Cascadic Conjugate Gradient Methods for Elliptic Partial Differential Equations I. Algorithm and Numerical Results
Cascadic Conjugate Gradient Methods for Elliptic Partial Differential Equations I. 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 outer iteration and (possibly preconditioned) conjugate gradient methods as inner iteration. Both iterations are known to minimize the energy norm of the arising iteration errors. A simple but efficient strategy to control the discretization errors versus the PCG iteration errors in terms of energy error norms is derived and worked out in algorithmic detail. In a unified setting, the relative merits of different preconditioners versus the case of no preconditioning is compared. Surprisingly, it appears that the cascadic conjugate gradient method without any preconditioning is not only simplest but also fastest. The numerical results seem to indicate that the cascade principle in itself already realizes some kind of preconditioning. A theoretical explanation of these observations will be given in Part II of this paper.

Key words. conjugate gradient method, multilevel finite element methods, preconditioning

AMS(AMOS) subject classification. 65F10, 65N30
## Contents

**Introduction**  
1

1. Energy Error Control in Cascadic Preconditioned CG Methods  
   1.1 Termination Criteria for Preconditioned CG Methods  
   1.2 Matching of Inner and Outer Iteration

2. Numerical Experiments  
   2.1 Adaptive Mode  
   2.2 Uniform Mode

**Conclusion**  
23

**Acknowledgements**  
23

**References**  
24
Introduction

This paper deals with cascadic preconditioned conjugate gradient methods – to be 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 that has been presented earlier in [7]. The former concept and its algorithmic realization will be significantly simplified and, at the same time, made considerably more robust.

Cascade type algorithms work down a sequence of linear systems of equations associated with finite elements on successively finer grids. Their characterizing features are

- The coarse grid linear system, which is assumed to have only a small up to a moderate number of degrees of freedom, is solved directly — say, by a sparse elimination technique.

- Finer grid systems are solved iteratively by preconditioned conjugate methods — to be 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.

- 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.

In the former realization of this concept due to [7], the whole iteration control mechanism was based on cheap approximate energy error norms, which led to a close coupling between the local error estimators and the PCG termination criterion. In the course of further development — see [5, 8, 10, 11] — the unclear quality of this approximation motivated the use of rather stringent heuristic safety factors. On one hand, this increased the robustness in critical examples, but on the other hand, this decreased the efficiency in less critical examples.

In what follows, this kind of approximation of the energy error norm is replaced by some other more reliable approximation, which is cheap, too. In Section 1, the general scheme of the CPCG iteration is derived in a unified frame without specifying the preconditioner. Details are first explained in the simpler finite dimensional case of pure PCG iterations. In the second step, the infinite dimensional case of the CPCG iteration is treated. The sequence of Galerkin approximations on successively finer grids is interpreted as 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 iteration. 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. The good old Laplace equation is once more used as a work horse. Comparison runs with the hierarchical bases preconditioner due to Yserentant [14], the multilevel preconditioner due to Bramble, Pasciak and Xu [6] and even without any 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 [7] — and a uniform mode. A convergence theory explaining the surprising numerical findings of Section 2 will be given in Part II of this paper.

1. Energy Error Control in Cascadic Preconditioned CG Methods

In this section CPCG methods are derived in a unified framework. In order to fix notation, consider a PDE problem given in weak formulation

\[ a(u, v) = \langle f, v \rangle, \quad v \in H, \quad (1.1) \]

wherein \( H \) is the appropriate Hilbert space, \( u \in H \) is the solution to be computed, and \( a(\cdot, \cdot) \) is a symmetric positive definite bilinear form. Up to now, in most scientific and engineering applications, \( H \) is just replaced by one finite dimensional space of possibly high dimension. However, as is the firm belief of the present author, a Hilbert space should rather be modelled by a sequence of spaces with successively increasing dimension to capture its characterizing asymptotic properties correctly. In this spirit, we assume here that Galerkin methods for a sequence of nested finite element spaces \( S_0 \subset S_1 \subset \ldots \subset S_i \subset H \) are applied. They give rise to a sequence of linear systems

\[ A_j u_j = b_j, \quad j = 0, 1, \ldots, i \quad (1.2) \]

which each represents the Galerkin approximation on \( S_j \). Let \( u_j \) denote the associated exact solutions of the above equations. The matrices \( A_j \) are all symmetric positive definite so that PCG methods can be applied.

