Generalized Aggregation Multilevel Solver

J. Fish and V. Belsky
Department of Civil Engineering and Scientific Computation Research Center
Rensselaer Polytechnic Institute, Troy, NY 12180

ABSTRACT

The paper presents a Generalized Aggregation Multilevel (GAM) solver, which automatically constructs nearly optimal auxiliary coarse models based on the information available in the source grid only. GAM solver is a hybrid solution scheme where approximation space of each aggregate (group of neighboring elements) is adaptively and automatically selected depending on the spectral characteristics of individual aggregates. Adaptive features include automated construction of auxiliary aggregated model by tracing "stiff" and "soft" elements, adaptive selection of intergrid transfer operators, and adaptive smoothing.

An obstacle test consisting of nine industry problems, such as ring-strut-ring structure, casting setup in airfoil, nozzle for turbines, turbine blade and diffuser casing as well as on poor conditioned shell problems, such as High Speed Civil Transport, automobile body and canoe, was designed to test the performance of GAM solver. Comparison to the state of the art direct and iterative (PCG with Incomplete Cholesky preconditioner) is carried out. Numerical experiments indicate that GAM solver possesses an optimal rate of convergence by which the CPU time grows linearly with the problem size, and at the same time, robustness is not compromised, as its performance is almost insensitive to problem conditioning.

1.0 Introduction

The performance of linear solvers in terms of CPU time for symmetric positive definite systems can be approximated as CN^{β}, where N is the number of degrees-of-freedom, and C, β are solution method dependent parameters. The major advantage of direct solvers is their robustness, which is manifested by the fact that parameters C and β are independent of problem conditioning (except for close to singular systems). Direct solvers are ideal for solving small up to medium size problems since the constant C for direct methods is significantly smaller than for iterative solvers, but becomes prohibitively expensive for large scale problems since the value of exponent for direct solvers is higher than for iterative methods. To make direct solvers more efficient various modifications of Gaussian elimination, which store and compute only the logical nonzeros of the factor matrix [1], have been developed. Nevertheless, fill-inn cannot be avoided but only minimized and serious consideration of iterative methods for large problems is a virtual reality.

Recent years saw a re-emergence of iterative solvers in finite element structural analysis due to increasing demand to analyze very large finite element systems. Conjugate Gradient method with a single level preconditioner, such as SSOR, Modified Incomplete Cholesky (MIC), Element-by-element (EBE), is considered by many commercial finite element code developers (ANSYS, COSMOS, ALGOR) as most suitable for commercial applications. The value of exponent β for CG type methods with a single level preconditioner typically ranges between 1.17 to 1.33 [2] depending on the preconditioner, while the value of constant C increases with degradation in problem conditioning.

Since the pioneering work of Fedorenko [3], multigrid literature has grown at an astonishing rate. This is not surprising since the multigrid-like methods possess an optimal rate of convergence among the iterative techniques β=1, i.e. computational work required to obtain fixed accuracy is proportional to the number of discrete unknowns. The principal idea of multigrid consists of capturing the oscillatory response of the system by means of smoothing, whereas remaining lower frequency response is resolved on the auxiliary coarse grid. Nevertheless, multigrid methods (or multigrid preconditioners within the CG method) thus far had only very little impact in computational structural analysis. There seem to be two basic reasons:

(i) Commercial software packages must be able to automatically produce a full sequence of auxiliary discretizations (finite element or boundary element meshes) that are gradual coarsenings of the source discretization.

(ii) For optimal multigrid convergence smooth solution components relative to a given discretization must be well approximated by subsequent coarser grids. Conventional or geometric multigrid method cannot guarantee that a sequence of auxiliary discretizations will possess this approximation property for general structural mechanics applications. For example, what is a good coarse discretization for frame structure or a wing structure, where each panel in the source mesh consists of a single or very few shell element.

These difficulties motivated the development of *Algebraic Multigrid* (AMG*)* [4] with the intent of providing a black box algebraic solver based on multigrid principles and exhibiting multigrid efficiency. While geometric multigrid approach constructs discretization sequence using auxiliary courser grids, AMG accomplish the same goal on the basis of the information available in the source matrix of equations only. By this technique the coarse level variables selected so as to satisfy certain criteria based on the source grid matrix. The most basic criterion is typically that each fine level degree-of-freedom should be strongly connected to some course level variable. However, the fact that algebraic multigrid uses information available in the source matrix only in constructing auxiliary discretizations, results in suboptimal rate of convergence.

The *aggregation based multilevel solver* is a hybrid scheme where some minor extra information (depending on the type of aggregation scheme) might be used to construct a hierarchy of coarser problems, but no sequence of coarser discretization is required. The concept of aggregation has been introduced by Leontief in 1951 [5] in the context input-output economics, where commodities in large scale systems where aggregated to produce smaller systems.

The concept of aggregation has been utilized within the context of the multigrid method by Bulgakov [6], [7] and Vanek [8]. In [6] aggregates consisting of non-intersecting groups of neighboring nodes were chosen to have translational degrees of freedom only, and consequently, the auxiliary coarse model could be constructed without knowledge of nodal coordinates or eigenvalue analysis. On the negative side, convergence was only guaranteed for scalar problems such as heat conduction. This algorithm has been improved in [7] by enriching the kinematics of the aggregate with rotational degrees of freedom (three in 3D, one in 2D) and constructing the prolongation operator on the basis of nodal coordinates. In general this approach does not guarantee that the coarse model captures the entire null space of the aggregate, such as in the case of pinned connections in frames or continuum problems where, for example, elements within an aggregate are connected at a single node. Furthermore, the convergence characteristics of this approach have been found to be not satisfactory for poor conditioned problems. These drawbacks motivated development of smoothed aggregation concept [8]. By this technique a tentative piecewise interpolation field consisting of a null space of individual aggregates is first defined and then corrected using Jacobi smoother in attempt to reduce the energy of coarse space basis functions. Our numerical experiments indicate that although smoothed aggregation markedly improves the rate of convergence in well conditioned continuum problems, computational efficiency in poor conditioned problems such as thin shell is not improved and in some cases degrades.

In the earlier aggregation schemes [7], [8] a typical coarsening ratio was about 3^{nsd} for Laplace operator, where nsd is a number of space dimensions. For well conditioned problems this is a nearly optimal ratio resulting in methods with remarkably low computational complexity. Unfortunately, for poor conditioned systems such as thin shells, the coarse problem fails to adequately capture the lower frequency response of the source problem. In attempt to develop a solution procedure possessing an optimal rate of convergence where CPU grows linearly with the problem size without compromising on robustness in the sense that the number of iterations is insensitive to problem conditioning, the present paper presents a generalization of the basic aggregation method by which approximation space of each aggregate is adaptively and automatically selected depending on the spectral characteristics of the aggregate.

The paper is organized as follows. Section 2 reviews the basic multigrid concepts. Mathematical foundation of the Generalized Aggregation Multilevel (GAM) method is given in Section 3. Adaptive features including automated construction of aggregated model by tracing “stiff” and “soft” elements, adaptive selection of intergrid transfer operators, and the Incomplete Cholesky based smoothing procedure with adaptive fill-in are described in Section 4. In Section 5 we conduct numerical studies on 3D industry problems, such as ring-strut-ring structure, casting setup in airfoil, nozzle for turbines, turbine blade and diffuser casing as well as on poor conditioned shell problems, such as High Speed Civil Transport (HSCT), canoe and automobile body. Comparisons to the state of the art direct [1] and iterative (PCG with Incomplete Cholesky preconditioner, Power Solver of ANSYS) are also included in Section 5.

2.0 Multigrid Principles

Consider a linear or linearized system of equations within a Newton-Raphson or related scheme

$$\boldsymbol{K}\boldsymbol{u} = \boldsymbol{f} \qquad \boldsymbol{u} \in \boldsymbol{R}^n \qquad \boldsymbol{f} \in \boldsymbol{R}^n \tag{1}$$

where $\boldsymbol{K}$ is *nxn* symmetric positive definite and sparse matrix.

The following notation is adopted. Auxiliary grid functions are denoted with subscript 0. For example, $\boldsymbol{u}_0$ denotes the nodal values of the solution in the auxiliary grid, where $\boldsymbol{u}_0 \in \boldsymbol{R}^m, m < n$. We also denote the prolongation operator from the coarse grid to the fine grid by $\boldsymbol{Q}$:

$$\boldsymbol{Q}:\boldsymbol{R}^m \to \boldsymbol{R}^n \tag{2}$$

The restriction operator $\boldsymbol{Q}^T$ from the fine-to-coarse grid is conjugated with the prolongation operator, i.e.:

$$\boldsymbol{Q}^T:\boldsymbol{R}^n \to \boldsymbol{R}^m \tag{3}$$

In this section superscripts are reserved to indicate the iteration count. Let $\boldsymbol{r}^i$ be the residual vector in the i -th iteration defined by

$$\boldsymbol{r}^i = \boldsymbol{f} - \boldsymbol{K}\boldsymbol{u}^i \tag{4}$$

where $\boldsymbol{u}^i$ - is the current approximation of the solution in the i -th iteration.

