Efficient Solution of Elliptic Partial Differential Equations via Effective Combination of Mesh Quality Metrics, Preconditioners, and Sparse Linear Solvers

Jibum Kim[1], Shankar Prasad Sastry[1], and Suzanne M. Shontz[1]

Department of Computer Science and Engineering
The Pennsylvania State University
University Park, PA 16802
jzk164@cse.psu.edu, sps210@cse.psu.edu, shontz@cse.psu.edu

Summary. In this paper, we study the effect the choice of mesh quality metric, preconditioner, and sparse linear solver have on the numerical solution of elliptic partial differential equations (PDEs). We smoothe meshes on several geometric domains using various quality metrics and solve the associated elliptic PDEs using the finite element method. The resulting linear systems are solved using various combinations of preconditioners and sparse linear solvers. We use the inverse mean ratio and vertex condition number metrics in addition to interpolation-based, scale-variant and scale-invariant metrics. We employ the Jacobi, incomplete LU, and SSOR preconditioners and the conjugate gradient, minimum residual, generalized minimum residual, and bi-conjugate gradient stabilized solvers. We focus on determining the most efficient quality metric/preconditioner/linear solver combination for the numerical solution of various elliptic PDEs.

1 Introduction

Discretization methods, such as the finite element (FE) method, are commonly used in the numerical solution of partial differential equations (PDEs). The accuracy of the computed PDE solution depends on the degree of the approximation scheme, the number of elements in the mesh [1], and the quality of the mesh [2, 3]. In addition, the stability and convergence of the finite element method is affected by poor quality elements [4].

Analytical studies have been performed at the intersection of meshing and linear solvers. For example, mathematical connections between mesh geometry, interpolation errors, and stiffness matrix conditioning for triangular and tetrahedral finite element meshes have been studied [5]. A mesh and solver co-adaptation strategy for anisotropic problems has been developed [6]. Rela-

tionships between the spectral condition number of the stiffness matrix and mesh geometry for second-order elliptic problems have been determined [7].

Several computational studies have been performed which examined the connections between finite element meshes and linear solvers in various contexts. For example, the effect of unstructured meshes on the preconditioned conjugate gradient solver performance for the solution of the Laplace and Poisson equations has been examined [8, 9]. In [10], the relative performance of multigrid methods for unstructured meshes was studied on fluid flow and radiation diffusion problems. Trade-offs associated with the cost of mesh improvement in terms of solution efficiency has been examined for fluids [11, 12].

In this paper, we examine the connections between geometry, mesh smoothing, and solution convergence for elliptic PDEs via an engineering approach. In particular, we seek answers to the following questions pertaining to the solution of an elliptic PDE on a given geometric domain. Which mesh quality metric should be used to smoothe the initial mesh? What is the most efficient combination of mesh quality metric, preconditioner, and solver for solving an elliptic PDE? What is the effect of modifying the PDE coefficients and boundary conditions on the answers to the above questions? Our goal is to determine the best quality metric/preconditioner/linear solver combination which results in a small condition number of the preconditioned matrix and fast solver convergence for a given PDE, geometric domain, and initial mesh.

To answer the above questions, we use Mesquite [13], a mesh quality improvement toolkit, and PETSc [14], a linear solver toolkit, to perform a numerical study investigating the performance of several mesh quality metrics, preconditioners, and sparse linear solvers on the solution of various elliptic PDEs of interest. The quality metric/preconditioner/linear solver combinations are compared on the basis of efficiency in solving several elliptic PDEs on realistic unstructured tetrahedral finite element meshes. We use Mesquite and PETSc in their native state with the default parameters. Only these two toolkits are employed so that differences in solver implementations, data structures, and other such factors would not influence the results.

