TWO GRID ITERATION WITH A CONJUGATE GRADIENT SMOOTHER APPLIED TO A GROUNDWATER FLOW MODEL

M.J. HAGGER*, K.A. CLIFFE†, AND A. SPENCE‡

Abstract. The well known two grid method is used to solve the large, sparse, linear system resulting from a finite element method discretisation of $-\nabla.(k\nabla p) = 0$. Several possible choices of smoother are considered. When tested on a physically realistic problem it is found that the use of the conjugate gradient method is particularly effective. A "computational modal analysis" on a model problem, discretised using biquadratic basis functions over 9 node quadrilaterals, is used to study the performance of each smoother. The use of conjugate gradients preconditioned by the two grid method is also considered, but this is shown to be less effective than the standard two grid method.

1. Introduction. In this paper we consider the use of the two grid method, with various smoothers, for solving systems of linear equations arising from a finite element discretisation of the 2 dimensional elliptic partial differential equation:

$$-\nabla.(k\nabla p) = 0 \tag{1.1}$$

with appropriate boundary conditions. A standard finite element method (FEM) formulation is used: choose a set of nodal basis functions Φ_j in some N dimensional finite element space. Seek a solution of the form $p = \sum_{j=1}^{N} x_j \Phi_j$, where the x_j represent the unknown nodal values of p. Hence obtain the $N \times N$ linear system $A\mathbf{x} = \mathbf{b}$, where A is the symmetric, positive definite, stiffness matrix and $\mathbf{b}$ depends on the boundary conditions. For more details on the finite element method see [15].

Equation (1.1) is used to model groundwater flow in porous media, where p represents the pressure and k the coefficient of permeability. Efficient solution of the groundwater flow problem is essential for many important problems, see, for example, [1]. Typical models, particularly in 3 dimensions, produce very large linear systems where direct solvers become impractical, due to large requirements of CPU time and memory. A number of iterative techniques for robust and efficient solution for problems of this type have been tried, see for example [1, 14]. We shall demonstrate the effectiveness of the two grid method as a solution technique for large 2 dimensional discretisations of (1.1), with approximately 60,000 degrees of freedom, arising from a physically realistic problem. The CPU time taken by the two grid method will be compared with that from a direct solver (MA32 from the Harwell subroutine library).

To obtain the results in this paper the Harwell groundwater flow modelling package, called NAMMU, is used. This is a very general package with a number of features for modelling, solving and displaying solutions of problems of the type (1.1). In the groundwater model that we will consider, piecewise biquadratic basis functions on 2 dimensional 9 node isoparametric quadrilateral elements are used for the discretisation. For more details on the finite element discretisation used and the specific features of the NAMMU package, see [13].

---

* School of Mathematical Sciences, University of Bath, Bath, BA2 7AY, United Kingdom. This author's research was supported by the Science and Engineering Council of UK, CASE studentship at Bath University and AEA Technology Harwell.

† Hydrogeology Department, B424.4, Harwell Laboratories, Didcot, OX11 0RA, United Kingdom (andrew.cliffe@uk.orgn.aea). This author's research forms a part of the underlying Corporate Research Programme of AEA Technology.

‡ School of Mathematical Sciences, University of Bath, Bath, BA2 7AY, United Kingdom (A.Spence@uk.ac.bath.maths).

The coefficient k is typically piecewise constant, with values ranging from 10^{12} to 10^{18} in realistic problems. In addition the creation of an accurate model of a typical underground rock structure often leads to highly unstructured grids with a range of element sizes, see, for example figure 6. The combination of discontinuities in k and the unstructured grid means that the resulting linear system is highly ill-conditioned, which creates problems with many iterative methods. Indeed, methods which perform well on more straightforward problems, like conjugate gradients with diagonal preconditioning or polynomial preconditioning, perform extremely poorly for this problem, see [12] for more details on this.

As has already been mentioned, biquadratic finite elements on quadrilaterals are used for the discretisation, the effect of which is that the resulting stiffness matrix is unlikely to be either diagonally dominant or an M-matrix. As a consequence of using the NAMMU package the resulting stiffness matrix is stored in an element form. In this form matrix-vector multiplications can be carried out very efficiently, but more complicated operations, involving the whole matrix A, or parts of the matrix, are not easily performed. For this reason when discussing the smoothers in the two grid iteration we will restrict attention to iterative methods that only require matrix-vector multiplications. This eliminates Gauss-Seidel and SOR smoothers.

The standard formulation of a two grid method is stated in §2, and follows the description in [10]. In §3 the following choices of smoother are considered:

- Richardson iteration
- Gradient method
- Conjugate gradients (CG)
- Conjugate residuals (CR).

In each case the linear system is either symmetrically diagonally scaled or a diagonal preconditioner is used. Note that Richardson's method with a diagonal preconditioner is identical to Jacobi's method. It is well known that for problems of the form (1.1), diagonal preconditioning is very effective at mitigating the influence of the discontinuous k, see, for example [17, 9]. In order to analyse the effect of these smoothers we carry out a "computational modal analysis" on a model problem to compare their effectiveness.

In §4 the use of the two grid method as a preconditioner for the conjugate gradient method is considered. The results in §5 show that, although this method converges satisfactorily, this approach is inferior to the standard two grid method with either conjugate gradients or conjugate residuals as a smoother.

Also in §5 we present results for realistic industrial problems, using the two grid method and the preconditioned conjugate gradient method. Both the conjugate gradient and conjugate residual methods are shown to be effective smoothers, whereas use of Richardson iteration or the gradient method as smoother gives poor overall rates of convergence.

For the rest of this paper we shall denote a fine grid linear system problem as

$$A_f \mathbf{x} = \mathbf{b},$$

this being a problem with N_f degrees of freedom. The corresponding coarse grid problem will have N_c degrees of freedom and the associated stiffness matrix will be denoted A_c, and we shall assume that each coarse grid point is also a point on the fine grid. One strategy in the two grid method is to halve the number of elements in each spatial direction on the fine grid to produce the coarse grid. The details of the FEM then mean that $N_c \approx \frac{1}{4} N_f$.

2. Two Grid Method. The two grid method employed, which will henceforth be referred to as TGM, is described in [10]. We denote prolongation and restriction matrices that transfer vectors between the grids by P and R respectively. Also we assume that an LU factorisation of A_c has been obtained before starting any TGM steps and so solves on the coarse grid require only forward and back substitutions.