The problem of the coarse grid correction for positive definite systems consists of the minimization of the energy functional on the subspace $\boldsymbol{R}^m$, i.e.:

$$\frac{1}{2}(\boldsymbol{K}(\boldsymbol{u}^i + \boldsymbol{Q}\boldsymbol{u}_0^i), \boldsymbol{u}^i + \boldsymbol{Q}\boldsymbol{u}_0^i) - (\boldsymbol{f}, \boldsymbol{u}^i + \boldsymbol{Q}\boldsymbol{u}_0^i) \Rightarrow min \quad \boldsymbol{u}_0^i \in \boldsymbol{R}^m \tag{5}$$

where (.,.) denotes the bilinear form defined by

$$(\boldsymbol{u}, \boldsymbol{v}) = \sum_{j=1}^{n} u_j v_j \qquad \boldsymbol{u}, \boldsymbol{v} \in \boldsymbol{R}^n \tag{6}$$

A direct solution of the minimization problem (5) yields a classical two-grid procedure. Alternatively, one may introduce an additional auxiliary grid for $\boldsymbol{u}_0$ and so forth, leading to a natural multi-grid sequence. To fix ideas we will consider a two-grid process resulting from the direct minimization of (5) which yields

$$\boldsymbol{K}_0\boldsymbol{u}_0^i = \boldsymbol{Q}^T(\boldsymbol{f} - \boldsymbol{K}\boldsymbol{u}^i) \tag{7}$$

where $\boldsymbol{K}_0 = \boldsymbol{Q}^T\boldsymbol{K}\boldsymbol{Q}$ -is the restriction of the matrix $\boldsymbol{K}$. The resulting classical two-grid algorithm can be viewed as a two-step procedure:

a) *Coarse grid correction*

$$\begin{aligned} \boldsymbol{r}^i &= \boldsymbol{f} - \boldsymbol{K}\boldsymbol{u}^i \\ \boldsymbol{u}_0^i &= \boldsymbol{K}_0^{-1}\boldsymbol{Q}^T\boldsymbol{r}^i \\ \tilde{\boldsymbol{u}}^i &= \boldsymbol{u}^i + \boldsymbol{Q}\boldsymbol{u}_0^i \end{aligned} \tag{8}$$

where $\tilde{\boldsymbol{u}}^l$ is a partial solution obtained after the coarse grid correction.

b) *Smoothing*

$$\boldsymbol{u}^{i+1} = \tilde{\boldsymbol{u}}^i + \boldsymbol{D}^{-1}(\boldsymbol{f} - \boldsymbol{K}\tilde{\boldsymbol{u}}^i) \tag{9}$$

where $\boldsymbol{D}$ is a preconditioner for smoothing. Any preconditioned iterative procedure which has good smoothing properties and requires little computational work per iteration step, can in principle, be used as a smoother in the multigrid process. In particular, various incomplete factorizations, have been found to possess good smoothing characteristics.

Let $\boldsymbol{u}$ be the exact solution of the source problem, then the error resulting from the coarse grid correction (8) can be cast into the following form

$$\tilde{\boldsymbol{e}}^i = \boldsymbol{u} - \tilde{\boldsymbol{u}}^i = (\boldsymbol{I} - \boldsymbol{Q}\boldsymbol{K}_0^{-1}\boldsymbol{Q}^T\boldsymbol{K})\boldsymbol{e}^i = (\boldsymbol{I} - \boldsymbol{C}^{-1}\boldsymbol{K})\boldsymbol{e}^i \tag{10}$$

where I is the identity $n \times n$ matrix and $\boldsymbol{C}^{-1} = \boldsymbol{Q}\boldsymbol{K}_0^{-1}\boldsymbol{Q}^T$ is a course grid preconditioner. Likewise the influence of smoothing on error reduction is given by:

$$\boldsymbol{e}^{i+1} = \boldsymbol{u} - \boldsymbol{u}^{i+1} = (\boldsymbol{I} - \boldsymbol{D}^{-1}\boldsymbol{K})\tilde{\boldsymbol{e}}^i \tag{11}$$

and from the equations (9), (10) the error vector of the two-grid process with one post-smoothing iteration can be expressed as:

$$\boldsymbol{e}^{i+1} = (\boldsymbol{I} - \boldsymbol{D}^{-1}\boldsymbol{K})(\boldsymbol{I} - \boldsymbol{C}^{-1}\boldsymbol{K})\boldsymbol{e}^i \tag{12}$$

Further denoting

$$\begin{aligned} \boldsymbol{M} &= \boldsymbol{I} - \boldsymbol{D}^{-1}\boldsymbol{K} \\ \boldsymbol{T} &= \boldsymbol{I} - \boldsymbol{Q}\boldsymbol{K}_0^{-1}\boldsymbol{Q}^T\boldsymbol{K} \end{aligned} \tag{13}$$

equation (12) with ν_1 post-smoothing and one ν_2 pre-smoothing iteration can be cast into in the following concise form

$$\boldsymbol{e}^{i+1} = \boldsymbol{M}^{\nu_1}\boldsymbol{T}\boldsymbol{M}^{\nu_2}\boldsymbol{e}^i \tag{14}$$

Based on equation (13) it can be easily shown that $\boldsymbol{T}$ is a projection operator, i.e. $\boldsymbol{T} = \boldsymbol{T}^2$, and hence $\|\boldsymbol{T}\|_K = 1$.

Equation (14) represents the sufficiency condition for the convergence of multigrid method provided that the iterative procedure employed for smoothing is convergent, i.e.

$\|\boldsymbol{M}\|_K < 1$. For recent advances on convergence analysis for multigrid like methods we refer to [9].

In practice, however, solution increment $\boldsymbol{u}^{k+1} - \boldsymbol{u}^{k} = \boldsymbol{P}^{-1}\boldsymbol{r}^{k}$ obtained from the multigrid method is used in the determination of the search direction within the conjugate gradient method. The inverse of the two-grid preconditioner with $\nu_1 = 1$ $\nu_2 = 0$ can be obtained from equation (12)

$$\boldsymbol{P}^{-1} = \boldsymbol{D}^{-1}(\boldsymbol{I} - \boldsymbol{K}\boldsymbol{C}^{-1}) + \boldsymbol{C}^{-1} \tag{15}$$

for which the closed form direct expression is given as

$$\boldsymbol{P} = \left[\boldsymbol{I} + (\boldsymbol{K} - \boldsymbol{D})\boldsymbol{Q}(\boldsymbol{Q}^{T}\boldsymbol{D}\boldsymbol{Q})^{-1}\boldsymbol{Q}^{T}\right]\boldsymbol{D} \quad . \tag{16}$$

3.0 Generalized Aggregation Multilevel (GAM) Solver

In aggregation scheme the coarse model is directly constructed from the source grid by decomposing the whole set of nodes into non-intersecting groups to be referred to as aggregates, and then for each aggregate assigning a reduced number of degrees of freedom. By doing so one reduces dimensionality of the source problem, while maintaining compatibility of the solution. The key issue is how to approximate the solution on each aggregate so that the coarse model, to be referred to as an aggregated model, will effectively capture the lower frequency response of the source system.

We start by relating (*Assertion 1*) the optimal characteristics of the aggregated mesh to the intergrid transfer operator properties of individual aggregates and interface regions between the aggregates.

Assertion 1:

The prolongation operator $\boldsymbol{Q}:\boldsymbol{R}^m \rightarrow \boldsymbol{R}^n$ is considered optimal for fixed $m \le n$ if $\left\|\boldsymbol{Q}^T\boldsymbol{K}\boldsymbol{Q}\right\|_2$ is minimal for all $\boldsymbol{Q}$ satisfying $\|\boldsymbol{Q}\|_2 = 1$ and $rank\,\boldsymbol{Q} = m$. Furthermore, among all the block diagonal prolongation operators, where each block corresponds to the prolongation operator of individual aggregate, the optimal prolongation operator is such that

$$\max_{e,a}\left\{\left\|\boldsymbol{K}_0^a\right\|_2, \left\|\boldsymbol{K}_0^e\right\|_2\right\} \Rightarrow min(\boldsymbol{Q}^a)$$
$$Subjected\quad to \qquad \left\|\boldsymbol{Q}^a\right\|_2 = 1 \qquad rank\,\boldsymbol{Q}^a = m_a \qquad \forall a \in G \tag{17}$$

where superscripts *a* and *e* denote the aggregates and interface elements between the aggregates, respectively. N_A and N_E is the total number of aggregates and interface ele-

ments, respectively*;* $\boldsymbol{Q}^a : \boldsymbol{R}^{m_a} \rightarrow \boldsymbol{R}^{n_a}$ and $\boldsymbol{Q}^e : \boldsymbol{R}^{m_e} \rightarrow \boldsymbol{R}^{n_e}$ are the prolongation operators for aggregate a and interface element *e;* $\boldsymbol{K}_0^a = (\boldsymbol{Q}^a)^T \boldsymbol{K}^a \boldsymbol{Q}^a$ and $\boldsymbol{K}_0^e = (\boldsymbol{Q}^e)^T \boldsymbol{K}^e \boldsymbol{Q}^e$ are the corresponding restricted stiffness matrices.
Note that minimization is carried out with respect to the prolongation operator for the aggregates only, that the prolongation operator $\boldsymbol{Q}^e$ for each element in the interface region is uniquely determined from $\{\boldsymbol{Q}^a\}_{a=1}^{N_A}$.

<u>*Proof:*</u>

Let Φ and Λ be a $n x n$ matrix of unitary eigenvectors and a diagonal $n x n$ matrix of eigenvalues of the stiffness matrix $\boldsymbol{K}$, respectively, partitioned as $\begin{bmatrix} \Phi_0 & \Phi_1 \end{bmatrix}$ and $\begin{bmatrix} \Lambda_0 & 0 \\ 0 & \Lambda_1 \end{bmatrix}$ so that $\|\Lambda_0\|_2 < \tilde{\gamma}$ and $\boldsymbol{Q} = \Phi_0 \alpha$ where Φ_0 consists of m unitary eigenvectors and $\Phi^T \Phi = \boldsymbol{I}$.