2 Finite Element Solution of Elliptic PDEs

We consider the solution of second-order elliptic PDEs using the finite element (FE) method. An elliptic PDE on a geometric domain, Ω, is defined as

$$-\Delta u + au = f \text{ on } \Omega, \tag{1}$$

where a and f are given functions. If $a = 0$, (1) reduces to Poisson's equation. We consider both Dirichlet and generalized Neumann boundary conditions on the boundary, $\partial\Omega$. The FE method [15] is used to discretize the domain, Ω, and to discretize the PDE resulting in the linear system

$$A\xi = b. \tag{2}$$

Triangular and tetrahedral meshes are used to discretize the domain, Ω, in 2D and 3D, respectively. The approximate solution, ξ, of u can be computed by solving (2). The matrix A is given by $A{=}K{+}M{+}N$, where K is the stiffness matrix; M is the mass matrix, and N is a matrix containing boundary information. For elliptic PDEs, K is a symmetric positive definite or nonnegative definite matrix, and M is a symmetric positive definite matrix [7]. The vector b is the sum of two vectors, F and G. For a two-dimensional geometric domain with generalized Neumann boundary conditions, $\overrightarrow{n} \cdot (\nabla\ u){+}du{=}g$, on $\partial\Omega$, where d and g are given functions. Furthermore, the K, M, N matrices and the F and G vectors can be computed as follows [15]:

$$K_{i,j} = \int_{\Omega} (\bigtriangledown\phi_i \cdot \bigtriangledown\phi_j)\, dx\ dy;\ M_{i,j} = \int_{\Omega} (a\phi_i \cdot \phi_j)\, dx\ dy;$$

$$N_{i,j} = \int_{\partial\Omega} (d\phi_i \cdot \phi_j)\, ds;\ F_i = \int_{\Omega} (f\phi_i)\, dx\ dy;\ G_i = \int_{\partial\Omega} (g\phi_i)\, ds,$$

where ϕ_i and ϕ_j are piecewise linear basis functions.

3 Mesh Quality Metrics

Table 1 provides the notation used to define the following mesh quality metrics: inverse mean ratio (IMR) [16], vertex condition number (VCN) [17], an interpolation-based, scale-variant metric (SV) [5], and an interpolation-based, scale-invariant metric (SI) [5]. Table 2 defines IMR, VCN, SV, and SI. For IMR and VCN, a lower value indicates a higher quality element. For SV and SI, a higher value indicates a higher quality element.

Notation	Definition
a, b, c, and d	Position vectors for vertices in a tetrahedral element
$C = [b-a;\ c-a;\ d-a]$	Jacobian of a tetrahedral element
$W = \begin{pmatrix} 1 & \frac{1}{2} & \frac{1}{2} \\ 0 & \frac{\sqrt{3}}{2} & \frac{\sqrt{3}}{6} \\ 0 & 0 & \frac{\sqrt{2}}{\sqrt{3}} \end{pmatrix}$	Incidence matrix for an equilateral tetrahedron
Ar_1, Ar_2, Ar_3, and Ar_4	Areas of the triangular faces in a tetrahedral element
l_{ij}	Length of the edge common to triangular face i and j in a tetrahedral element
Vol	Volume of a tetrahedral element
Area	Area of a triangular element
s_1, s_2, and s_3	Edge lengths of a triangular element

Table 1. Notation used in the definition of mesh quality metrics in Table 2. The above notation is for the 3D case. Similar quantities can be defined in 2D.

Quality Metric	Formula
IMR	$\left\|CW^{-1}\right\|_F^2 \,/\, (3\left\|\det(CW^{-1})\right\|^{\frac{2}{3}})$ [16]
VCN	$\|C\|_F \|C^{-1}\|_F$ [17]
SV (2D)	Area$/s_1 s_2 s_3$ [5]
SV (3D)	$(\text{Vol} \sum_{i=1}^{4} Ar_i)/(\sum_{1\leq i<j\leq 4} Ar_i Ar_j l_{ij}^2)$ [5]
SI (2D)	Area$/(s_1 s_2 s_3)^{\frac{2}{3}}$ [5]
SI (3D)	$\text{Vol}\left((\sum_{i=1}^{4} Ar_i)/(\sum_{1\leq i<j\leq 4} Ar_i Ar_j l_{ij}^2)\right)^{\frac{3}{4}}$ [5]

Table 2. The mesh quality metric definitions.