The steps in the TGM are as follows: from an initial vector $\mathbf{x}_0$ we perform a coarse grid correction followed by $\mu(\geq 1)$ smoothing steps to produce $\mathbf{x}_1$:

$$\mathbf{r} = \mathbf{b} - A_{\mathrm{f}}\mathbf{x}_0 \tag{2.2}$$

$$\mathbf{r}_{\mathrm{c}} = R\mathbf{r} \tag{2.3}$$

$$\mathbf{d}_{\mathrm{c}} = A_{\mathrm{c}}^{-1}\mathbf{r}_{\mathrm{c}} \tag{2.4}$$

$$\mathbf{x}_{\frac{1}{2}} = \mathbf{x}_0 + P\mathbf{d}_{\mathrm{c}} \tag{2.5}$$

$$\mathbf{x}_1 = S^{\mu}(\mathbf{x}_{\frac{1}{2}}) \tag{2.6}$$

Note that steps (2.2-2.5) can be together written as the *coarse grid correction* step

$$\mathbf{x}_{\frac{1}{2}} = \mathbf{x}_0 + PA_c^{-1}R(\mathbf{b} - A_f\mathbf{x}_0). \tag{2.7}$$

Step (2.6) represents the application of μ steps of the *smoothing iteration* and one step of the TGM can be written as

$$\mathbf{x}_1 = S^{\mu}[\mathbf{x}_0 + PA_{\mathrm{c}}^{-1}R(\mathbf{b} - A_{\mathrm{f}}\mathbf{x}_0)]. \tag{2.8}$$

In order to solve a given problem, this two grid step is applied a number of times until some preset convergence criteria is met.

The choices of smoothers will be examined in more detail in §3, together with some discussion on convergence analysis. For the remainder of this section we consider the choice of prolongation and restriction matrices for the finite element discretisation of (1.1).

2.1. Prolongation. First denote the i^{th} FEM nodal basis function for the fine grid as Φ_i^f, and the m^{th} coarse grid basis functions as Φ_m^c. By requiring that all coarse grid points are also fine grid points it is immediately apparent that the coarse grid basis functions can be written in terms of the fine grid basis functions, that is

$$\Phi_m^c = \sum_{i=1}^{N_f} a_i^{(m)}\Phi_i^f \tag{2.9}$$

for some coefficients $a_i^{(m)}$. Hence, if τ_j^f denotes the j^{th} fine grid node point then clearly

$$\Phi_m^c(\tau_j^f) = a_j^{(m)} \tag{2.10}$$

since the nodal basis functions satisfy $\Phi_i^f(\tau_j^f) = \delta_{ij}$.

In order to define the prolongation matrix from $\mathbb{R}^{N_c}$ to $\mathbb{R}^{N_f}$, consider any vector $\mathbf{q}^c \in \mathbb{R}^{N_c}$ with $(\mathbf{q}^c)_m = q_m^c$. There is a corresponding function in the coarse finite element space defined by

$$q^c(\tau) = \sum_{m=1}^{N_c} q_m^c\Phi_m^c(\tau). \tag{2.11}$$

Evaluation of (2.11) at the fine grid nodes τ_j^f, $j = 1, ..., N_f$, gives the corresponding fine grid vector, $\mathbf{q}^f$, with the j^{th} component given by

$$q_j^f = \sum_{m=1}^{N_c} q_m^c a_j^{(m)}. \tag{2.12}$$

Hence,

$$\mathbf{q}^f = P\mathbf{q}^c \tag{2.13}$$

where the rectangular matrix P is given by

$$P_{jm} = a_j^{(m)}, \tag{2.14}$$

these coefficients $a_j^{(m)}$ being given in (2.10) above.

2.2. Restriction. First note that in the matrix-vector description of the TGM the restriction operation in (2.3) acts on the fine grid residual and produces a "coarse grid residual". To define an appropriate restriction for the FEM it is natural to consider the residuals of an approximate solution to the weak form of the pde with respect to both the fine and coarse meshes.

Given some approximate solution, p^f, the i^{th} residual corresponding to the i^{th} basis function, on the fine grid, is given by

$$r_i^f = \int_\Omega k\nabla p^f.\nabla\Phi_i^f - \int_{\partial\Omega} \Phi_i^f k\nabla p.\mathbf{n} \tag{2.15}$$

where Ω and $\partial\Omega$ denote the domain and the domain boundary respectively, $\mathbf{n}$ denotes the outward unit normal and Φ_i^f is the i^{th} fine grid basis function. Correspondingly, on the coarse grid the m^{th} residual is given by

$$r_m^c = \int_\Omega k\nabla p^f.\nabla\Phi_m^c - \int_{\partial\Omega} \Phi_m^c k\nabla p.\mathbf{n} \tag{2.16}$$

where Φ_m^c is the m^{th} coarse grid nodal basis function. Using (2.9) we have

$$\begin{aligned} r_m^c &= \sum_{i=1}^{N_f} \left(\int_\Omega k\nabla p^f.a_i^{(m)}\nabla\Phi_i^f - \int_{\partial\Omega} a_i^{(m)}\Phi_i^f k\nabla p.\mathbf{n} \right) \\ &= \sum_{i=1}^{N_f} a_i^{(m)} \left(\int_\Omega k\nabla p^f.\nabla\Phi_i^f - \int_{\partial\Omega} \Phi_i^f k\nabla p.\mathbf{n} \right) \end{aligned} \tag{2.17}$$

which by (2.15), gives

$$r_m^c = \sum_i a_i^{(m)} r_i^f. \tag{2.18}$$

In matrix form we can write

$$\mathbf{r}^c = R\mathbf{r}^f \tag{2.19}$$

where $\mathbf{r}^c$ and $\mathbf{r}^f$ denote the vectors with components r_m^c and r_i^f respectively. The rectangular matrix R is given by

$$R_{mj} = a_j^{(m)} \tag{2.20}$$

with the coefficients $a_j^{(m)}$ given by (2.10) above.

Finally, by comparing the definitions of the prolongation and restriction matrices, ie (2.14) and (2.20), we see that

$$R = P^T. \tag{2.21}$$

From a computational point of view this has obvious advantages; only the prolongation matrix needs to be calculated thus halving both the calculations required and the storage requirements. In addition, when the two grid iteration is used as a preconditioner for the conjugate gradient method, this choice also ensures that the preconditioning matrix is symmetric, which is important to ensure that the resulting method will converge, see [8, §10.3]. In Hackbusch[10, §3.6], this relation between R and P is discussed, in a more abstract form, and is called the "canonical" choice for a finite element method. The idea of using the weak form of the pde to define the restriction matrix, which links the residuals on the two grids, is well known, see for example [7].

3. Different smoothers on a model problem. The overall philosophy of the two grid method, and multigrid methods in general, is well known (see [4, 10, 6]) and so we merely sketch the main points which are relevant to this paper. In order to provide an efficient solution procedure the smoother and coarse grid solver must work in tandem. For theoretical purposes only, consider an expansion of the error in a fine grid solution, $\mathbf{x}_i$ say, of the form

$$\mathbf{e}_i = \mathbf{x} - \mathbf{x}_i = \sum_{j=1}^{N_f} \alpha_i^{(j)} \mathbf{v}_j \tag{3.22}$$

where $\mathbf{v}_j$ are eigenvectors of A_f, corresponding to eigenvalues $\lambda_j(\lambda_1 \leq \lambda_2 \leq \ldots \leq \lambda_{N_f})$. As a rough classification, the error components, $\alpha_i^{(j)}$, for $j = 1, \ldots, N_c$ are termed the "low frequency" components, and for $j = N_c + 1, \ldots, N_f$ they are the "high frequency" components. (Recall that for a typical 2 dimensional problem we will have $N_c \approx \frac{1}{4} N_f$. Now, the hope is that the smoothing iteration should damp out the high frequency errors, while the coarse grid correction should reduce the low frequency errors. However, as is seen by the simple example in [10, p27], the coarse grid correction step also redistributes the error over all frequencies. We see this effect clearly in a computational modal analysis on a model problem later in §3.1.