The spectral norm of stiffness matrix of the auxiliary model $\|\boldsymbol{K}_0\|_2$ can be bounded utilizing consistency condition [11]

$$\|\boldsymbol{K}_0\|_2 = \|\boldsymbol{Q}^T \boldsymbol{K} \boldsymbol{Q}\|_2 = \|\alpha^T \Lambda_0 \alpha\|_2 \le \|\alpha\|_2^2 \|\Lambda_0\|_2 \tag{18}$$

Furthermore, since $\|\boldsymbol{Q}\|_2 = \|\boldsymbol{Q}^T \boldsymbol{Q}\|_2^{1/2} = \|\alpha^T \Phi_0^T \Phi_0 \alpha\|_2^{1/2} = \|\alpha\|_2 = 1$ we obtain

$$\|\boldsymbol{K}_0\|_2 \le \tilde{\gamma} \tag{19}$$

which completes the first part of theorem. For the second part we bound the maximum eigenvalue of the system [10] by the maximum eigenvalue of the subdomain (aggregate or interface element)

$$\|\boldsymbol{K}_0\|_2 \le \max_{e,a} \left\{ \|\boldsymbol{K}_0^a\|_2, \|\boldsymbol{K}_0^e\|_2 \right\} \tag{20}$$

Assertion 1 states that the quality of aggregated model is governed by the maximum spectral radius of individual subdomains. The next assertion formulates certain minimum requirements for the construction of $\boldsymbol{Q}^a$ aimed at ensuring the lower bound of the minimal eigenvalue of the two-grid preconditioned system $\boldsymbol{P}^{-1}\boldsymbol{K}$. It assumes the worse case scenario where smoothing does not affect lower frequency response errors.

Assertion 2:

Consider the two-level method with $\nu_1 = 1 \quad \nu_2 = 0$ and smoothing affecting only high frequency modes of error. Then the lower frequency response of the two-level system characterized by the lower bound of Rayleigh quotient $\rho(\boldsymbol{x}) = (\boldsymbol{x}^T\boldsymbol{K}\boldsymbol{x})/(\boldsymbol{x}^T\boldsymbol{P}\boldsymbol{x})$ is governed by the lowest eigenvalue among all the aggregates provided that the prolongation operator of each aggregate is spanned by the space, which at a minimum contains the null space of that aggregate.

Proof:

Let ϕ^a and λ^a be n_a eigenvectors and eigenvalues of the aggregate a. Nodal solution u^a on each aggregate can be expressed as a linear combination of its eigenvectors

$$\boldsymbol{u}^a = \boldsymbol{\phi}^a\hat{\boldsymbol{u}}^a \tag{21}$$

whereas global solution vector, denoted as $\boldsymbol{u} = \boldsymbol{\phi}\hat{\boldsymbol{u}}$ can be assembled from its aggregates. Let $\begin{bmatrix}\boldsymbol{\phi}_0 & \boldsymbol{\phi}_1\end{bmatrix}$ and $\begin{bmatrix}\boldsymbol{\lambda}_0 & \boldsymbol{0} \\ \boldsymbol{0} & \boldsymbol{\lambda}_1\end{bmatrix}$ be the partitions of ϕ and λ, respectively, such that $\|\lambda_0\|_2 < \hat{\gamma}$. Then the system of equations can be transformed into hierarchical form:

$$\begin{bmatrix}\hat{\boldsymbol{K}}_{00} & \hat{\boldsymbol{K}}_{01} \\ \hat{\boldsymbol{K}}_{10} & \hat{\boldsymbol{K}}_{11}\end{bmatrix}\begin{bmatrix}\hat{\boldsymbol{u}}_0 \\ \hat{\boldsymbol{u}}_1\end{bmatrix} = \begin{bmatrix}\hat{\boldsymbol{f}}_0 \\ \hat{\boldsymbol{f}}_1\end{bmatrix} \tag{22}$$

where

$$\hat{\boldsymbol{u}}_i = \boldsymbol{\phi}_i^T\boldsymbol{u}_i \qquad \hat{\boldsymbol{f}}_i = \phi_i^T\boldsymbol{f}_i \qquad \hat{\boldsymbol{K}}_{ij} = \phi_i^T\boldsymbol{K}\boldsymbol{\phi}_j \qquad i, j = 0, 1 \tag{23}$$

Similarly, any smoothing preconditioner $\boldsymbol{D}$ can be transformed into hierarchical form $\hat{\boldsymbol{D}}$ as follows

$$\hat{\boldsymbol{D}}_{ij} = \phi_i^T\boldsymbol{D}\phi_j \tag{24}$$

Let $\boldsymbol{Q} = \phi_0$, then the prolongation operator $\hat{Q}$ defined in hierarchical basis is given

$$\hat{\boldsymbol{Q}} = \begin{bmatrix}\phi_0^T \\ \phi_1^T\end{bmatrix} Q = \begin{bmatrix}\boldsymbol{I} \\ \boldsymbol{0}\end{bmatrix} \tag{25}$$

Substituting equations (22)-(25) into (16) yields the two grid preconditioner $\hat{\boldsymbol{P}}$ ($\nu_1 = 1 \quad \nu_2 = 0$) defined in the hierarchical basis

$$\hat{\boldsymbol{P}} = \begin{bmatrix} \hat{\boldsymbol{K}}_{00} & 0 \\ \hat{\boldsymbol{K}}_{10} & \hat{\boldsymbol{D}}_{11} \end{bmatrix} + \begin{bmatrix} 0 & \hat{\boldsymbol{K}}_{00} \\ 0 & \hat{\boldsymbol{K}}_{10} - \hat{\boldsymbol{D}}_{10} \end{bmatrix} \hat{\boldsymbol{D}}_{00}^{-1} \hat{\boldsymbol{D}}_{01} \tag{26}$$

If we further assume that smoothing affects higher frequency response only in the sense that

$$\hat{\boldsymbol{D}}_{00}^{-1} \hat{\boldsymbol{u}}_0 = 0 \qquad \forall \hat{\boldsymbol{u}}_0 \tag{27}$$

then the resulting two-level preconditioner can be cast into the block Gauss-Seidel form:

$$\hat{P} = \begin{bmatrix} \hat{\boldsymbol{K}}_{00} & 0 \\ \hat{\boldsymbol{K}}_{10} & \hat{\boldsymbol{D}}_{11} \end{bmatrix} \tag{28}$$

To estimate the lower bound of Rayleigh quotient of the two-level preconditioned system we utilize again the theorem that bounds the lower eigenvalue of the system (aggregated model) by minimum eigenvalue of any subdomain i consisting of either aggregates a or interface elements e:

$$\min_{i, \hat{\boldsymbol{x}}} \{ \rho^i (\hat{\boldsymbol{x}}^i) \} \le \min_{\boldsymbol{x}} \{ \rho (\hat{\boldsymbol{x}}) \} \tag{29}$$

It remains to examine under which condition Rayleigh quotient on each aggregate or interface element is bounded from below. For the two-grid preconditioner given in (28) the Rayleigh quotient for each aggregate or interface element is given as

$$\rho^i (\hat{\boldsymbol{x}}^i) = \frac{(\hat{\boldsymbol{x}}_0^i)^T \hat{\boldsymbol{K}}_{00}^i \hat{\boldsymbol{x}}_0^i + 2 (\hat{\boldsymbol{x}}_0^i)^T \hat{\boldsymbol{K}}_{01}^i \hat{\boldsymbol{x}}_1^i + (\hat{\boldsymbol{x}}_1^i)^T \hat{\boldsymbol{K}}_{11}^i \hat{\boldsymbol{x}}_1^i}{(\hat{\boldsymbol{x}}_0^i)^T \hat{\boldsymbol{K}}_{00}^i \hat{\boldsymbol{x}}_0^i + (\hat{\boldsymbol{x}}_0^i)^T \hat{\boldsymbol{K}}_{01}^i x_1^i + (\hat{\boldsymbol{x}}_1^i)^T \hat{\boldsymbol{D}}_{11}^i \hat{\boldsymbol{x}}_1^i} \tag{30}$$

Let $N(\hat{\boldsymbol{K}}_{00}^i)$ be the null space of $\hat{\boldsymbol{K}}_{00}^i$ defined as

$$N(\hat{\boldsymbol{K}}_{00}^i) = \{ (\hat{\boldsymbol{x}}_0^i)^T \hat{\boldsymbol{K}}_{00}^i \hat{\boldsymbol{x}}_0^i = 0 \qquad \forall \hat{\boldsymbol{x}}_0^i \in SPAN(\phi_0^i) \} \tag{31}$$

Then the Rayleigh quotient is bounded from below $\rho^i (\hat{\boldsymbol{x}}^i) > 0$ if $\hat{\boldsymbol{K}}_{00}^i$ contains all the rigid body modes of $\hat{\boldsymbol{K}}^i$, i.e. $N(\hat{\boldsymbol{K}}_{00}^i) = N(\hat{\boldsymbol{K}}^i)$. This condition can be easily satisfied if the prolongation operator for each aggregate is spanned by the space containing the rigid body modes of that aggregate. In addition, for all interface elements $N(\hat{\boldsymbol{K}}_c^e) = \varnothing$, where $\hat{\boldsymbol{K}}_c^e$ is the interface element stiffness matrix constrained along the boundary between interface elements and aggregates. Loosely speaking, each interface element should be connected to aggregates at a number of degrees of freedom greater or equal then the dimension of the null space of that element $dim\left\{N(\hat{\boldsymbol{K}}^e)\right\}$.

So far we have proposed how to assess the quality of intergrid transfer operators (*Assertion 1*) and what are the properties that it should maintain (*Assertion 2*). In the subsequent proposition we describe a heuristic approach, which on the bases of the two assertions, attempts to construct a nearly optimal aggregated model.

Proposition 1:

For given $\{m_a\}_{a=1}^{N_a}$ a nearly optimal aggregation model can be constructed if (i) prolongation operator $\boldsymbol{Q}^a:\boldsymbol{R}^{m_a} \to \boldsymbol{R}^{n_a}$ on aggregate a is spanned by $m_a \le n_a$ eigenvectors corresponding to m_a lowest eigenvalues on aggregate a, where $m_a \ge dim\left\{N(\hat{K}^a)\right\}$, and (ii) in forming the aggregated model soft elements determined by the Gerschgorin upper bound of their maximal eigenvalue $\max_i\left(\sum_j \left|k_{ij}^e\right|\right)$ are placed at the interface, where $\boldsymbol{K}^e = [k_{ij}^e]$.