4 Mesh Optimization

We denote the elements of a mesh and the number of mesh elements by E and $|E|$, respectively. The overall quality of the mesh, Q, is a function of the individual element qualities, q_i, where q_i is the quality of the i^{th} element in the mesh. The mesh quality depends on both the choice of q_i, which is described in Section 3, and the function used to combine them. For the IMR and VCN quality metrics, we define the overall mesh quality, Q, as the sum of squares of the individual elements' qualities:

$$Q = \sum_{i=1}^{|E|} q_i^2. \tag{3}$$

For the SV and SI quality metrics, we define the overall mesh quality, Q, as the sum of the squares of the reciprocal of the individual elements' qualities:

$$Q = \sum_{i=1}^{|E|} \frac{1}{q_i^2}. \tag{4}$$

We use the Fletcher-Reeves nonlinear conjugate gradient method [18] in Mesquite [13] to minimize Q (defined by either (3) or (4)). Mesquite employs a line search version of the nonlinear conjugate gradient method.

5 Iterative Linear Solvers

Four iterative Krylov subspace methods are employed to solve the preconditioned linear system. The conjugate gradient (CG) solver is a well-known

iterative method for solving systems with symmetric positive definite matrices. It produces a sequence of orthogonal vectors on successive iterations. Let P be the preconditioner (to be described in Section 6). The convergence rate of the preconditioned CG method depends upon the condition number, κ, of $P^{-1}A$. In the 2-norm, κ can be approximated as

$$\kappa_2(P^{-1}A) = \left\|P^{-1}A\right\|_2 \left\|P^{-1}A\right\|_2^{-1} \approx \lambda_{max}(P^{-1}A)/\lambda_{min}(P^{-1}A),$$

where λ_{max} and λ_{min} are the maximum and minimum eigenvalues of $P^{-1}A$, respectively. The fastest convergence occurs when eigenvalues are clustered around the same non-null value and, hence, κ is near 1.

The minimal residual (MINRES) algorithm solves linear systems with symmetric indefinite matrices. It generates a sequence of orthogonal vectors and attempts to minimize the residual in each iteration. Similar to CG, a small condition number for $P^{-1}A$ and clustering of eigenvalues around the same non-null value results in fast convergence.

For nonsymmetric matrices, the generalized minimal residual (GMRES) method is one of the most widely used iterative solvers. Similar to CG and MINRES, GMRES computes orthogonal vectors on each iteration; however, the entire sequence needs to be stored. Therefore, the version of GMRES which restarts GMRES every m steps, i.e., GMRES(m), is used in practice. It is known that a large value of m is effective in decreasing the convergence time; however, the optimal value of m depends upon the problem [19].

The biconjugate gradient stabilized (Bi-CGSTAB) method is a biorthogonalization technique, which generates two sets of biorthogonal vectors instead of producing long orthogonal vectors. Bi-CGSTAB is known to have comparable or even faster convergence than other biorthogonalization methods such as the conjugate gradient squared method. However, Bi-CGSTAB sometimes shows an irregular convergence rate similar to other biorthogonalization methods [20]. Bi-CGSTAB and GMRES are the most widely-used iterative methods for solving systems based on nonsymmetric matrices.

6 Preconditioners

The objective of introducing a preconditioner, P, into the solution of a linear system is to make the system easier to solve, whereby reducing the convergence time of the iterative solver. The reader is referred to [21] (and the references therein) for further information on iterative solvers and preconditioners. In this paper, four preconditioners are employed. The first is the Jacobi preconditioner, which is simply the diagonal of A.

The second is the symmetric successive over relaxation (SSOR) preconditioner. SSOR is similar to Jacobi but decomposes A into L (the strictly lower triangular part), D (the diagonal), and U (the strictly upper triangular part), i.e., A=L+D+U. The SSOR preconditioner is given by

$$P = (D - \omega L)D^{-1}(D - \omega U),$$

where ω represents the relaxation coefficient.