In order to simplify the presentation, PCG methods for a single system of equations are treated first. Afterwards their combination with Galerkin finite element methods is studied.
1.1 Termination Criteria for Preconditioned CG Methods

Consider a system of $n$ linear equations to be solved

$$Ax = b$$  \hfill (1.3)

with $A$ being a symmetric positive definite matrix. Let $B$ be a given symmetric positive definite preconditioner such that

$$B \sim A^{-1}.$$  \hfill (1.4)

With $x_0$ as starting guess of the solution $x$ a PCG method can be applied. We introduce the PCG iterates $x_k$, the residuals $r_k$, and the preconditioned residuals $\bar{r}_k$. With this notation, the PCG method can be written as follows:

$$r_0 = b - Ax_0, p_0 = \bar{r}_0 = Br_0,$$
$$\sigma_0 = \langle r_0, \bar{r}_0 \rangle,$$
$$k = 1, 2, \ldots :$$
$$\alpha_{k-1} = \frac{\langle p_{k-1}, Ap_{k-1} \rangle}{\sigma_{k-1}},$$
$$x_k = x_{k-1} + \frac{1}{\alpha_{k-1}} p_{k-1},$$
$$r_k = r_{k-1} - \frac{1}{\alpha_{k-1}} Ap_{k-1}, \quad \bar{r}_k = Br_k,$$
$$\sigma_k = \langle r_k, \bar{r}_k \rangle,$$
$$\beta_k = \frac{\sigma_k}{\sigma_{k-1}},$$
$$p_k = \bar{r}_k + \beta_k p_{k-1}.$$  \hfill (1.5)

The PCG convergence theory is naturally expressed in terms of the well–known energy norm

$$\| \cdot \|_A = (\cdot, \cdot)^A_A = \langle \cdot, A \cdot \rangle^A_A.$$  \hfill (1.6)

Let now $e_k$ denote the iterative error, which implies that $r_k = A e_k$. Then a natural termination criterion for any PCG iteration would be to require that, for some user prescribed parameter $\varepsilon_{\text{red}}$ and some iteration index $m$, the condition

$$\| e_m \|_A \leq \varepsilon_{\text{red}} \| e_0 \|_A,$$  \hfill (1.7)

holds. However, the direct computation of the iterative values $\| e_k \|_A$ is often prohibitive. Upon keeping in mind that

$$\sigma_k = (e_k, e_k)_{ABA} \sim (e_k, e_k)_A,$$  \hfill (1.8)
the above termination criterion might be approximated by
\[ \sigma_m \leq \varepsilon^2_{\text{red}} \sigma_0. \] (1.9)

Unfortunately, the replacement of the theoretical criterion (1.7) by the practical criterion (1.9) induces some uncertainty, which is described by the following rather elementary lemma.

**Lemma 1.1** Let \( \lambda_{\min} \) and \( \lambda_{\max} \) denote the minimum and maximum eigenvalue of the matrix \( AB \). Then the following estimates hold:
\[ \lambda_{\min}(e_k, e_k) \leq \sigma_k \leq \lambda_{\max}(e_k, e_k). \] (1.10)

**Proof.** With \( A, B \) both being symmetric positive definite matrices, the following decomposition is useful:
\[ \sigma_k = (e_k, e_k)_{ABA} = \left( A^{\frac{1}{2}} e_k, \hat{A} A^{\frac{1}{2}} e_k \right) \]
wherein \( \hat{A} = A^{\frac{1}{2}} B A^{\frac{1}{2}} \) is once more a positive definite symmetric matrix. Hence its eigenvalues \( \lambda \) are real and positive. Moreover, they are also the eigenvalues of the matrices \( AB \) or \( BA \). With \( \lambda_{\min} \) and \( \lambda_{\max} \) as defined above, the estimate of the lemma is directly obtained. \( \square \)

From this lemma, we obtain the relations
\[ \lambda_{\min}(e_0, e_0) \leq \sigma_0, \quad \sigma_m \leq \lambda_{\max}(e_m, e_m) \]