Recall that for our smoothers we only consider iterative methods that are based on matrix-vector multiplications. This leads to the four choices: Richardson iteration, gradient method, conjugate gradient method and conjugate residual method. For each of these methods a diagonal preconditioning is also applied. For more details on these iterative methods see [8, 11].

In order to understand the convergence properties of stationary smoothers it is common to examine the reduction on each eigenmode, or frequency, of the error, see Hackbusch[10, §2.4]. For example, the following simple analysis gives the best choice of damping for the Richardson method. Define the damped Richardson method as:

$$\hat{\mathbf{x}} = \mathbf{x} + \omega(\mathbf{b} - A\mathbf{x}), \tag{3.23}$$

now, by writing the error as in (3.22) we can write the error component in the j^{th} eigendirection after one Richardson step as

$$\alpha_{i+1}^{(j)} = (1 - \omega\lambda_j)\alpha_i^{(j)}.$$

With the need to smooth high frequency components in mind, we require that $|1 - \omega\lambda_j|$ be minimised for all those eigenvalues in the fine grid component range, which is taken to be approximately

$$\frac{\lambda_N}{4} \leq \lambda_j \leq \lambda_N$$

corresponding to the high frequency range of eigenvectors (when $N_c \approx \frac{1}{4}N_f$). We deduce that the best choice for ω is

$$\omega_{best} = \frac{8}{5\lambda_N}. \tag{3.24}$$

For the theoretical modal analysis to be useful for general matrices and other smoothers it is best if simple expressions for the eigenvectors are known. For the matrices derived from the FEM with biquadratic basis functions over 9 node quadrilaterals simple expressions are not known. In addition, theoretical analysis of the non-stationary smoothers is not easy, in general, and it is difficult to interpret the results in a helpful way. Hence the "computational modal analysis" in the following section is used to compare the four smoothers.

3.1. Computational Modal Analysis. Consider a small, 2 dimensional discretisation of (1.1), on a uniform mesh and over a rectangular domain, and with constant k over the entire domain. Hence (1.1) reverts to Laplace's equation, ie

$$\nabla^2 p = 0. \tag{3.25}$$

We construct a uniform mesh of quadrilaterals and use biquadratic basis functions such that the fine grid problem has 1056 degrees of freedom and the corresponding coarse grid problem has 289. For a fine grid problem of this size we can compute both the exact solution, using the direct solver, and all the eigenvectors and eigenvalues of the fine grid stiffness matrix. Hence we can explicitly calculate the error components in the expansion (3.22). This is useful because, as was said earlier, it is not possible to carry out a modal analysis along the lines of [10, p25].

In order to see the distribution of error over the entire eigenspectrum we plot the modulus of the coefficients $\alpha_j^{(i)}$ against the eigenmode number j, though in order to improve readability and understand better the qualitative features of the results we group the error components $\alpha_j^{(i)}$ in sets of 5 and plot the maximum modulus in each set. More precisely we plot the sequence

$$\alpha_k^* = \max\{|\alpha_{5(k-1)+1}^{(i)}|, ..., |\alpha_{5k}^{(i)}|\}. \tag{3.26}$$

In figure 1 we plot these maxima, on a log scale, after one coarse grid correction. The continuous line gives a qualitative idea of the distribution of the maximum error components over the entire frequency range.

To show how the iteration proceeds for a particular smoother, four TGM steps are performed, each with five smoothing steps. The Richardson method using the damping coefficient given by (3.24). In figures 2-5 we compare the smoothers, each figure having the same structure, namely: writing ν as the TGM step number, on the first row the error distribution is shown after:

(a) A coarse grid solve

(b) one smoothing step ($\mu = 1$)

(c) three smoothing steps ($\mu = 3$)
(d) five smoothing steps ($\mu = 5$)
The second, third and fourth rows then show the maximum error components after two, three and four TGM steps.

As predicted, in figure 2 we see that the damped Richardson method is indeed successful at reducing the high frequency error components well. It is also clear from this figure that the coarse grid correction step reduces the error over the coarse grid frequencies but also redistributes the error over all the frequencies.

Figure 3 shows the effect of using the gradient method. Comparing this with the previous Richardson figure it can be seen that there is little difference. However, since the gradient method has a convergence rate very similar to that of the optimally damped Richardson iteration this is not so surprising.

From figures 4 and 5, it is clear that both CG and CR are greatly superior to either the Richardson or gradient smoother, for this simple problem at least. Also the conjugate gradient and conjugate residuals show little apparent difference. Comparison of the final few pictures indicate that the conjugate residual smoother has performed slightly better than the conjugate gradient smoother.

These results are presented purely in terms of iterations, which is potentially misleading since the CG and CR methods require more work per iteration than either the Richardson or gradient method. Later results in §5, on a realistic problem, will demonstrate that this extra work per step is well worth the effort.

For this model problem at least, whilst all the smoothers work, the CG and CR smoothers are easily the most effective.

4. Preconditioned Conjugate Gradients. As discussed in [12], simple diagonal or polynomial preconditioners of CG fail to give an adequate rate of convergence for the physical problem discussed here. Hence a more natural idea is to use a preconditioner that encapsulates more of the features of the underlying pde. This is achieved by the use of the TGM as a preconditioner for the CG method. This method is quite widely used in the literature, see for example [16, 5]. Another common preconditioner is the incomplete factorisation, usually referred to as ILU, see for example [3]. This technique may not capture the features of the pde as well as the TGM preconditioning. In addition, as was discussed in §1, methods that require full assembly of the stiffness matrix A_f are not attractive in the NAMMU environment.

First note that in order to use TGM as a preconditioner we require that the TGM iteration matrix be both symmetric and positive definite. This symmetry condition can be achieved by the choice of P and R in §2 and by using pre and post smoothing steps, S_1^μ and S_2^μ constructed such that $S_1^\mu = (S_2^\mu)^T$. The TGM iteration matrix can then be guaranteed to be positive definite by requiring that the two grid method converges in its own right, see [11, Lemma 10.7.1].

The usual preconditioning step for the CG algorithm involves solving

$$M\mathbf{z}_k = \mathbf{r}_k$$

for $\mathbf{z}_k$, where M is the preconditioning matrix and $\mathbf{r}_k$ is the residual at the k^{th} step, see [8, §10.3]. For the TGM preconditioning this is replaced by applying one step of the TGM, ie as given in the steps (2.2-2.6), applied to the problem

$$A_f\mathbf{z}_k = \mathbf{r}_k$$

with an initial guess of $\mathbf{z}_k^{(0)} = 0$.