The incomplete LU (ILU) preconditioner with level zero fill-in (ILU(0)) and level one fill-in (ILU(1)) are used as the third and fourth preconditioners, respectively. The basic idea of the ILU preconditioner is to determine lower ($\tilde{L}$) and upper triangular ($\tilde{U}$) matrices such that the matrix $\tilde{L}\tilde{U}$-A satisfies certain constraints [21]. The ILU preconditioner works well for many problems, but it fails when it encounters negative or zero pivots.

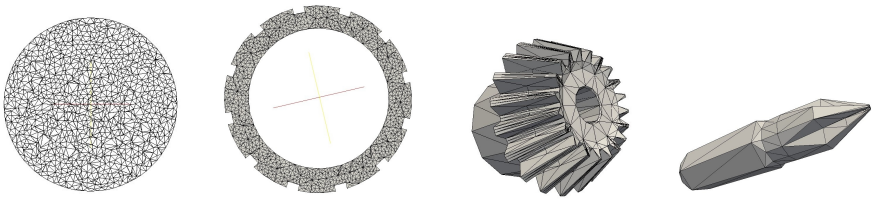
7 Numerical Experiments

Three experiments are performed to investigate the following questions. (1) Which mesh quality metric should be used to smoothe the initial mesh? (2) What is the most efficient combination of mesh quality metric, preconditioner, and solver for solving an elliptic PDE? (3) What is the effect of modifying the PDE coefficients and boundary conditions on the answers to the above questions? Table 3 summarizes the experiments and corresponding PDE problems to be solved. In Experiments 7.1 and 7.2, we focus on solving Poisson's equation with Dirichlet boundary conditions in order to obtain insight on the above questions and then extend the results to other PDE problems in Experiment 7.3. The machine employed for this study is equipped with an Intel Xeon x550 processor (2.67 GHz) and 24GB of RAM [22].

Exp. No.	Exp. Name.	PDE Problems
7.1	Choice of metric	(a) $-\Delta u$=1 on Ω, u=0 on $\partial\Omega$
7.2	Best combination	$-\Delta u$=1 on Ω, u=0 on $\partial\Omega$
7.3	Modification of	(b) $-\Delta u+100u$=1 on Ω, u=0 on $\partial\Omega$
	PDE coefficients and	(c) $-\Delta u+100u$=1 on Ω, $\overrightarrow{n} \cdot (\nabla u)+100u$=1 on $\partial\Omega$
	boundary conditions	(d) $-\Delta u$=1 on Ω, $\overrightarrow{n} \cdot (\nabla u)+u$=1 on $\partial\Omega$
		(e) $-\Delta u$=1 on Ω, $\overrightarrow{n} \cdot (\nabla u)+100u$=1 on $\partial\Omega$

Table 3. Listing of numerical experiments. The letters (a) through (e) represent the five PDE problems under consideration.

Geometric Domains. The four geometric domains considered in our experiments are shown in Figure 1. Circle and gear are 2D problems, whereas bevel and drill are 3D problems. Triangle [23] and Tetgen [24] were used to generate initial meshes. Half the interior vertices in each mesh were perturbed to create test meshes that were further from optimal. Properties of the test meshes and the corresponding finite element matrices are shown in Table 4. Here, nnz is



(a) Circle mesh (b) Gear mesh (c) Bevel mesh (d) Drill mesh

Fig. 1. Coarse initial meshes on the circle, gear, bevel, and drill geometric domains indicative of the actual initial meshes.

mesh	# vertices	# elements	nnz
Circle (2D)	508,173	1,014,894	3,554,395
Gear (2D)	499,842	995,142	3,489,810
Bevel (3D)	492,003	3,001,591	7,583,137
Drill (3D)	500,437	3,084,942	7,757,509

Table 4. Properties of meshes on geometric domains and matrix A

the number of non-zeros in matrix A.