Hence, by virtue of the **sufficient** condition
\[ \sigma_m \leq \frac{1}{\kappa_2(AB)} \varepsilon^2_{\text{red}} \sigma_0 \quad \text{with} \quad \kappa_2(AB) = \frac{\lambda_{\max}}{\lambda_{\min}}, \]
we are able to guarantee the desired condition (1.7). Since the value \( \kappa_2(AB) \) is unknown in most cases, the above criterion (1.9) is usually implemented in the modified form
\[ \sigma_m \leq \rho \varepsilon^2_{\text{red}} \sigma_0, \quad \rho < 1, \] (1.11)
with some heuristic safety factor \( \rho \) understood to compensate for the described uncertainty of the energy error approximation \( \sigma_k \). Nevertheless, this safety factor strongly depends upon the preconditioner and, in particular, must be chosen extremely small for ill-conditioned systems in the absence of preconditioning. However, too stringent choices of \( \rho \) will blow up computing times beyond a reasonable level.
In order to circumvent this undesirable and possibly inefficient termination criterion, we now turn to a different type of energy error approximation. For its derivation, we recur to the basic orthogonality relation

\[ \| x - x_{k+1} \|_A^2 = \| x - x_k \|_A^2 - \| x_{k+1} - x_k \|_A^2, \tag{1.12} \]

which is well-known to hold for any PCG iteration independent of the choice of preconditioner. For convenience we introduce the notation

\[ \epsilon_k = \| x - x_k \|_A^2, \quad \epsilon_k = \| x_{k+1} - x_k \|_A^2 \]

and therefore rewrite (1.12) as

\[ \epsilon_{k+1} = \epsilon_k - \epsilon_k, \tag{1.13} \]

which, with \( \epsilon_n = 0 \), immediately yields that

\[ \epsilon_k = \sum_{l=k}^{n-1} \epsilon_l, \quad 0 \leq k < n. \tag{1.14} \]

The above theoretical termination criterion (1.7) now reads

\[ \epsilon_m \leq \epsilon_{\text{red}}^2 \epsilon_0. \tag{1.15} \]

With (1.5) we observe that

\[ \epsilon_k = \frac{\langle p_k, Ap_k \rangle}{\alpha_k^2} = \frac{\sigma_k}{\alpha_k}. \tag{1.16} \]

In this form, the iterative quantities \( \epsilon_k \) are seen to be extremely cheap to compute. So we may want to use them within an approximation of the above termination criterion (1.15). For this purpose, we replace the unknown solution \( x \) by its best available estimate \( x_{m+1} \). Upon introducing the notation

\[ \epsilon_0^{(m)} = \| x_{m+1} - x_0 \|_A^2 = \sum_{k=0}^{m} \epsilon_k \tag{1.17} \]

we obviously have the relation

\[ \epsilon_0^{(m)} = \epsilon_0 - \epsilon_{m+1} \leq \epsilon_0. \tag{1.18} \]

A corresponding bound for the left hand side of (1.15) can be obtained recalling that the cg method is superlinearly convergent even in Hilbert spaces (see e.g.
so that there exists a threshold index $k_0$ and an associated contraction factor $\Theta$ such that

$$\varepsilon_k \leq \Theta \varepsilon_{k-1}, \quad \Theta < 1 \quad k > k_0.$$ (1.19)

This assumption implies that

$$\varepsilon_m \leq \bar{\varepsilon}_m = \frac{\varepsilon_m}{1 - \Theta}.$$ (1.20)

From (1.18) and (1.20) we can see that the condition

$$\bar{\varepsilon}_m \leq \varepsilon^2 \text{red}^{(m)}$$ (1.21)

is sufficient to guarantee condition (1.15). In the absence of better information an estimate of the kind

$$\Theta \doteq \hat{\Theta}_m = \frac{\varepsilon_m}{\varepsilon_{m-1}}$$ (1.22)

will be applied.

Finally, we want to emphasize that this device now allows to perform a fair efficiency comparison of different preconditioners $B$ including even the case $B = I$, since it is essentially independent of the choice of $B$.