In our numerical experiments a single step of Richardson's iteration is used for each of the pre- and post-smoother steps.

Note that this method can be equivalently considered as either TGM preconditioned conjugate gradients or a conjugate gradient accelerated TGM. In the latter form the convergence properties of this method are considered by Hackbusch[11, §10.8.3]. There we see that if the TGM iteration has a convergence rate of γ then using the TGM as a preconditioner for CG yields a combined convergence rate of approximately $\gamma/4$. This result is, to some extent, borne out by the numerical results presented in figure 15.

In the next section we will present results for this version of CG and compare with the standard TGM method discussed in §2.

5. Numerical Results. For the results presented here we consider a model problem for groundwater flow in a complex geological medium with realistic contrasts in the permeability field. A sample grid, with only a few thousand degrees of freedom, is given in figure 6.

Two combinations of grids are considered to compare the effectiveness of the four choices of smoothers from §3 and the TGM preconditioned conjugate gradients.

PROBLEM 1. *Consider a 2 dimensional fine grid mesh with $N_f = 58944$, the coarse grid was obtained by roughly halving the number of elements in each spatial direction, resulting in $N_c = 15021 \approx N_f/4$.*

A second discretisation with a less refined coarse grid was also set up.

PROBLEM 2. *Consider a 2 dimensional fine grid mesh with $N_f = 58944$, the coarse grid was obtained by essentially quartering the number of elements in each spatial direction, resulting in $N_c = 3827 \approx N_f/16$.*

Note that we expect problem 2 to require more overall smoother steps to produce a solution of the same accuracy, since the coarse solution will not represent the fine solution as well as in problem 1. Further note that the coarsening used in problem 1 represents the approach commonly used in the literature.

All the results presented here were run on a Cray YMP. In all cases the methods are compared to the direct frontal solver, the Harwell MA32 code, which requires 65.07 seconds to solve the problem exactly. Where possible this time will be indicated on the graphs by a ∇, (see figure 7). The code MA32 was also used to find the LU factors of A_c that are needed for the coarse grid solves (see (2.4)). For $N_c = 15021$, which is the case in problem 1, the cost of this one-off factorisation was only 2.69 seconds. For all the runs an upper limit of approximately 1400 matrix-vector multiplications was set, inside of which convergence was reasonably expected, although at this limit considerably more CPU time would have been used than for the frontal method.

The exact solution on the fine grid is available, which means that a stopping criteria could be based on a reduction in error. However, we prefer to choose a convergence criterion more appropriate to an iterative method where the solution is unknown. To this end require

$$\frac{\| \mathbf{r}_i \|_2}{\| \mathbf{r}_0 \|_2} < 10^{-7} \tag{5.27}$$

where $\mathbf{r}_i = \mathbf{b} - A_f \mathbf{x}_i$. This is the simplest of choices. Alternative, and perhaps better choices when using CG-like methods, are:

(1) The relative 2-norm of the diagonally preconditioned residual.
(2) the choices in [2], which use an estimate of the condition number of the matrix, to estimate the associated error.

From a practical point of view we would wish to stop when the relative error, ie $\| \mathbf{e}_i \| / \| \mathbf{e}_0 \|$, had been reduced to a level comparable with the error in the FEM discretisation. In all the cases presented here that converged, this level of error was in fact reached.

5.1. Results for the Smoothers. For each of the smoothers we present results for three different values μ and compare the overall solution time. Before presenting the results, we should point out that since the maximum eigenvalue of the fine grid matrix, A_f, is not known a priori, the damping coefficient used for the Richardson method is taken as simply $2/\lambda_G$, where λ_G is a Gershgorin estimate of the largest eigenvalue of A_f.

Consider first the results for problem 1. The first thing to point out about the Richardson and gradient method smoothers is that they failed to reach the convergence criterion in all cases. In fact, as can be seen from figures 7 and 8, after an initial promising start, subsequent convergence of the error was very slow. It is, however, of some interest to note that the gradient method version shows very little dependence on the number of smoother steps performed prior to a coarse grid solve.

The results for the conjugate gradient method in figure 9 are much more satisfactory, the convergence criterion was met in all cases, moreover convergence was achieved in less than one third of the CPU time required for the direct solve for almost all of the number of smoother steps tested. In addition we see a marked difference in performance depending on the number of smoother steps used, the $\mu = 10$ case takes nearly 30% more time than the $\mu = 20$ case, while with $\mu = 40$ convergence is faster again. It is also far more apparent from these graphs at which point the coarse grid corrections occur, and the associated improvement that they give to the solution.

In a similar way to the conjugate gradients case, the conjugate residuals used as a smoother, are seen to be effective, see figure 10. In addition the CR smoothers also show an improved overall convergence rate for larger numbers of smoothing steps. However, comparisons of these two methods show little apparent difference in convergence rates.

In figure 11 the "best" results, in terms of convergence rate, from each of the four methods are compared. It is clear the the CG and CR smoothers are significantly better than either of the Richardson or gradient smoothers, despite the fact that they require more work per iteration. The indication is that relatively simple iterative methods are incapable of solving these problems, and since the more complicated Gauss-Seidel, SOR etc have already been ruled out, see §1, these CG like methods would seem to be the natural choice. Finally note that the gradient method performs better than the Richardson iteration, which is perhaps expected, since in our formulations the gradient method is essentially an "optimally" damped version of the Richardson iteration.

Next consider the CG smoother for problem 2. In figure 12 we compare the results for CG with $\mu = 10, 20$ and 40. Again the performance with $\mu = 10$ is very poor, indeed in this case it now takes more time than the direct solver to converge. As for problem 1 it is again found that $\mu = 40$ gives the best performance.

As a final comparison, working again with problem 1, we compare the error reduction versus iterations, by which it is meant the number of smoother iterations, a comparison of these results with those in figure 11 show very little difference, despite the fact that both the conjugate gradient and conjugate residual algorithms require significantly more work, in terms of scalar products, than the other methods. This is an indicator of how much the matrix-vector multiplications dominate the overall work.

Finally note that for all four of the smoothers considered here it is by no means obvious how to choose μ, the number of smoother steps per TGM step, for optimal convergence. Certainly for the CG and CR cases there is a marked difference in performance between the smallest and largest numbers. From the results presented here it would seem that choosing a large number ($\mu = 40$) is best.

5.2. TGM preconditioning. First consider problem 1. For comparison purposes the results for the best CG smoothed TGM are also plotted, see figure 13. The CG smoothed code is clearly more effective, although it should be borne in mind that this is for the "best" choice of number of smoothing steps, which as yet is not something that can be predicted prior to running the code. For comparison purposes again recall that the direct solver takes 65.07 seconds to solve the problem.

In figure 14 the results for problem 2, which has the less refined coarse grid, are given. Again we find that the CG smoothed TGM algorithm is far superior.

It should be noted that for both these examples the convergence does become quite slow in the later stages, perhaps indicating that this method is not so robust as the CG smoothed TGM.