Mesh Smoothing. In order to improve the quality of each mesh, a local implementation of the nonlinear conjugate gradient mesh smoothing method (described in Section 4) was used in conjunction with the mesh quality metrics (described in Section 3). The minimum, average, and maximum mesh qualities before and after smoothing for the circle, gear, bevel, and drill are shown in Table 5. For our experiments, accurate mesh smoothing corresponds to five iterations of smoothing, as the mesh quality did not improve significantly after five iterations. Similarly, one iteration of smoothing was employed for inaccurate mesh smoothing, as a significant improvement in the mesh quality was observed after just one iteration. The results in Table 5 indicate that, in addition to an improvement in the average mesh quality, the quality of the worst mesh elements improves as accurate mesh smoothing is applied.

Finite Element Solution. The FE method described in Section 2 is used to discretize the domain, Ω, and to generate a linear system of the form $A\xi{=}b$. PETSc [14] is used to generate the preconditioners, P, and to solve the linear system, $P^{-1}A\xi{=}P^{-1}b$. We employ the solvers and preconditioners described in Sections 5 and 6, respectively, to solve the preconditioned linear system. Table 6 enumerates the 16 preconditioner-solver combinations used in our experiments. The default parameters for each preconditioner and solver were employed except for the GMRES restart value. We employed a restart value of 100 which was the most effective in preliminary experiments. This is consistent with the fact that larger m values often result in decreased solver time [19].

(a) 2D circle and gear meshes

Metric	Smoothing	Circle			Gear		
		min	avg	max	min	avg	max
IMR	Initial	1.000	1.399	1916	1.000	1.570	339025
	Inaccurate	1.000	1.123	53.527	1.000	1.066	153.772
	Accurate	1.000	1.039	4.727	1.000	1.043	6.343
VCN	Initial	1.000	1.393	479	1.000	1.503	67806
	Inaccurate	1.001	1.113	10.726	1.000	1.063	30.196
	Accurate	1.000	1.039	2.920	1.000	1.042	3.535
SV	Initial	5.80e-4	3.35e-3	4.595	2.24e-3	5.41e-3	1194
	Inaccurate	4.21e-4	2.87e-3	0.220	8.30e-4	3.60e-3	0.736
	Accurate	7.18e-4	2.72e-3	0.028	7.66e-4	3.53e-3	0.055
SI	Initial	1.107	1.372	1709	1.115	1.589	299877
	Inaccurate	1.113	1.198	40.526	1.108	1.159	87.734
	Accurate	1.117	1.149	3.686	1.109	1.145	3.972

(b) 3D bevel and drill meshes

Metric	Smoothing	Bevel			Drill		
		min	avg	max	min	avg	max
IMR	Initial	1.000	1.501	1235	1.000	1.487	2132
	Inaccurate	1.000	1.320	63.656	1.000	1.311	15.568
	Accurate	1.000	1.233	26.459	1.000	1.226	12.334
VCN	Initial	1.069	1.927	1537	1.067	1.887	3767
	Inaccurate	1.056	1.367	39.244	1.039	1.362	5.272
	Accurate	1.031	1.262	27.157	1.030	1.258	3.756
SV	Initial	0.262	3.492	65652	0.208	0.710	32477
	Inaccurate	0.199	2.505	322	0.204	0.518	29.588
	Accurate	0.200	2.210	110.355	0.206	0.457	6.492
SI	Initial	1.000	2.018	38421	1.000	1.967	86880
	Inaccurate	1.000	1.471	243.199	1.000	1.458	272.944
	Accurate	1.000	1.316	67.963	1.000	1.305	17.744

Table 5. The quality of the initial, inaccurately smoothed, and accurately smoothed meshes for the circle, gear, bevel, and drill geometric domains.

The default stopping criteria in PETSc were employed. For example, the absolute tolerance, abstol, and the relative tolerance, rtol, are set to 1e-50 and 1e-05, respectively. The maximum number of iterations for solving the preconditioned linear system is set to 10,000. When the preconditioned linear system is solved, ξ_0 is set to the default value of 0. The preconditioned linear system converges on the i^{th} iteration if the following inequality is satisfied:

$$\|r_i\| < \max(\text{rtol}\,\|r_0\|, \text{abstol}), \tag{5}$$

where r_i is the residual at the i^{th} iteration and r_0 is the initial residual.