### 1.2 Matching of Inner and Outer Iteration

We now turn to the infinite dimensional case. Assume that the PDE problem is given in the weak formulation (1.1) in terms of the $L_2$ inner product $\langle \cdot, \cdot \rangle$ and the positive definite bilinear form $a(\cdot, \cdot)$ in some appropriate Hilbert space $H$. The CPCG method to be described is applicable over the whole range of elliptic PDE problems without specification of the preconditioner. As already mentioned, the Galerkin method applied to the sequence of nested finite element spaces $S_j, j = 0, 1, \ldots, i$ generates a sequence of associated Galerkin approximations $u_j \in S_j$. Let $n_j = \text{dim}S_j$. For ease of writing we will not distinguish between the solutions $u_j$ of the Galerkin equations

$$a(u_j, v) = \langle f, v \rangle \quad v \in S_j,$$ (1.23)

and the exact solutions $u_j$ of the corresponding systems of $n_j$ linear equations

$$A_j u_j = b_j.$$ (1.24)

The meaning will be clear from the context.
**Galerkin method as outer iteration.** Even though the Galerkin method is actually applied only to a finite sequence of FE spaces, we may nevertheless imagine to continue this sequence by a virtual tail thus obtaining an infinite sequence of spaces. At discretization level \( j \), let

\[
e_j = u - u_j, \quad \epsilon_j = a(e_j, e_j), \quad \varepsilon_j = a(u_{j+1} - u_j, u_{j+1} - u_j)
\]
denote the discretization errors and corresponding norms. Then, under the assumption that the spaces \( S_j \) are nested, the following orthogonality relation is well-known to hold

\[
\epsilon_{j+1} = \epsilon_j - \varepsilon_j, \quad (1.25)
\]

which is exactly the same relation as (1.13) for the PCG iteration. For an appropriate sequence of subspaces of the Hilbert space \( H \) we have convergence \( u_j \to u \) and therefore \( \epsilon_{\infty} = 0 \), which implies that

\[
\epsilon_j = \sum_{l=j}^{\infty} \varepsilon_l, \quad j = 0, 1, \ldots \quad (1.26)
\]

This is just the analog of the finite dimensional representation (1.14) for the PCG iteration. We may therefore most naturally regard the sequence of Galerkin approximations as an outer iteration within the CPCG iteration. As for the convergence of this sequence, we will just copy (1.15) to obtain the theoretical termination criterion

\[
\epsilon_i \leq \varepsilon_h^2 \epsilon_0, \quad (1.27)
\]

with some input parameter \( \varepsilon_h \) to be prescribed by the user (the subscript \( h \) stands for mesh size) and some index \( i \) to count for the finest actually computed discretization level. Upon proceeding as in the previous section, we will try to replace the not implementable termination criterion (1.27) by a sufficient and implementable termination criterion. As above, let \( \Theta \) denote a contraction factor understood to satisfy the relation

\[
\epsilon_j \leq \Theta \epsilon_{j-1}, \quad \Theta < 1 \quad j > j_0
\]

for some threshold index \( j_0 \). We introduce the notation

\[
\epsilon_0^{(i)} = a(u_{i+1} - u_0, u_{i+1} - u_0) = \epsilon_0 - \epsilon_{i+1} = \sum_{j=0}^{i} \varepsilon_j, \quad \bar{\epsilon}_i = \frac{\epsilon_i}{1 - \Theta}, \quad (1.28)
\]

For illustration purposes, consider the Laplace equation on some polygonal domain \( \Omega \) 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 ultimately lead to
\[ \Theta \doteq \Theta_h = \frac{1}{4}, \quad \bar{\epsilon}_i \doteq \frac{4}{3} \epsilon_i. \] (1.29)
This choice of \( \bar{\epsilon}_i \) is meant to take care of the virtual tail of the outer iteration. In the case of adaptive meshes — assuming energy error equidistribution and once more linear finite elements — we will expect a contraction factor
\[ \Theta \doteq \hat{\Theta}_j = \left( \frac{n_{j-1}}{n_j} \right)^\frac{3}{2}, \quad j > j_0, \] (1.30)
on the basis of theoretical results of [2]. This choice, too, will be used later in the appropriate context.
Because of the relations \( \epsilon_0^{(i)} \leq \epsilon_0, \bar{\epsilon}_i \leq \epsilon_i \), we end up with the sufficient condition
\[ \bar{\epsilon}_i \leq \epsilon_0^2 \epsilon_0^{(i)} \] (1.31)
for termination at finite element level \( i \). The implementation of this criterion requires the iterative quantities \( \epsilon_j \), which — as we will see below — can be obtained cheaply from the PCG iteration.