Finally in figure 15 a comparison of TGM, preconditioned TGM, and Richardson smoothed TGM, with 1 step pre and post smoothing is made. This being a plot of error versus iteration count. Here we do indeed see a speedup of at least 1/4 as predicted by the result from Hackbusch[11, §10.8.3].

6. Conclusions. For FEM discretisations of (1.1), where the grid is highly non-uniform, the two grid approach with conjugate gradients as a smoother is a highly effective method. A large number of smoother steps, would seem to be the best choice. For the large 2 dimensional problem considered here, with nearly 60,000 degrees of freedom and run on a Cray YMP, the TGM with conjugate gradient smoother gives a solution in less than 30% of the time taken by the MA32 direct solver.

Finally, using a two grid method to precondition a conjugate gradient method, whilst giving a method that converges, uses about twice as much CPU time as the standard two grid method.

REFERENCES

[1] S. F. Ashby, R. D. Falgout, S. G. Smith, and A. F. B. Tompson, *Modeling groundwater flow on MPPS*, in Proceedings of the scalable parallel libraries conference, 1993.

[2] S. F. Ashby, M. J. Holst, T. A. Manteuffel, and P. E. Saylor, *The role of the inner product in stopping criteria for conjugate gradient iteration*, Tech. Report UCRL-JC-112586, November 1992.

[3] O. Axelsson and V. A. Barker, *Finite element solution of boundary value problems - theory and computation*, Academic press, 1984.

[4] R. Bank and C. Douglas, *Sharp estimates for multigrid rates of convergence with general smoothing and acceleration*, SIAM J. on Numer. Anal., 22 (1985).

[5] D. Braess, *On the combination of the multigrid method and conjugate gradients.*, in Multigrid Methods II, 1985.

[6] A. Brandt, *Multi-level adaptive solution to boundary-value problems*, Math. Comp., 31 (1977), pp. 333–390.

[7] H. Deconinck and C. Hirsch, *A multigrid finite element method for the transonic potential equation*, in Multigrid Methods, 1981.

[8] G. H. Golub and C. F. V. Loan, *Matrix Computations, second edition.*, John Hopkins University Press, 1989.

[9] A. Greenbaum, *Diagonal scalings of the Laplacian as preconditioners for other elliptic differential operators*, SIAM J. on Matrix Anal. and Appl., 13 (1992), pp. 862–846.

[10] W. Hackbusch, *Multigrid Methods and Applications*, Springer-Verlag, 1985.
[11] ———, *Iterative Solution of Large Sparse Systems of Equations*, Springer-Verlag, 1994.
[12] M. Hagger, *Iterative solution of large, sparse systems of equations, arising in groundwater flow models*, PhD thesis, University of Bath, To be completed in 1994.
[13] L. Hartley and C. Jackson, *NAMMU (Release 6.1) User Guide*, AEA-D&R 0472.
[14] M. C. Hill, *Solving groundwater flow problems by conjugate-gradient methods and the strongly implicit procedure.*, Water resources research, 26 (1990), pp. 1961–1969.
[15] C. Johnson, *Numerical solution of Partial Differential Equations by the Finite Element method*, CUP, 1987.
[16] R. Kettler, *Analysis and comparison of relaxation schemes in robust multigrid and preconditioned conjugate gradient methods*, in Multigrid Methods, 1981.
[17] A. Ramage and A. Wathen, *On preconditioning for finite element equations on irregular grids.* To appear SIAM J. Matrix Anal. and Appl., 1993.

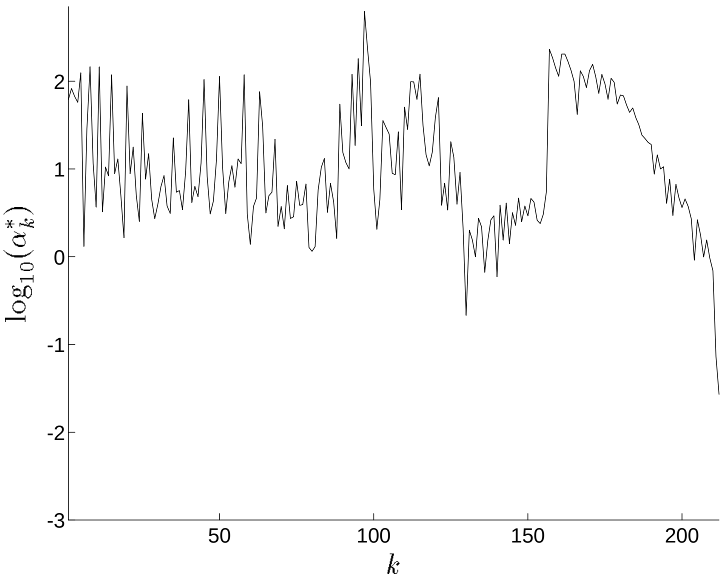

Fig. 1. *Plot of maximum components, α_k^* (see (3.26)), in the error expansion (3.22). Note that $1 \le k \le 212$ corresponds to a fine grid with $N_f = 1056$.*

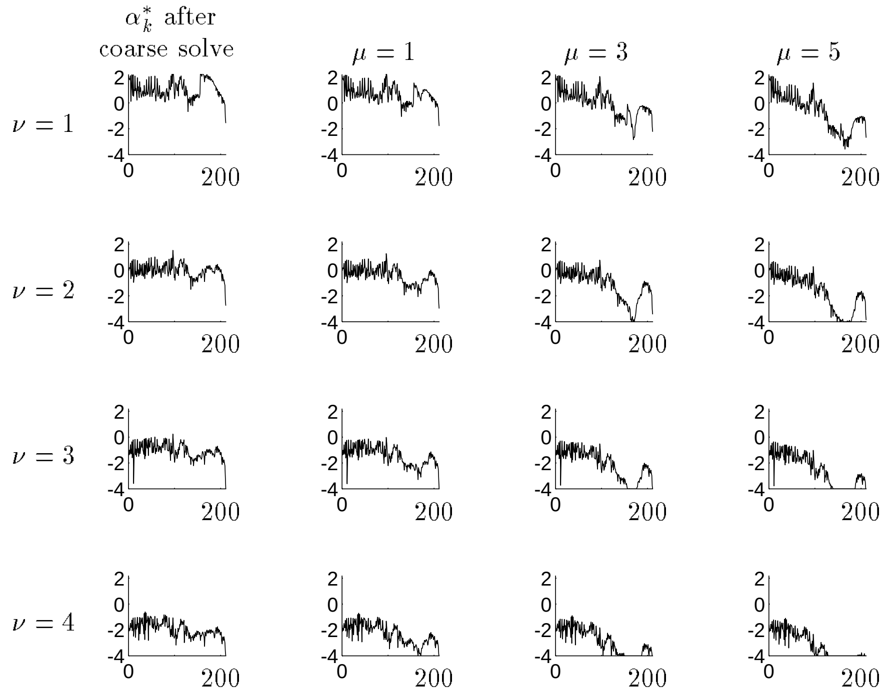

FIG. 2. *Computational Modal Analysis with Richardson Smoothing, with axes as in figure 1.*