Discussion:

We first show that for fixed m_a the prolongation operators $\boldsymbol{Q}^a$ that minimizes $\left\|\boldsymbol{K}_0^a\right\|_2$ is obtained as a linear combination of m_a lowest eigenvectors of $\boldsymbol{K}^a$. This fact directly follows from equation (19) in the context of individual aggregates

$$\left\|\boldsymbol{K}_0^a\right\|_2 \le \gamma^a \qquad \forall\, \boldsymbol{Q}^a = \left\{\left\|\boldsymbol{Q}^a\right\|_2 = 1 \qquad rank\,\boldsymbol{Q}^a = m_a\right\} \tag{32}$$

where γ^a is the maximum eigenvalue of eigenvectors spanning the space of $\boldsymbol{Q}^a$. Furthermore, if we select $\gamma^a \le \gamma\ \forall a$, then the spectral norm of individual aggregates does not exceed user prescribed tolerance γ.

The spectral radius of the restricted interface element stiffness matrix $\boldsymbol{K}_0^e$ is given as

$$\left\| \boldsymbol{K}_0^e \right\|_2 = \left\| (Q^e)^T \boldsymbol{K}^e Q^e \right\|_2 \leq \left\| Q^e \right\|_2^2 \left\| \boldsymbol{K}^e \right\|_2 \tag{33}$$

Since $\boldsymbol{Q}^e$ is a diagonal block of $\boldsymbol{Q}$ it can be easily shown that $\left\| \boldsymbol{Q}^e \right\|_2 \leq \| \boldsymbol{Q} \|_2 = 1$ and thus using Gerschgorin theorem for the maximal eigenvalue of $\boldsymbol{K}$ follows that

$$\left\| \boldsymbol{K}_0^e \right\|_2 \leq \max_i \left(\sum_j \left| k_{ij}^e \right| \right) \tag{34}$$

where $\boldsymbol{K}^e = [k_{ij}^e]$.

4.0 Adaptive features

This section describes three features of adaptivity built into the Generalized Multilevel Aggregation procedure. Some of the notation in this section differs from that introduced in the previous sections.

First we present the algorithm for automated construction of aggregates on the element-by-element basis as opposed to node-by-node procedure employed in [7,8]. In accordance with this approach it is necessary to determine the rigid-body modes and other low-frequency modes based on the aggregate stiffness matrices. We present two versions of the aggregate formation algorithm: the basic version which utilizes a topological information only, and the adaptive version which in addition to the topological information utilizes elemental stiffness matrices in the process of the aggregated model construction.

The second adaptive feature is related to the selection of parameter γ, which plays a central role in constructing the prolongation operator. This parameter has a direct effect on the restriction of the stiffness matrix, the sparsity pattern of resulting auxiliary stiffness matrix as well as on effectiveness of the auxiliary model to capture the lower frequency response.

Finally, we employ Modified Incomplete Cholesky Factorization for pre- and post-smoothing. The number of fill-ins as well as diagonal-scaling needed to preserve the positive definiteness of the system and to provide the fastest rate of convergence of the iterative process are also determined adaptively.

4.1 Aggregation algorithm

Prior to describing the technical details of the aggregation algorithm we introduce the concept of “stiff” and “soft” elements which is utilized in the process of aggregate formation.

The element is considered “stiff” if the spectral radius of its stiffness matrix is relatively large compared to other elements and vice versa. Following Proposition 1, we will attempt wherever possible to place “soft” elements at the interface between the aggregates, and “stiff” elements within the aggregates. This approach is a counterpart of the idea of “weak” and “strong” nodal connectivity employed in [8] in the context of node-by-node aggregation.

The approximation for the maximum eigenvalue can be easily estimated using Gerschgorin theorem in the context of the element stiffness matrices:

$$\max_a \lambda_e^a \le \beta^e \quad \beta^e = \max_i \left(\sum_j \left| k_{ij}^e \right| \right) \tag{35}$$

In the remaining of this subsection we focus on the aggregation algorithm.

Consider the finite element mesh containing N_E elements and N_N nodes. Let $C(i)$ be the set of nodes belonging to the element E^i :

$$C(i) = \{N^j : N^j \in E^i\} \tag{36}$$

The goal of the aggregation algorithm described below is to construct a set of N_A aggregates denoted as

$$A = \left\{ A^i, i \in [1, N_A] \right\} \tag{37}$$

satisfying the following conditions:

(i). *Element-by-element aggregation*:

$$A = \left\{ E^j, j \in M_E \right\} \tag{38}$$

$M_E^i \subset [1, N_E]$ - set the of element numbers corresponding to the aggregate *i*

(ii). *Disjoint covering*: elements belonging to different aggregates can not be neighbors. Two elements E^i and E^j are considered to be neighbors if

$$C(i) \cap C(j) \neq \varnothing \tag{39}$$

(iii). *Full nodal covering*: each node belongs to some aggregate:

$$\bigcup_{\{i:E^i \in A\}} C(i) = \left\{N^j, j \in [1, N_N]\right\} \tag{40}$$

(iv). *Marking the ‘slave’ nodes and nodes with essential boundary conditions as separate aggregates*: each node containing either essential boundary condition and/or ‘slave’ degree(s)-of-freedom, which depends on so called ‘master’ degree(s)-of-freedom, is considered as an aggregate. Denote the set of such nodes as N_{NB}.

Step 1. Setup.

1.1. For each node $N^j, j = [1, N_n]$ select the elements containing this node:

$$B(j) = \{E^i : N^j \in E^i\} \tag{41}$$

1.2. For each element $E^i, i = [1, N_e]$ select the set of neighboring elements $NE(i)$, that are the elements containing common nodes:

$$NE(i) = \{E^k : E^k \in B(j), j \in C(i)\} \backslash E^i \tag{42}$$

Step 2. Start-up aggregation.

2.1. Define the set of elements NA available for aggregation. These are all the elements which do NOT contain nodes with essential boundary conditions or the ‘slave’ nodes:

$$NA = [1, N_e] \backslash \left\{B(j), N^j \in N_{NB}\right\} \tag{43}$$

2.2. Find the “peripheral” element E^s, that is the element with minimal number of neighbors:

$$s = \underset{i \in NA}{\arg min} |NE(i)| \tag{44}$$

where $|X|$ is a number of elements in the set X. Element E^s is a starting element for the aggregation algorithm.

2.3. Setup:

-the current aggregate counter $i = 1$;

- the set of interface elements $NI = [1, N_e] \backslash NA$, i.e. the elements between different aggregates.

Step 3. Formation of the current aggregate.

Basic aggregation version:

aggregate A^i contains the element E^s and **all** it's available neighbors:

$$A^i = E^s \cup (NE(s) \cap NA) \tag{45}$$

Adaptive aggregation version:

aggregate A^i contains the element E^s and **those** of it's available neighbors which satisfy the relative stiffness condition:

$$A^i = E^s \cup \{E^j \in NE(s) \cap NA, \beta^j \ge \mu\beta^s\} \tag{46}$$

where β^j is a Gerschgorin upper bound on the stiffness matrix maximal eigenvalue of the element E^j, and μ is a coarsening parameter.

Step 4. Update the sets of the interface and available elements.

4.1. Update the set of the interface elements:

$$NI = NI \cup \{(E^k \in NE(j), E^j \in A^i) \cap (E^k \notin A^i)\} \tag{47}$$

4.2. Update the set of the available elements:

$$NA = NA \backslash \{(E^k \in NE(j), E^j \in A^i) \cup A^i\} \tag{48}$$

Step 5. Find the new starting element.

5.1. Form the set of "frontal" elements FR, that are available elements neighboring to the interface elements:

$$FR = \{(E^k \in NE(j), E^j \in NI) \cap NA\} \tag{49}$$

5.2. Basic version: select **arbitrary** new starting element belonging to FR:

$$E^s \in FR \tag{50}$$

Adaptive version: select the **stiffest** new starting element from FR:

$$s = \underset{j:E^j \in FR}{\arg max}(\beta^j) \tag{51}$$

Step 6. Stopping criteria.

If $FR = \varnothing$ then stop; else $i = i + 1$ and repeat *steps 3-6*.

Remark 1. For simplicity we only presented the aggregation algorithm for lower order elements. In the case of higher order elements the "full nodal covering" requirement may not be satisfied at the completion of the algorithm described above. There will be a significant number of nodes belonging to the elements in the interface region giving rise to very large auxiliary coarse model. To further reduce the size of the auxiliary model the same aggregation algorithm is recursively applied for the interface elements only until all the nodes would be covered by some aggregate. This procedure also provides a "cleaning" phase to ensure that all nodes in the source grid are included within one of the aggregates.

Remark 2. A similar scheme can be applied for the p-type discretization with only exception that the aggregates may contain only a single element in order to reduce the aggregate size. Higher order modes in the interface region are treated as indicated in the Remark 1.

Remark 3. The aggregation algorithm described in this section deals with multi-point constrains in the conventional way since the elements containing the "slave" nodes form a separate aggregate. Each multi-point constrain can be represented as follows:

$$x_s = Tx_m \tag{52}$$

where x_s, x_m are the 'slave' and 'master' degrees-of-freedom, respectively; T is a transformation matrix representing the multi-point constrain (MPC) data. In accordance with (52) the vector $x = (x_s, x_m)^T$ can be expressed as:

$$x = \begin{bmatrix} T \\ I \end{bmatrix} x_m \tag{53}$$

Given the decomposition of the element stiffness matrix K_e for elements containing the 'slave' degrees-of-freedom

$$K_e = \begin{bmatrix} K_e^{ss} & K_e^{sm} \\ K_e^{ms} & K_e^{mm} \end{bmatrix} \tag{54}$$

the modified element stiffness matrix $\tilde{K}_e$ corresponding to master degrees-of-freedom only is given by:

$$\tilde{K}_e = T^T K_e^{ss} T + K_e^{ms} T + T^T K_e^{sm} + K_e^{mm} \tag{55}$$

4.2 Adaptive construction of prolongation operator

One of the key issue in the proposed aggregation procedure is a selection of parameter γ. All the eigenvectors of the eigenvalue problem on each aggregate corresponding to the eigenvalues $\lambda^a \le \gamma$ are included within the diagonal block of the global prolongation operator. In order to make this parameter dimensionless the eigenvalue problem on each aggregate is formulated in the following manner:

$$\boldsymbol{K}^a \phi^a = \lambda^a \boldsymbol{D}^a \phi^a \tag{56}$$

where $\boldsymbol{D}^a$ is a diagonal of $\boldsymbol{K}^a$. Typically 6-50 modes are needed to satisfy $\lambda^a \le \gamma$ requirement. Lanczos algorithm with partial orthogonalization [13] was adopted.