**PCG method as inner iteration.** In previous versions of cascade type algorithms — such as [5, 8, 7] — the criterion (1.31) could not be used, since there only rough discretization error indicators from the mesh refinement strategies were available. In the absence of any better estimator, these rough indicators were also used to monitor the PCG iteration by means of a termination criterion of the undesirable type (1.11). In what follows, an alternative technique based on the preceding considerations 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 and \( e_{j,k} = u - u_{j,k} \) for the iterative errors. On levels \( j = 1, 2, \ldots, i \), the iteration index \( k \) formally runs within \( k = 0, \ldots, n_j \). 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
\[ u_{j,0} = u_{j-1}, \quad u_j = u_{j,n} \quad \text{for } n = n_j. \]
As introduced in (1.16), let
\[ \epsilon_{j,k} = \frac{\sigma_k}{\alpha_k} \quad \text{on levels } j > 0 \]
denote the iterative energy error contributions. Then the key observation is that the discretization errors can be represented as
\[ \epsilon_{j-1} = a(u_j - u_{j-1}, u_j - u_{j-1}) = \sum_{k=0}^{n_j-1} \epsilon_{j,k}. \] (1.32)
This result can be easily seen from
\[
a(u_j - u_{j-1}, u_j - u_{j-1}) = a(u_{j,n} - u_{j,0}, u_{j,n} - u_{j,0}) = \langle e_{j,0}, A_j e_{j,0} \rangle = \sum_{k=0}^{n_j-1} \varepsilon_{j,k}.
\]
In words: The (exact) PCG iteration on discretization level \( j \) supplies the energy norm of the 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
\[
\tilde{u}_{j,0} = \tilde{u}_{j-1}, \quad \tilde{u}_j = \tilde{u}_{j,m+1} \quad \text{for some truncation index} \ m = m_j. \tag{1.33}
\]
With the computationally available quantities \( \tilde{\varepsilon}_{j,k} \) instead of the \( \varepsilon_{j,k} \) we obtain the analog of result (1.32) now in the form
\[
\tilde{\varepsilon}_{j-1} = a(\tilde{u}_j - \tilde{u}_{j-1}, \tilde{u}_j - \tilde{u}_{j-1}) = \sum_{k=0}^{m_j} \tilde{\varepsilon}_{j,k}. \tag{1.34}
\]
The associated discretization errors
\[
\tilde{\varepsilon}_j = a(u - \tilde{u}_j, u - \tilde{u}_j)
\]
can be seen to satisfy
\[
\tilde{\varepsilon}_j = a(u - \tilde{u}_j, u - \tilde{u}_j) \geq \min_{v \in \mathcal{S}_j} a(u - v, u - v) = a(u - u_j, u - u_j) = \varepsilon_j. \tag{1.35}
\]
Herein the FE spaces are 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 (1.25) the recursion
\[
\tilde{\varepsilon}_{j+1} = \tilde{\varepsilon}_j - \tilde{\varepsilon}_j \tag{1.36}
\]
holds — this time not as a consequence of the Galerkin minimization property, but due to the cascade property (1.33) 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 terms \( \tilde{\varepsilon}_j \) by computationally available terms of the form
\[
\tilde{\varepsilon}_0^{(i)} = \sum_{j=0}^{i} \tilde{\varepsilon}_j. \tag{1.37}
\]
With \( \tilde{\epsilon}_0 = \epsilon_0 \) — due to the direct solution on the coarse grid — and (1.35) 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 (1.31) can be replaced by \( \tilde{\epsilon}_0^{(i)} \).