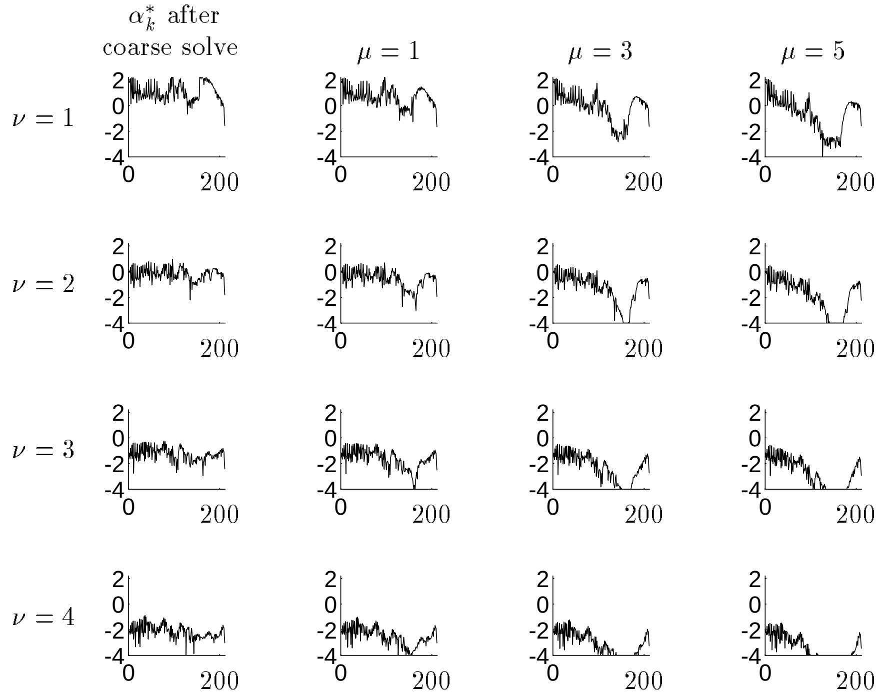

FIG. 3. *Computational Modal Analysis with Gradient Smoothing, with axes as in figure 1.*

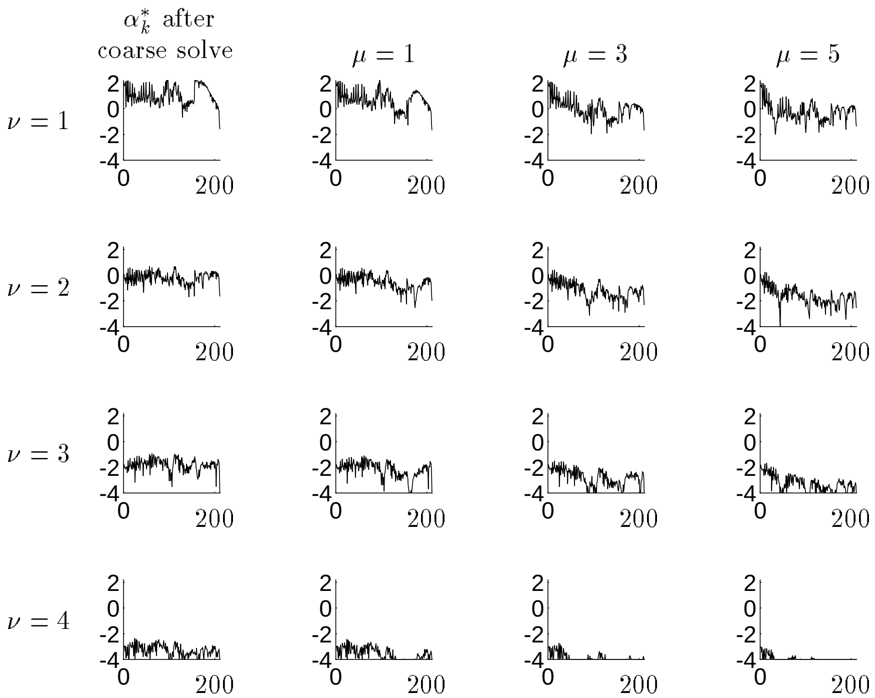

FIG. 4. *Computational Modal Analysis with Conjugate Gradient Smoothing., with axes as in figure 1*

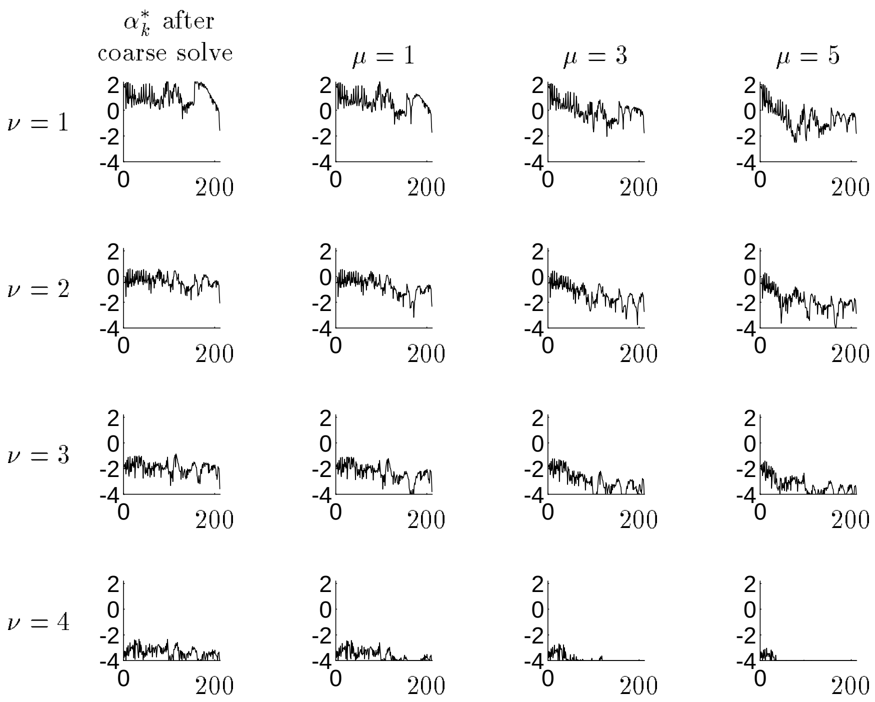

FIG. 5. *Computational Modal Analysis with Conjugate Residual Smoothing, with axes as in figure 1.*