The value of the parameter γ determines effectiveness of coarse grid correction. In the limit as $\gamma \to \max_a \lambda^a$ auxiliary problem captures the response of the source system for all frequencies and therefore the two-level procedure converges in a single iteration even without smoothing. On the negative side, for large values of γ, eigenvalue analysis on each aggregate becomes prohibitively expensive and the auxiliary matrix becomes both large and dense. At the other extreme in the limit as $\gamma \to 0$ the prolongation operator contains the rigid body modes of all the aggregates only, and thus auxiliary problem becomes inefficient for ill-posed problems.

4.3 Adaptive smoothing

Selection of smoothing procedure is another important issue as the cost of smoothing is a major expense in multi-level procedures. Comprehensive studies conducted in [12] revealed that one of the most efficient smoothing schemes is based on Modified Incomplete Cholesky factorization (MIC). We employed two versions of MIC, with and without additional fill-ins using “by value” as the fill-in strategy. By this technique one compares the values of the terms in the incomplete factor and chooses the largest ones to be included [14]. One of the most important parameters in both versions is the diagonal-scaling parameter α which insures positive definiteness of the incomplete factor. The value of α is determined adaptively. Its optimal value depends on the number of fill-ins. For larger number of fill-ins the optimal value of the diagonal-scaling decreases. The optimal number of fill-ins is determined experimentally, whereas the value of the diagonal-scaling parameter is determined adaptively by incrementally increasing it until all positive pivots are obtained.

5.0 Numerical examples

An obstacle test as shown in Figures 1 and 2 comprised of the following industry and model problems was designed to (i) determine the optimal values of computational parameters and to (ii) compare GAM solver with existing state-of-the-art solvers:

Diffuser Casing with Gates for Casting: 10 node tetrahedral elements; 131,529 d.o.f.s.

Turbine Blade with Platform: 10 node tetrahedral elements; 207,840 d.o.f.s.

Nozzle for Turbines: 10 node tetrahedral elements; 131,565 d.o.f.s.

Casting Setup for Casting in Airfoil: 10 node tetrahedral elements; 158,166 d.o.f.s.

Concentric Ring-Strut-Ring Structure: 4 node tetrahedral elements; 102,642 d.o.f.s.

High Speed Civil Transport (HSCT): MIN3 [15] shell elements; 88,422 d.o.f.s.

Automobile Body: 3 node DKT+DMT shell [16], 2 node beam elements; 265,128 d.o.f.s.

Automobile Body: MIN3 [15] shell and 2 node beam elements; 265,128 d.o.f.s.

Concrete canoe: 8 node ANS [17] shell elements; 132,486 d.o.f.s.

5.1 Parametric study

In this section we present the results of numerical investigation of the following computation parameters: limiting eigenvalue parameter γ for selection of the modes to be included in the prolongation operator; number of fill-ins and diagonal scaling parameter α for Modified Incomplete Cholesky factorization; and coarsening computation parameter μ.

5.1.1 Prolongation parameters

In order to determine optimal value of γ in terms of the CPU time we have carried out extensive computational experiments for wide range of industrial problems, including well-posed and ill-posed cases. Surprisingly, it has been found that the optimal value of γ is independent of the problem condition and slightly differs for different problems. For example, it can be seen that for poorly conditioned HSCT problem (Figure 3) significant reduction of the number of iterations was observed as γ increased from 0.0020 to 0.0040. Optimal value of γ, which minimizes the CPU time for this problem was equal to 0.0035 independently of quality of MIC smoother (number of fill-ins and diagonal scaling parameter). On the other hand, for the Diffuser Casing (Figure 4), Automobile Body (Figure 6), Concentric Ring-Strut-Ring Structure and Joint of Two Cylinders problems the CPU time was practically independent of γ. However, for the Nozzle for Turbine problem (Figure 5) significant reduction of the number of iterations was observed for relatively large values of γ ranging from 0.0075 to 0.0100 and the optimal value of γ, which minimizes the CPU time for this problem, was equal to 0.0100. Based on these results we have built in $\gamma = 0.0050$ for further numerical studies and comparisons, which provides a reasonably good performance for all problems considered.

5.1.2 Smoothing parameters

The efficiency of MIC based smoothing procedure highly depends on the two computational parameters: the number of fill-ins and the diagonal-scaling. Typically, increasing the number of fill-ins allows to decrease the value of the diagonal-scaling parameter. It can be seen (Figure 3 and Figure 7) that for the HSCT, Diffuser Casing, Concentric Ring-Strut-Ring Structure problems the optimal value of fill-ins is equal to $5-6$, with minimal value of diagonal-scaling parameter α which ensures positive pivots. For the HSCT problem the effect of number of fill-ins and the value α presented in Figure 3 indicated that the optimal computational performance is obtained with 4-6 fill-ins. For the Nozzle for Turbines (Figure 5) and Joint of Two Cylinders problems it was observed that the number of fill-ins has no effect on the effectiveness of the iterative process. We did not consider number of fill-ins greater then 8 due to increased in-core memory requirements.

Based on the computational experiment the following strategy has been developed for determination of nearly optimal values of α and number of fill-ins:

- MIC with number of fill-ins is equal to 6
- Initial diagonal-scaling parameter $\alpha = 0.01$
- Increasing α by the increment of $\Delta\alpha = 0.0025$ if non-positive pivot is encountered in the process of incomplete factorization, or if the two-level iteration procedure diverges.

5.1.3 Aggregation parameters

Numerical experiments in obstacle test indicated that the value of the coarsening parameter μ had very little effect on the convergence of the iteration procedures. The only problem where considerable improvement was observed was a 2-D problem for randomly distributed short fibers in matrix material, where fiber/matrix stiffness ratio was equal to 100. The problem was modeled using quadrilateral finite elements. For this problem converge solution was achieved in 23 iterations using basic aggregation algorithm, while using adaptive version of aggregation procedure with optimal value $\mu = 1.68$ the convergence was achieved in 18 iterations. In subsequent studies we employed the basic version of aggregation procedure.

5.2 Comparison with other solvers and discussion

First we present the comparison of GAM solver with traditional Skyline Direct solver. Figure 9 shows the rate of convergence in term of CPU time versus problem size for the Diffuser Casing with Gates for Casting problem. It can be seen that in contrast to other solvers considered the CPU time grows linearly with problem size for GAM solver. Even for relatively small problem with 35,000 d.o.f.s. GAM outperforms traditional Skyline solver by factor of 27. For the problem with 70,000 d.o.f.s. GAM solver outperforms Sparse Direct solver by factor of 9 and PCG with Modified Incomplete Cholesky preconditioner by factor of 12.

In the second set of problems GAM is compared with “smoothed aggregation” technic introduced in [8]. We have observed that for a 2-D model elasticity problem on a square domain this approach gives an improvement in terms of number of iterations (16 instead of 23). However for ill-posed shell problem (HSCT) the number of iteration becomes almost twice larger (154) in comparison with the basic GAM version. Furthermore, smoothing of the approximation field on each aggregate creates denser prolongation operator, which in turn increases CPU time of restriction and yields denser auxiliary matrix.

Table 1 contains split up CPU times including aggregation, restriction of stiffness matrix, factorization of auxiliary matrix, incomplete factorization of source matrix, and iterative procedure of GAM solver for all obstacle test problems. Finally, Table 2 and Table 3 compare GAM Solver in terms of the CPU and memory requirements with the Sparse Direct Solver [1] and PCG Solver with Modified Incomplete Cholesky preconditioner. Computations were carried out on SUN SPARC 10/51 Workstation.

So far only in-core solution methods have been considered. Clearly an ultimate solution engine should have an out-of-core capabilities, since it is not usually possible to keep the entire stiffness matrix in RAM. An out-of-core version of GAM is currently being investigated.

REFERENCES

1 A.George and J.W.H. Liu, ‘The evolution of the minimum degree ordering algorithm,’

Report CS-87-06, North York, Otario York University, 1987.

2 O.Axelsson and V.A.Barker, '*Finite Element Solution of Boundary Value Problems,*' Academic Press, 1984.

3 R. P. Fedorenko, 'A relaxation method for solving elliptic difference equations,' *USSR Computational Math. and Math. Phys*., Vol. 1, No. 5, pp.1092-1096, 1962.

4 A.Brandt, S.F.McCormick, and J.W.Ruge, 'Algebraic multigrid (AMG) for sparse matrix equations' in *Sparsity and Its Applications*, D.J. Evans, ed., Cambridge Univ. Press, Cambridge, 1984.

5 W. Leontief, '*The Structure of the American Economy 1919-1939*, Oxford U.P., NY, 1951.

6 V.E.Bulgakov, 'Multi-level iterative technique and aggregation concept with semi-analytical preconditioning for solving boundary-value problems', *Communications in Numerical Methods in Engineering*, Vol. 9, 649-657, (1993).

7 V.E.Bulgakov and G.Kuhn, 'High-performance multilevel iterative aggregation solver for large finite-element structural analysis problems', *Int. J. For Numerical Methods In Engineering*, Vol. 38, 3529-3544, (1995).

8 P. Vanek, 'Acceleration of convergence of a two-level algorithm by smoothing transfer operator,' *Applications of Mathematics*, Vol. 37, pp. 265-274, (1992).