We now turn to the approximation of the left–hand side of (1.31), which means that we have to consider an approximation of the quantity \( \epsilon_i \) at the final level \( i \). Unfortunately, this quantity cannot be bounded on either side by its associated estimate \( \tilde{\epsilon}_i \), 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 \( i \) only the discretization error \( \tilde{\epsilon}_{i-1} \) for the previous level is actually available. In this situation, we recur to the result (1.30) for the approximation
\[
\epsilon_i \doteq \hat{\Theta}_i \tilde{\epsilon}_{i-1}, \quad \text{(1.38)}
\]
which then means to approximate the termination criterion (1.31) by the approximate termination criterion
\[
\hat{\epsilon}_i = \frac{\hat{\Theta}_i \tilde{\epsilon}_{i-1}}{1 - \hat{\Theta}_h} \leq \epsilon_h^{2 \tilde{\epsilon}_0^{(i)}}. \quad \text{(1.39)}
\]
This criterion will be used for the automatic determination of the actually needed discretization level \( i \) according to the user prescribed relative accuracy \( \epsilon_h \). The term \( \hat{\epsilon}_i \) will be returned to the user as discretization error estimate on level \( i \).

We are therefore only 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 the termination criterion (1.21) we will require
\[
\frac{\tilde{\epsilon}_{j,m}}{1 - \Theta_m} \leq \hat{\rho}^2 \hat{\Theta}_j \tilde{\epsilon}_{j-1} \quad \text{for some truncation index } m = m_j, \quad \text{(1.40)}
\]
wherein the safety factor \( \hat{\rho} \) must be chosen as an internal default parameter. Obviously, the above condition is most stringent on the final level \( i \). Upon combining the above results (1.34) and (1.36), we recall the following global error relation
\[
\tilde{\epsilon}_i = a(u - \tilde{u}_i, u - \tilde{u}_i) = \epsilon_0 - \sum_{j=1}^{i} \sum_{k=0}^{m_j} \tilde{\epsilon}_{j,k}. \quad \text{(1.41)}
\]
This relation seems to indicate that the final level restriction from (1.40) should already be observed on coarser levels to avoid unnecessary more costly iterations.
on the finest levels. Therefore we suggest to replace criterion (1.40) by the following iterative termination criterion for the inner PCG iteration

\[
\frac{\tilde{\varepsilon}_{j,m}}{1 - \hat{\Theta}_m} \leq (\hat{\rho} \varepsilon_h)^2 \tilde{\varepsilon}_0^{(i)} \quad \text{for some truncation index } m = m_j.
\] (1.42)

Obviously, this criterion is now more stringent on coarser levels — which, however, turned out to safeguard the adaptive mesh refinement device and, at the same time, to save costly iterations on finer levels. In total, numerical experiments confirmed the expectation that condition (1.42) is more efficient than condition (1.40). Robust runs were obtained with an empirical choice of \( \hat{\rho} = \frac{1}{4} \).

With the two termination criteria (1.39) for the Galerkin outer iteration and (1.42) for the inner PCG iteration, the whole CPCG iteration control is now complete. It applies independent of any special choice of preconditioner or even without any preconditioner.

**Remark.** 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 are dissolved from one level to the next. We have corrected the affected ones of the above formulas in terms of the thus introduced energy error differences. However, the effect was so minor that this modification was finally decided to be left out.

## 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. This means that in the above derived CPCG control the problem dependent choice \( \Theta_h = 0.25 \) has to be inserted — see (1.29). The picture in 3–D appeared to be essentially the same as in 2–D, only computing times and storage requirements for the test runs blew up considerably. Therefore, for the mere purpose of illustration, the subsequent presentation is restricted to the 2–D case. Three examples testing three different algorithmic aspects will be considered.
Example 1: *Peak Problem.* Given the PDE

\[-\Delta u = f,\]

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: Example 1, solution on level $j = 7$, $n = 1975$ nodes (adaptive mode).
Example 2: Re-entrant corner problem. Given the PDE

$$-\Delta u = 0 ,$$

Dirichlet boundary conditions are imposed such that

$$u = r^{1/\alpha} \sin(\phi/\alpha) , \quad \alpha = 7/2$$

is the solution.