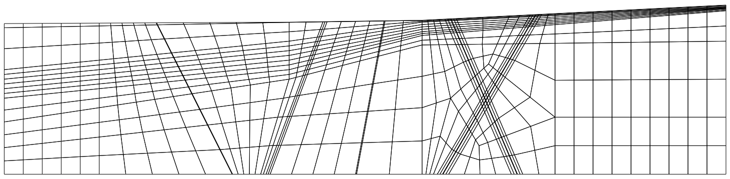

FIG. 6. *Sample grid for realistic problems.*

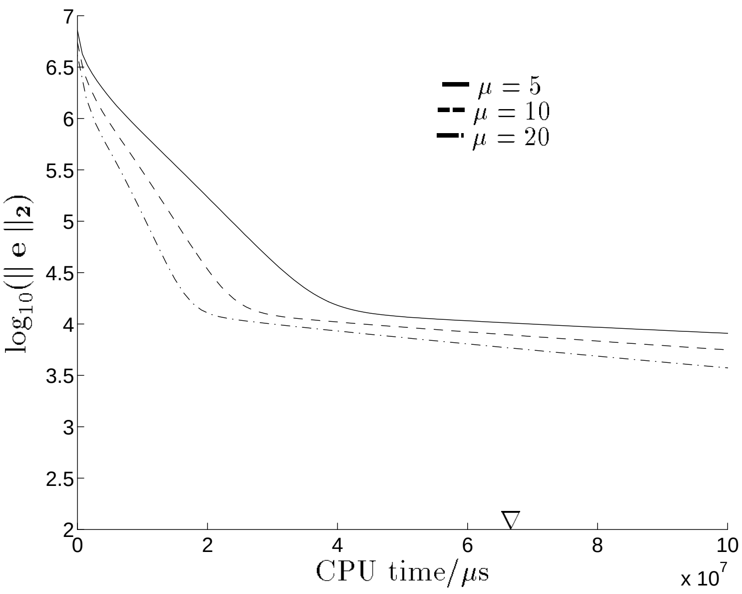

FIG. 7. *Problem 1: Richardson as smoother.*

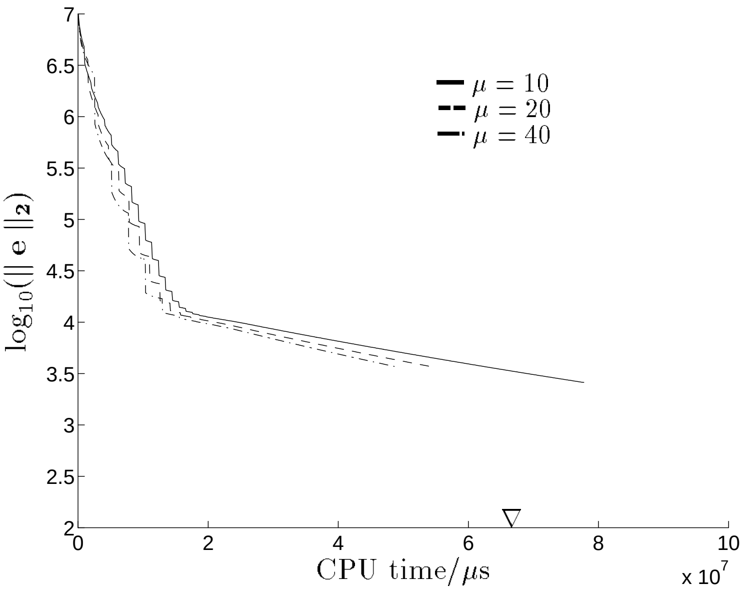

FIG. 8. *Problem 1: Gradient method as smoother.*

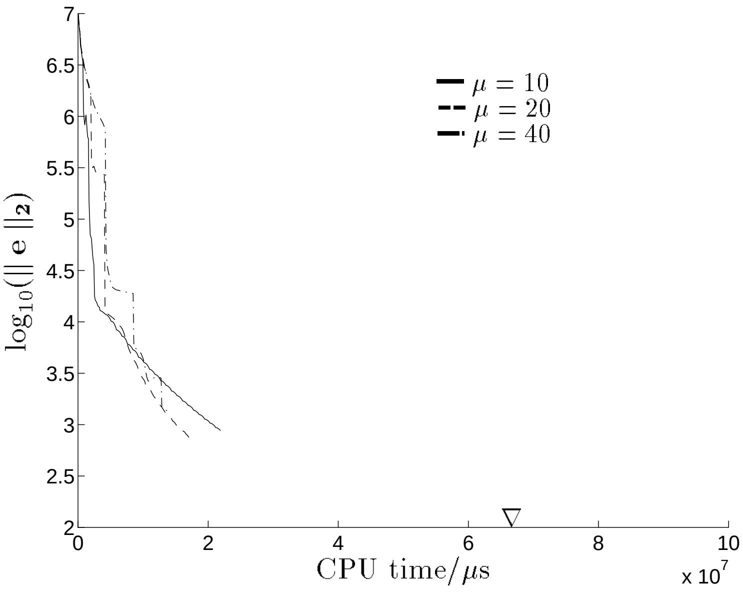

FIG. 9. *Problem 1: Conjugate gradient method as smoother.*

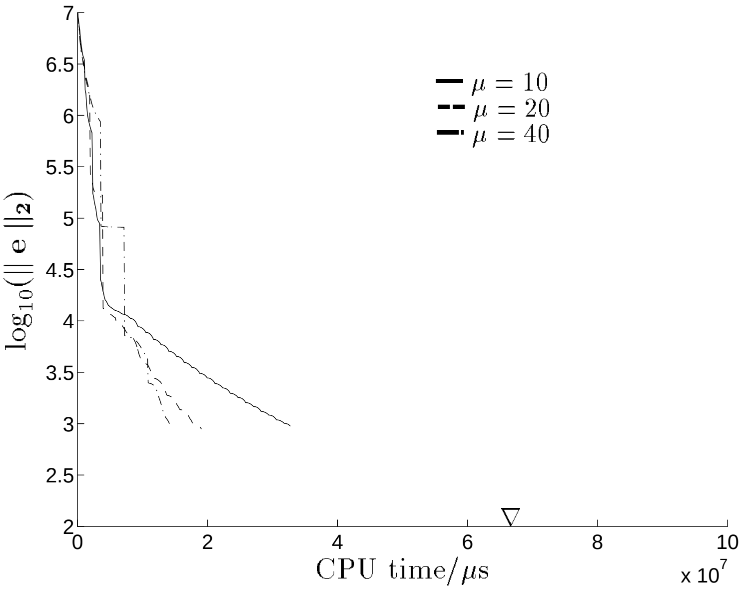

FIG. 10. *Problem 1: Conjugate residual method as smoother.*

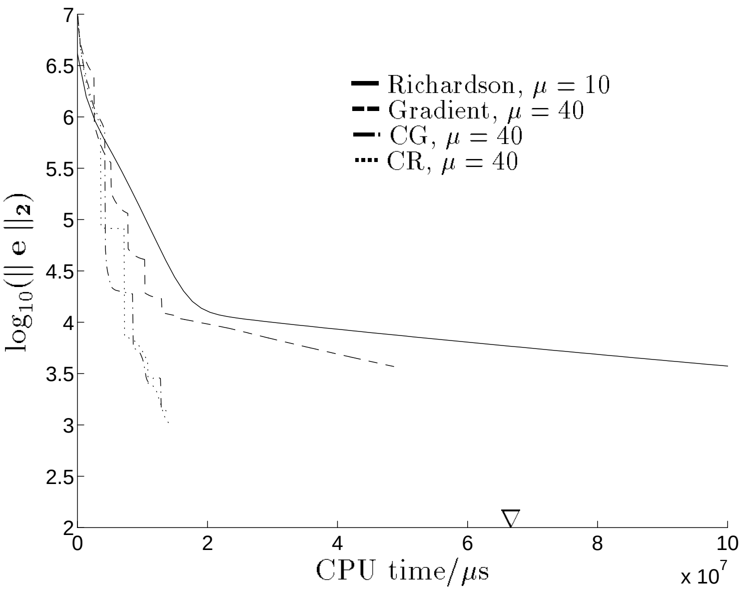