9 W. Hackbusch and U. Trottenberg, *Multigrid Methods*, Springer-Verlag, Berlin, 1992.

10 I. Fried, 'Discretization and round-off Errors in the Finite Element Analysis of elliptic Boundary Value Problems and Eigenvalue Problems, *Ph.D. Thesis, MIT*, 1971.

11 G.W. Stewart, '*Introduction to matrix computations*,' Academic Press, NY, 1973.

12 O. Axelsson, 'Analysis of Incomplete Matrix Factorizations as Multigrid Smoothers for Vector and Parallel Computers,' *Applied Mathematics and Computatio*n, Vol. 19, 3-22, (1986).

13 H.D.Simon, 'The Lanczos algorithm with partial reorthogonalization,' *Math. Comp.* 42, pp. 115-142, (1984).

14 M.A. Ajiz and A.Jennings, 'A robust incomplete Cholesky conjugate gradient algorithm,' *Int. J. For Numerical Methods In Engineering*, Vol. 20, pp. 949-966, (1984)

15 A. Tessler, 'A C^0 Anisotropic three-node shallow shell element,' *Comp. Meth. Appl. Mech. Eng.*, Vol. 78, pp. 89-103, (1990).

16 J. Fish and T.Belytschko, 'Stabilized rapidly convergent 18-degrees-of-freedom flat shell triangular element,' *Int. J. For Numerical Methods In Engineering*, Vol. 33, pp. 149-162, (1992)

17 K.C.Parks and G,Stanley, 'A curved C^0 shell element based on assumed natural coordinate strains,' *Journal of Applied Mechanics*, Vol. 108, 1986, pp.278-290.

Figures and Tables Captions

Figure 1: Obstacle test 3D problems.

Figure 2: Obstacle test shell problems.

Figure 3: GAM solver performance in terms of (a) iteration count, and (b) CPU seconds as a function of limiting eigenvalue parameter γ and number of fill-ins for HSCT problem with MIN3 elements.

Figure 4: GAM solver performance in terms of (a) iteration count, and (b) CPU seconds as a function of limiting eigenvalue parameter γ and number of fill-ins for Diffuser Casing problem with 10-node Tets.

Figure 5: GAM solver performance in terms of (a) iteration count, and (b) CPU seconds as a function of limiting eigenvalue parameter γ and number of fill-ins for Nozzle for Turbines problem.

Figure 6: GAM solver performance in terms of (a) iteration count, and (b) CPU seconds as a function of limiting eigenvalue parameter γ and number of fill-ins for Automobile Body problem with DKT+DMT elements.

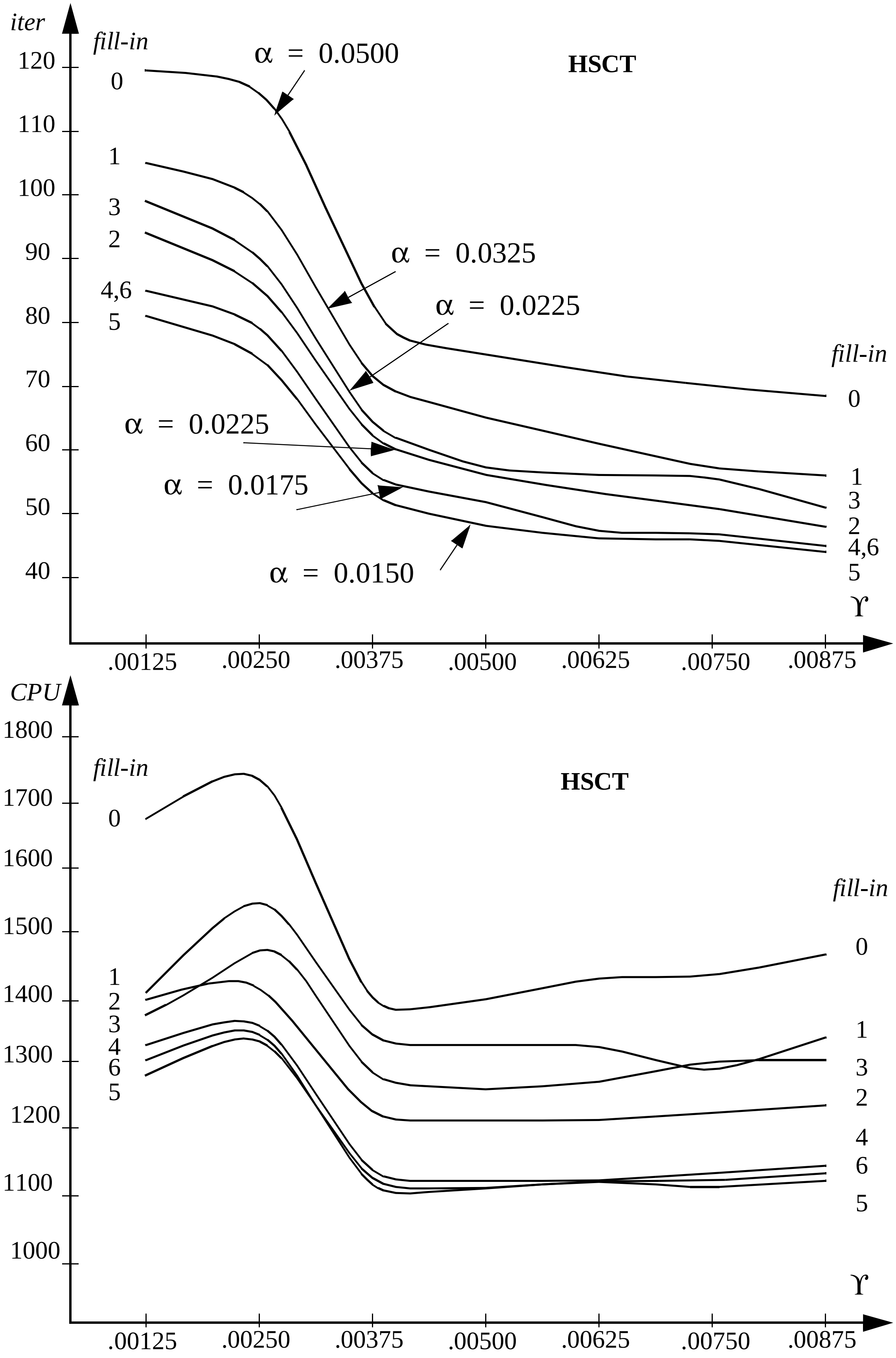
Figure 7: GAM solver performance in terms of iteration count and CPU seconds as a function of number of fill-ins (limiting eigenvalue parameter $\gamma = 0.00625)$ for (a) Diffuser Casing problem with 10-node Tets, and (b) Ring-Strut-Ring problem with 4-node Tets.

Table 1: GAM solver breakdown times in seconds.

Table 2: Comparisons of GAM, PCG(MIC) and Sparse [1] solvers in terms of CPU seconds and iteration count.

Table 3: Comparisons of GAM, PCG(MIC) and Sparse [1] solvers in terms of memory (MB).

FIGURE 3



iter
fill-in
120
110
100
90
80
70
60
50
40
0
1
3
2
4,6
5
α = 0.0500
HSCT
α = 0.0325
α = 0.0225
fill-in
0
α = 0.0225
1
α = 0.0175
3
2
4,6
α = 0.0150
5
ϒ
.00125
.00250
.00375
.00500
.00625
.00750
.00875
CPU
1800
1700
1600
1500
1400
1300
1200
1100
1000
fill-in
0
HSCT
1
2
3
4
6
5
fill-in
0
1
3
2
4
6
5
ϒ
.00125
.00250
.00375
.00500
.00625
.00750
.00875