Preconditioner	Solver			
	CG	**GMRES**	**MINRES**	**BI-CGSTAB**
Jacobi	1	2	3	4
SSOR	5	6	7	8
ILU(0)	9	10	11	12
ILU(1)	13	14	15	16

Table 6. The sixteen combinations of preconditioners and solvers. For example, 10 refers to using the ILU(0) preconditioner with the GMRES solver.

Timing. In our experiments, the total time is defined as the sum of the smoothing time and the solver time. The smoothing time is the time to achieve an accurately smoothed mesh as described in Section 4 except for our preliminary experiments which also included inaccurate mesh smoothing. The solver time is the time the solver takes to satisfy (5) and includes the time to generate P and to solve $P^{-1}\ A\xi{=}P^{-1}\ b$. When the linear system diverges or does not converge, we report failure.

Inaccurate Mesh Smoothing. For our preliminary experiments, we consider solving Poisson's equation with Dirichlet boundary conditions, i.e., problem (a) in Table 3, on inaccurately and accurately smoothed meshes for a given domain. We consider inaccurate mesh smoothing, as engineers often perform inaccurate mesh smoothing in practice.

Experimental results for inaccurate and accurate mesh smoothing are shown in Table 7. For this experiment, the SSOR preconditioner with the GMRES solver is employed. These results are representative of the results obtained when other preconditioner-solver combinations are used. Table 7 shows the smoothing time and the solver time when employing various amounts of mesh smoothing for the three mesh quality metrics described in Section 3. We observe the following rank ordering with respect to smoothing time: IMR $<$ VCN $<$ SV $<$ SI. The ranking is in order of fastest to slowest. This rank ordering also holds true when we consider the smoothing and solver time together. The rank ordering between different quality metrics will be discussed in more detail in Section 7.1. Table 7 also shows that accurately smoothed meshes result in matrices with a lower condition number than those of inaccurately smoothed meshes. This results in a lower solver time for accurately smoothed meshes. However, in terms of the total time, inaccurately smoothed meshes result in faster total time than do accurately smoothed meshes for the SV and SI quality metrics because the smoothing time is very high for these two metrics.

Because accurately smoothed meshes were observed to result in lower total time than inaccurately smoothed meshes, only accurately smoothed meshes are considered from here onwards.

	Mesh quality metrics								
	NOS	IMR		VCN		SV		SI	
Smoothing	0	1	5	1	5	1	5	1	5
λ_{min} $(P^{-1}A)$	1.8e-05	2.4e-5	2.7e-05	2.5e-05	2.7e-05	2.4e-05	2.7e-05	2.4e-05	2.7e-05
λ_{max} $(P^{-1}A)$	1	1	1	1	1	1	1	1	1
$\kappa(P^{-1}A)$	55252	40822	36853	40347	36845	41979	37013	41411	36914
Smoothing time	0	8	41	13	67	98	517	142	755
Solver time	1448	905	630	951	771	1037	773	898	612
Total time	1448	913	671	964	838	1135	1290	1040	1367

Table 7. Minimum and maximum eigenvalues, condition number, and timing (sec) as a function of quality metric and amount of mesh smoothing on a circle domain. Smoothing represents the number of smoothing iterations performed. NOS (no smoothing) describes the initial mesh. For this table, the SSOR preconditioner with the GMRES solver was employed.

7.1 Choice of Mesh Quality Metric

The goal of this experiment is to determine the best mesh quality metric for solving Poisson's equation with Dirichlet boundary conditions based on efficiency and accuracy.

Efficiency of the Solution. The time required to smoothe the test meshes using various quality metrics is shown in Table 8. If we consider the smoothing time alone, the following rank ordering is seen: IMR < VCN < SV < SI. This is because numerical computation of the IMR metric for mesh optimization is highly optimized in Mesquite. Numerical computation of the other mesh quality metrics, i.e., VCN, SV, and SI, are not as optimized and are less efficient to compute. The computation of SV and SI is also more expensive than the other two quality metrics.