FIG. 11. *Problem 1: "Best" results from each smoother.*

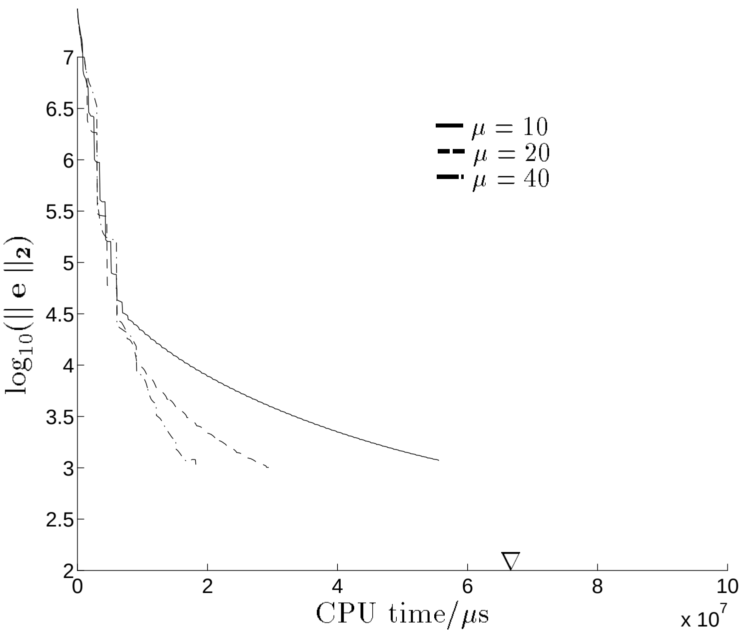

FIG. 12. *Problem 2: Conjugate Gradients as Smoother.*

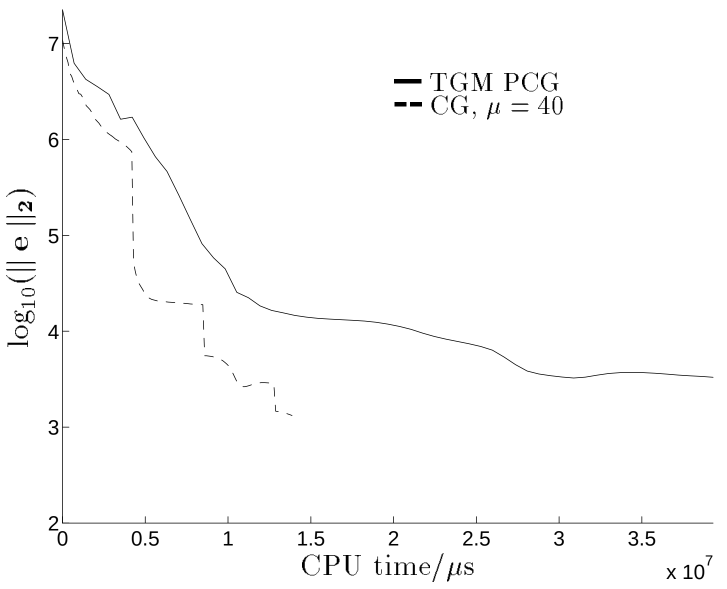

FIG. 13. *Problem 1: Comparison of TGM preconditioning and CG smoothing.*

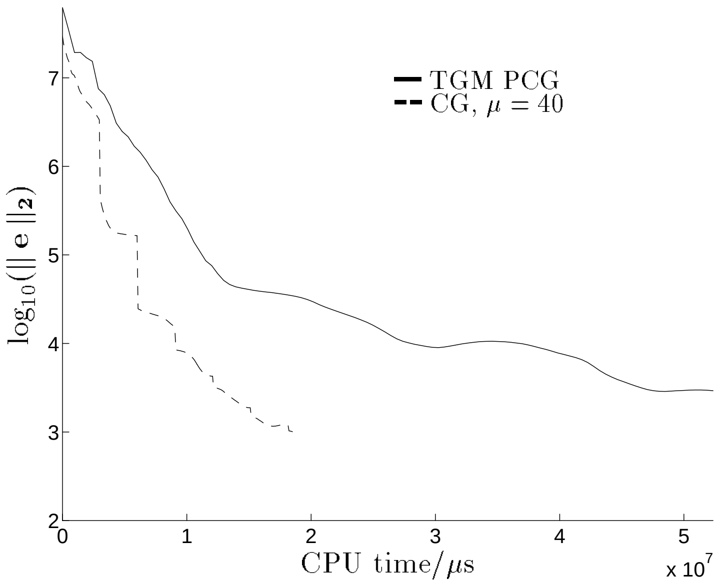

FIG. 14. *Problem 2: Comparison of TGM preconditioning and CG smoothing.*

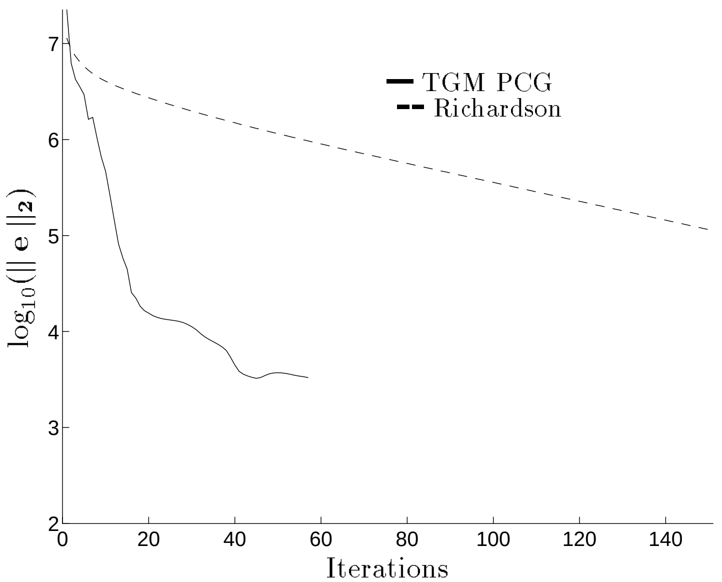

FIG. 15. *Problem 1: Comparison of TGM preconditioned CG and Richardson smoothing.*