FIGURE 4

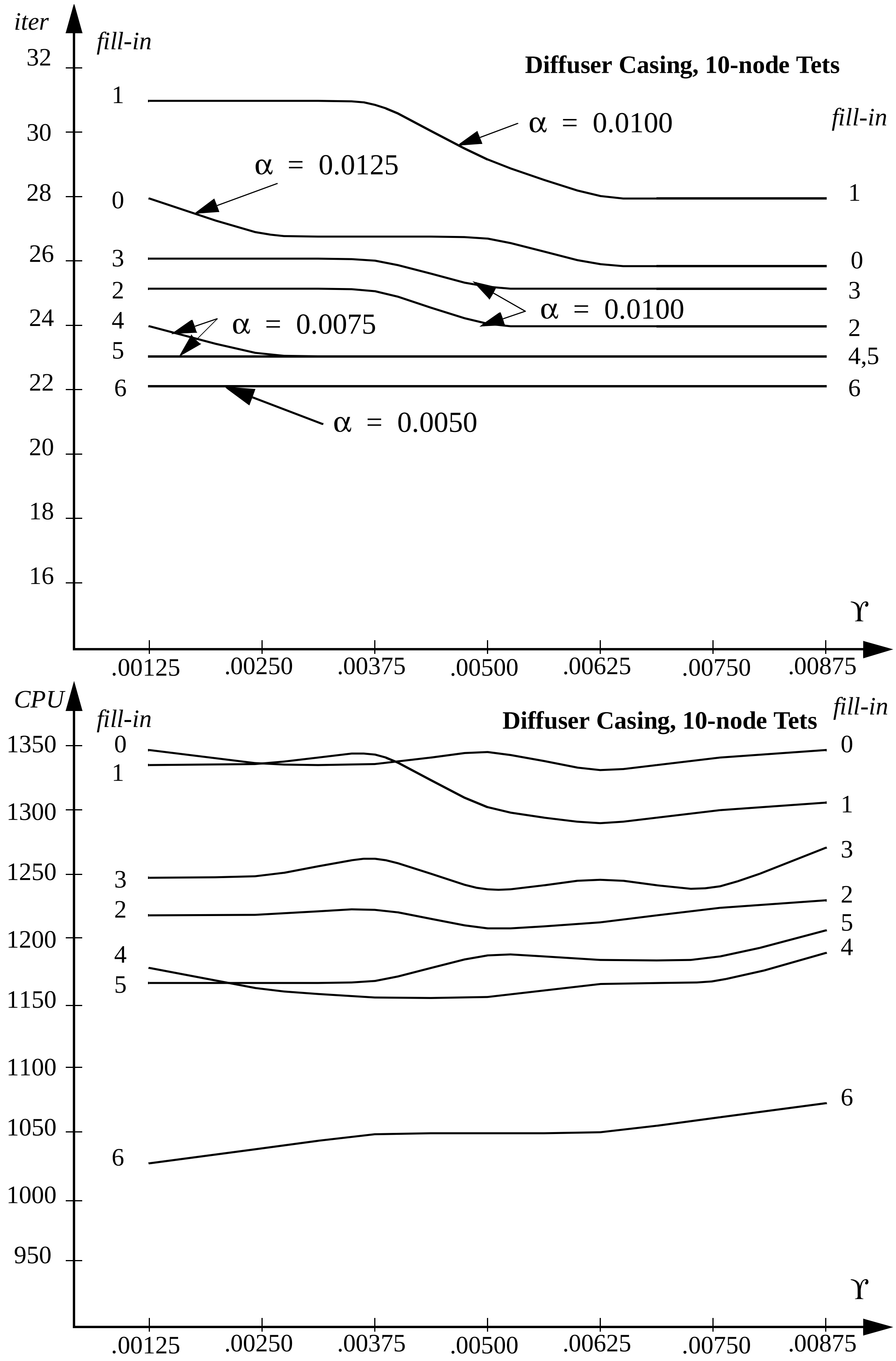

iter
fill-in
Diffuser Casing, 10-node Tets
fill-in
32
30
28
26
24
22
20
18
16
1
0
3
2
4
5
6
α = 0.0100
α = 0.0125
α = 0.0100
α = 0.0075
α = 0.0050
1
0
3
2
4,5
6
ϒ
.00125
.00250
.00375
.00500
.00625
.00750
.00875
CPU
fill-in
Diffuser Casing, 10-node Tets
fill-in
1350
1300
1250
1200
1150
1100
1050
1000
950
0
1
3
2
4
5
6
0
1
3
2
5
4
6
ϒ
.00125
.00250
.00375
.00500
.00625
.00750
.00875

FIGURE 5

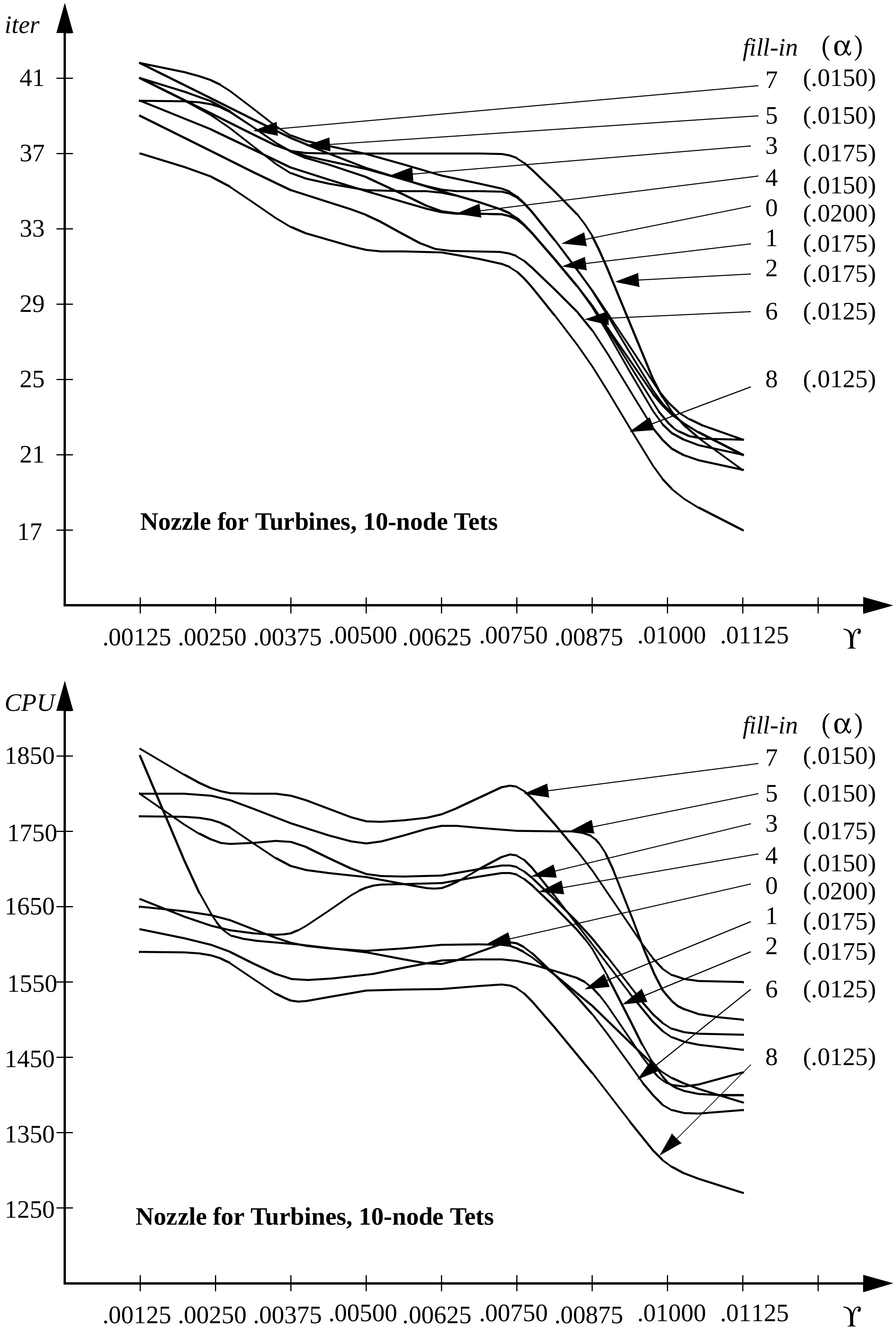
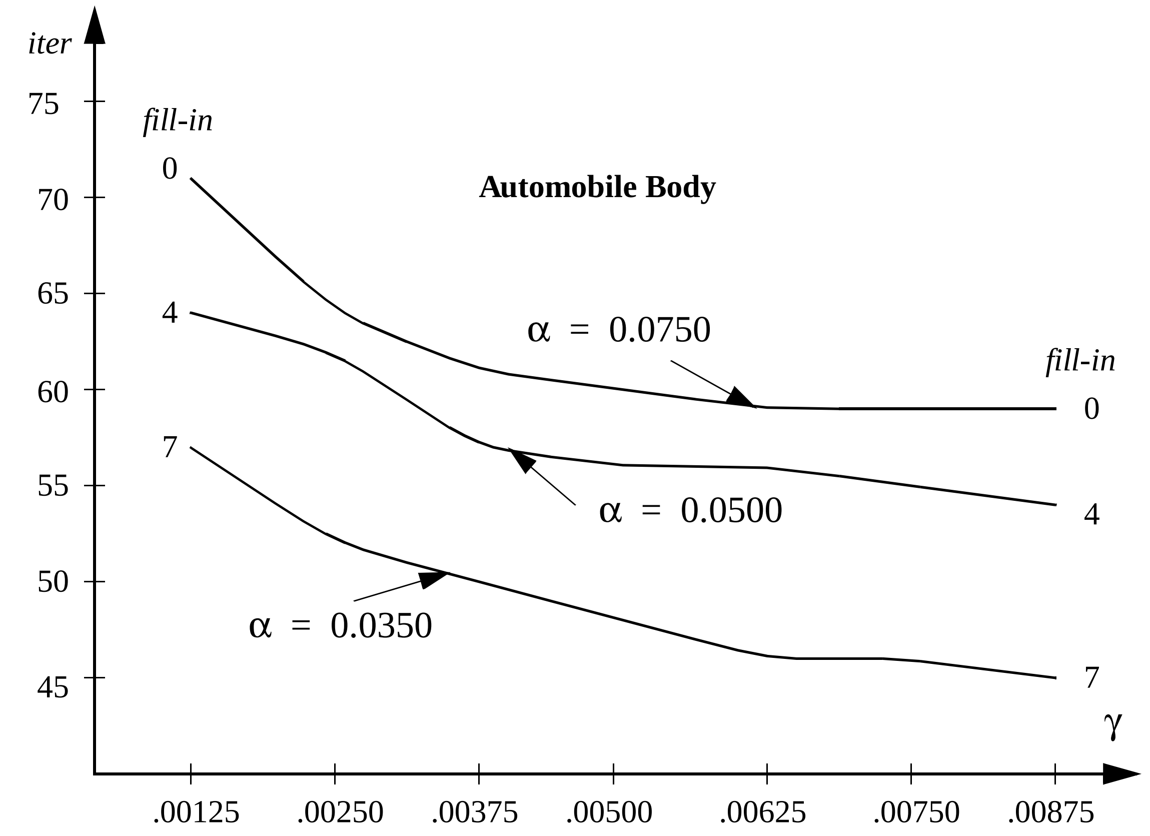

iter
41
37
33
29
25
21
17
fill-in (α)
7 (.0150)
5 (.0150)
3 (.0175)
4 (.0150)
0 (.0200)
1 (.0175)
2 (.0175)
6 (.0125)
8 (.0125)
Nozzle for Turbines, 10-node Tets
.00125 .00250 .00375 .00500 .00625 .00750 .00875 .01000 .01125
ϒ
CPU
1850
1750
1650
1550
1450
1350
1250
fill-in (α)
7 (.0150)
5 (.0150)
3 (.0175)
4 (.0150)
0 (.0200)
1 (.0175)
2 (.0175)
6 (.0125)
8 (.0125)
Nozzle for Turbines, 10-node Tets
.00125 .00250 .00375 .00500 .00625 .00750 .00875 .01000 .01125
ϒ