Figure 2: Example 2, solution on level $j = 26$, $n = 4702$ nodes (adaptive mode).
Example 3: *Polynomial problem.* Given the PDE

$$-\Delta u = -2(x^2 + y^2 - x - y),$$

Dirichlet boundary conditions are imposed such that

$$u = x(x - 1)y(y - 1)$$

is the solution.

![Example 3 solution](image)

Figure 3: Example 3, solution on level $j = 6$, $n = 33025$ nodes (uniform mode).
The comparative performance of the following three variants of the CPCG algorithm will be presented:

- CPCG–HB: algorithm with *hierarchical bases* preconditioner [14]
- CCG: algorithm *without any* preconditioner

In the simple case of the Laplace equation, the HB preconditioner is known to be nearly uniformly spectrally equivalent in 2–D — see [14], whereas the BPX preconditioner is known to be uniformly spectrally equivalent independent of the space dimension — see [9]. In 2–D, the expected numerical efficiency will be nearly the same for both preconditioners. Based on the subtle condition number estimates of Xu [13], 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 slightly less disastrous in highly non–uniform grids (with an additive 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 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 energy norm improvement factor from initial to final grid; since, in the best case, we can expect an iterative improvement of 1 binary digit of accuracy for the Laplace equation with linear finite elements, we will count in binary digits below. All 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 [7] 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 [8] 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 [7]. Let $\eta_j$ denote the contribution from a single edge out of the (usually non–uniform) mesh associated with level $j$. Then $\eta_{j-1}$ is the contribution of the corresponding father edge. The extrapolation technique due to [1] then supplies certain guesses

$$\tilde{\eta}_{j+1} = \frac{\eta_j^2}{\eta_{j-1}}$$

(2.1)

for the expected contributions from the son edges on an imagined level $j + 1$ obtained from uniform mesh refinement. There are several reasonable heuristics to exploit this information. In connection with the CPCG scheme, the following strategy is favored here: Whenever an edge contribution on level $j$ is above the threshold value

$$\eta_{cut} = \min\{\max \tilde{\eta}_{j+1}, \Theta_h \max \eta_j\},$$

(2.2)

then this edge is refined for the realized level $j + 1$. The above heuristic choice $\Theta_h$ reflects the idea of thinning out uniform meshes without losing the associated error improvement.

In Fig. 4 and Fig. 5, the initial and certain refined grids are shown for Example 1 and Example 2, respectively.
Figure 4: Example 1, grids on levels $j = 0$ and $j = 7$.

Figure 5: Example 2, grids on levels $j = 0$ and $j = 15$. 
In Fig. 6 (Example 1) and Fig. 7 (Example 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 in both cases. The comparison in terms of number of iterations is given (for Example 1) in Fig. 8. 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 (1.42), 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. It should be mentioned that the above Fig. 8 does not contradict Fig. 10, p. 3198 in [8], 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.
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, \textit{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 in two examples: by construction, Example 3 does not require a non-uniform mesh (compare Fig. 3), whereas Example 1 does require a non-uniform mesh for efficiency reasons (compare Fig. 1).

![Figure 9: Uniform mode, Example 1.](image)
In Fig. 9 (Example 1) and Fig. 10 (Example 3) the comparative iteration times are presented. Obviously, the uniform mode here does not show any visible difference in the effect that has already been observed 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.

**Remark.** In the 3–D comparison runs, CCG held its clearly superior position, whereas HB and BPX interchanged their role — as expected from their spectral equivalence properties.
Figure 11: Uniform versus adaptive mode, Example 1.

The above experiments should not leave the impression that adaptivity does not pay off. For this reason, Fig. 11 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 the limiting factor in 3-D!
Conclusion

The present paper derived a simple but efficient strategy to control the discretization errors of the Galerkin FEM in combination with the PCG iteration errors in terms of the associated energy error norms. In a unified setting, the relative merits of different preconditioners versus the case of no preconditioning could be reliably 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 fact, the asymptotic behavior of the CCG method turned out to be the same as the one of the CPCG method with BPX preconditioning — a preconditioner, which is known to be uniformly spectrally equivalent. In the test examples, 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 explanation of these observations will be given in Part II of this paper.

Acknowledgements

The author gratefully acknowledges invaluable computational assistance by R. Roitzsch and B. Erdmann.
References


