditioners.* Math. Comp.**55** (1990), pp. 1–22.
7. P. Deuflhard: *Cascadic Conjugate Gradient Methods for Elliptic Differential Equations I. Algorithm and Numerical Results.* Konrad–Zuse–Zentrum Berlin (ZIB), Preprint SC 93–23 (1993).
8. P. Deuflhard: *On Error Control in Preconditioned CG Iterations.* In preparation.
9. P. Deuflhard, P. Leinen, H. Yserentant: *Concepts of an Adaptive Hierarchical Finite Element Code.* IMPACT **1** (1989), pp. 3–35.
10. B. Erdmann, R. Roitzsch, F. Bornemann: *KASKADE numerical experiments.* Technical Report TR 91–1, Konrad–Zuse–Zentrum Berlin (ZIB), (1991).
11. P. Oswald: *On Discrete Norm Estimates Related to Multilevel Preconditioners in the Finite Element Method.* In: Proceedings of the International Conference on the Constructive Theory of Functions, Varna 1991 (to appear).
12. R. Roitzsch: KASKADE *User's Manual.* Konrad–Zuse–Zentrum Berlin (ZIB), Technical Report Report TR 89–4 (1989)
13. R. Roitzsch: *KASKADE programmer's manual.* Konrad–Zuse–Zentrum Berlin (ZIB), Technical Report TR89–5 (1989).
14. J. Xu: *Theory of Multilevel Methods.* Report No. AM 48, Department of Mathematics, Pennsylvania State University (1989).
15. H. Yserentant: *On the Multilevel Splitting of Finite Element Spaces.* Numer. Math. **49** (1986), pp. 379–412 .

KONRAD-ZUSE-ZENTRUM BERLIN, HEILBRONNER STR. 10, D-10711 BERLIN-WILMERSDORF, GERMANY

E-mail address: deuflhard@sc.zib-berlin.de