FIGURE 6


iter
75
70
65
60
55
50
45
fill-in
0
4
7
Automobile Body
α = 0.0750
α = 0.0500
α = 0.0350
fill-in
0
4
7
γ
.00125
.00250
.00375
.00500
.00625
.00750
.00875

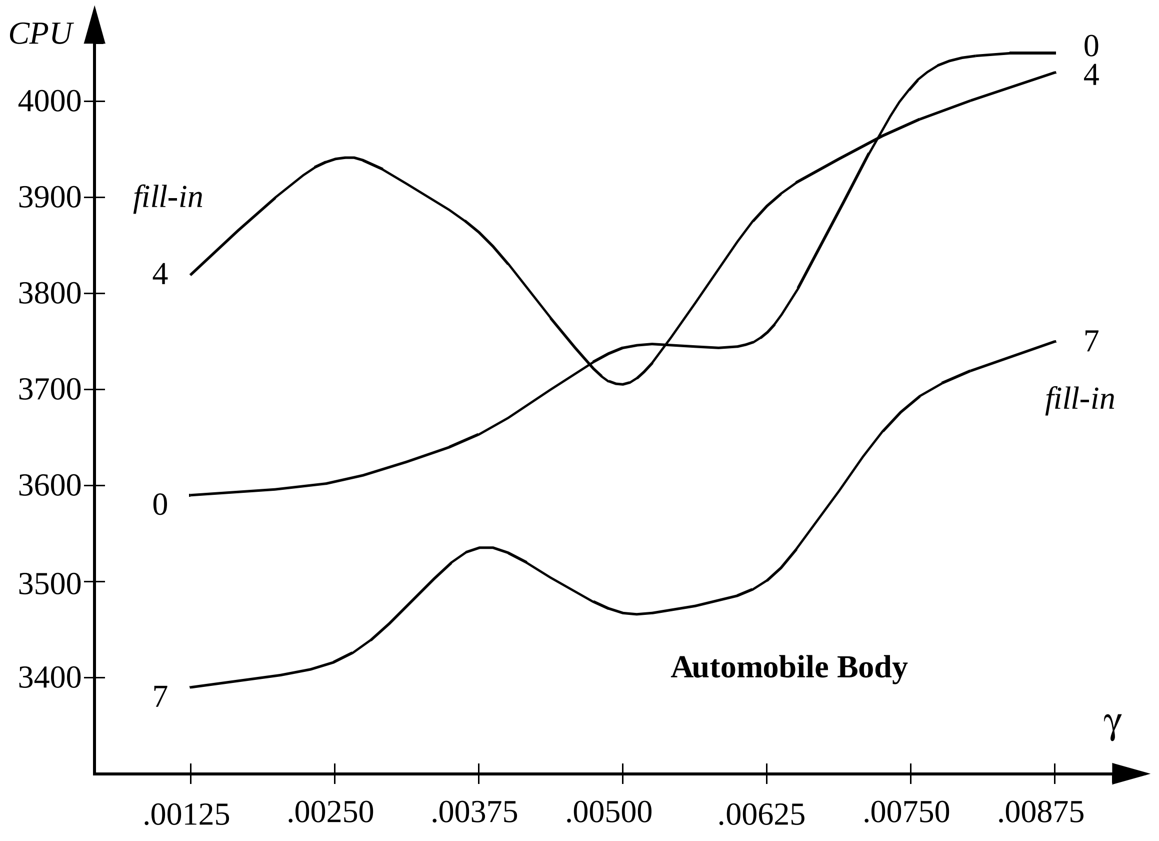

CPU
4000
3900
3800
3700
3600
3500
3400
fill-in
4
0
7
0
4
7
fill-in
Automobile Body
γ
.00125
.00250
.00375
.00500
.00625
.00750
.00875

FIGURE 7

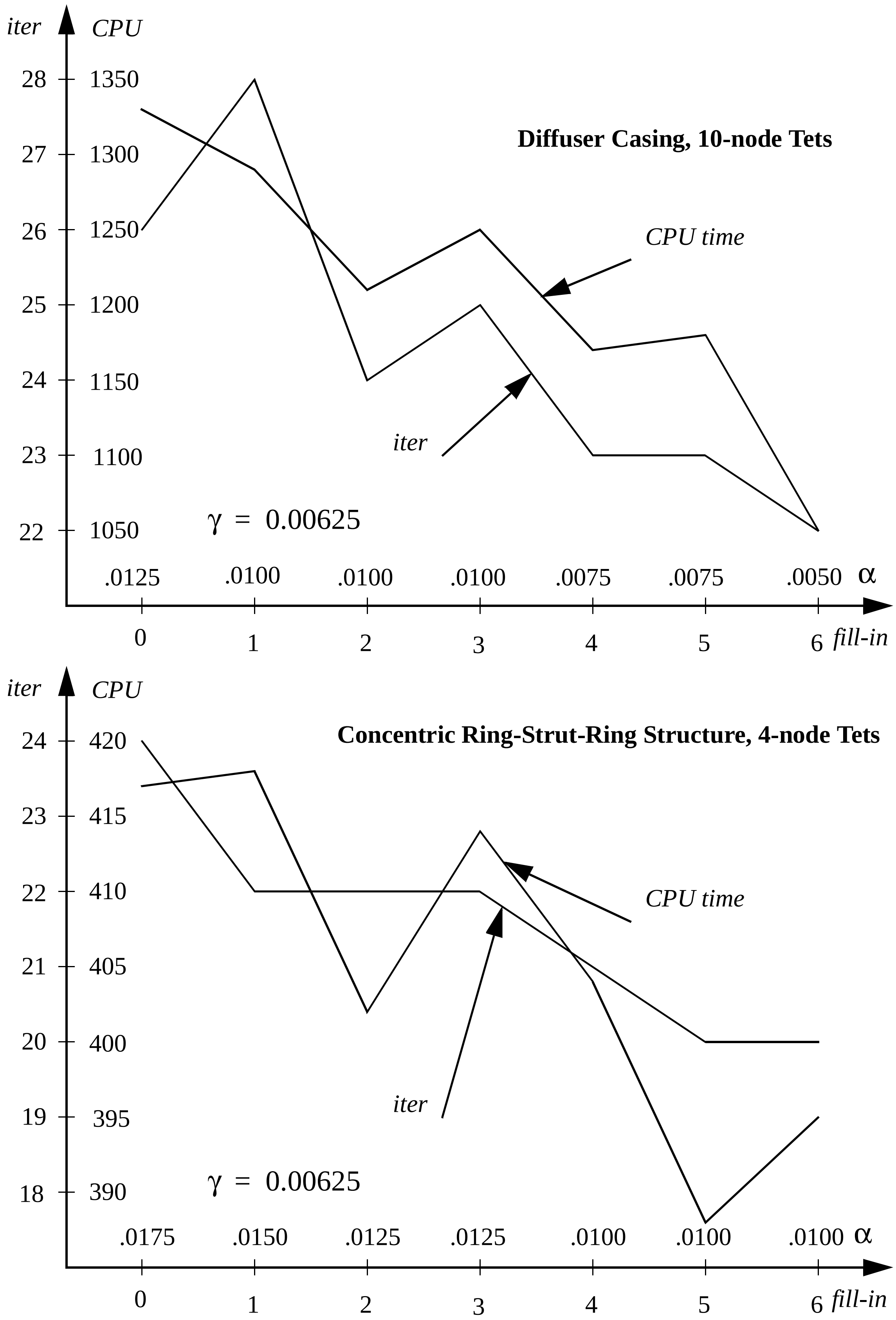

iter
CPU
28
27
26
25
24
23
22
1350
1300
1250
1200
1150
1100
1050
Diffuser Casing, 10-node Tets
CPU time
iter
γ = 0.00625
.0125
.0100
.0100
.0100
.0075
.0075
.0050
α
0
1
2
3
4
5
6
fill-in
iter
CPU
24
23
22
21
20
19
18
420
415
410
405
400
395
390
Concentric Ring-Strut-Ring Structure, 4-node Tets
CPU time
iter
γ = 0.00625
.0175
.0150
.0125
.0125
.0100
.0100
.0100
α
0
1
2
3
4
5
6
fill-in

TABLE 1

Problem title	Solver (total)	Aggregation	Restriction	Factorization	Incomplete factorization	Iteration process	Number of iterations
Diffuser Casing	1021	162	110	126	93	530	22
Turbine Blade	2378	294	280	861	190	753	19
Concentric Structure	346	97	19	17	29	184	18
Nozzle for Turbine	1288	165	166	137	103	717	34
Casting Setup	1493	197	150	169	114	863	30
HSCT (MIN3)	1255	85	117	341	34	678	56
Automobile(DKT +DMT)	2778	266	209	258	217	1828	48
Automobile (MIN3)	3146	324	390	622	230	1580	42
Canoe	1126	110	110	48	98	760	32

TABLE 2

Problem title	Sparse	PCG (MIC)		GAM	
	CPU(s)	CPU(s)	# of iterations	CPU(s)	# of iterations
Diffuser Casing	8692	12276	1531	1021	22
Turbine Blade	out of memory	9862	757	2378	19
Concentric Structure	687	3881	1083	346	18
Nozzle for Turbine	7271	8290	1056	1288	34
Casting Setup	3150	33879	3755	1493	30
HSCT (MIN3)	994	24685	7278	1255	56
Automobile (DKT+DMT)	2678	76003	5939	2788	48
Automobile (MIN3)	2678	83877	6594	3146	42
Canoe	1351	8106	1254	1126	32

TABLE 3

Problem title	Sparse	PCG (MIC)	GAM
Diffuser Casing	995	141	172
Turbine Blade	>1500	255	311
Concentric Structure	337	51	79
Nozzle for Turbine	996	138	169
Casting Setup	1012	149	199
HSCT (MIN3)	207	71	99
Automobile (DKT+DMT)	512	205	255
Automobile (MIN3)	512	205	265
Canoe	395	61	88