Mesh	Mesh quality metrics			
	IMR	VCN	SV	SI
Circle	41	67	517	755
Gear	39	65	523	784
Bevel	146	208	270	366
Drill	159	213	277	377

Table 8. Mesh smoothing time (sec) for various mesh quality metrics

Table 9 shows typical timing results for smoothing the bevel mesh according to the various quality metrics. In terms of the solver time, there is a significant difference between solving linear systems on smoothed versus nonsmoothed (NOS) meshes. After smoothing, the condition number of $P^{-1}A$ decreases significantly, and the minimum and maximum eigenvalues move closer

to 1. This results in faster convergence of the linear solution on the smoothed meshes. However, in terms of solver time, as long as the same preconditioner and solver are employed, there is little difference when various mesh quality metrics are used to smoothe the mesh, as the metrics essentially all try to generate equilateral elements, and there are no poorly shaped elements after smoothing. The element shapes in the mesh highly affect both the eigenvalues and the condition number of A. Equilateral elements result in a smaller condition number than do poorly shaped elements. Here, poorly shaped means elements with very small (near 0°) or very large (near 180°) dihedral angles [5]. Thus, when the preconditioner and solver are fixed, the choice of mesh quality metric does not significantly affect the condition number or the solver time. Further information on the relationship between eigenvalues and the condition number can be found in [5]. For the total time, the rank ordering amongst mesh quality metrics is: IMR < VCN < SV < SI. A similar trend was observed for meshes on the other domains when solving Poisson's equation with Dirichlet boundary conditions, i.e., problem (a) in Table 3.

Mesh quality metrics	NOS	IMR	VCN	SV	SI
λ_{min} $(P^{-1}A)$	0.005	0.0063	0.0064	0.0063	0.0064
λ_{max} $(P^{-1}A)$	18.2	1.8	1.8	1.8	1.8
$\kappa(P^{-1}A)$	3216	285	285	285	283
Smoothing time	0	159	213	277	377
Solver time	3250	50	48	54	42
Total	3250	209	261	331	419

Table 9. Minimum and maximum eigenvalues, condition number, and timing (sec) for the matrix $P^{-1}A$ for the bevel meshes before and after smoothing. The ILU(0) preconditioner with the GMRES solver was used to solve the resulting linear system.

Accuracy of the Solution. The exact solution of Poisson's equation with Dirichlet boundary conditions (i.e., problem (a) in Table 3) on the unit circle is given by: $u = (1 - x^2 - y^2)/4$, where x and y denote the x and y vertex coordinates, respectively. We compare our FE solution, u_h, with the exact solution to verify the accuracy of our solution. The discretization error, e, between the FE solution and the exact solution is defined as $e = \|u - u_h\|_\infty$. Our results show that for all mesh quality metrics, $e \approx 1e-4$, for the unit circle domain in Table 4. Furthermore, the choice of metric does not affect the solution accuracy for meshes of this size (i.e., approximately 1M elements).

7.2 Best combination of metric, preconditioner and solver

In Section 7.1, we observed that IMR was the most efficient quality metric for smoothing when solving Poisson's equation with Dirichlet boundary conditions (i.e., problem (a) in Table 3). In this experiment, we seek to determine

the most efficient combination of mesh quality metric, preconditioner, and solver for this problem.

The solver time as a function of the different mesh quality metrics for various preconditioner-solver combinations is presented in Table 10 for the circle, gear, bevel, and drill domains. The '*' entries in these tables correspond to preconditioner-solver combinations which either diverge or do not converge. In most cases, the best combinations, which have the fastest solver time, are the SSOR, ILU(0), and ILU(1) preconditioners with the CG solver (5, 9, and 13 in Table 6). The ILU(0) or ILU(1) preconditioners with the MINRES solver (7 and 11 in Table 6) show comparable performance. We observe that the best preconditioner varies with respect to the geometric domain and the mesh quality metric. However, the best solvers do not change in most cases. The CG and MINRES solvers consistently show better performance than do GMRES and Bi-CGSTAB. The main reason is that both CG and MINRES take advantage of the symmetry properties of $P^{-1}A$, because the solvers are designed for symmetric matrices.

Table 10 also shows the number of iterations required to converge, which is an implementation-independent metric. In terms of the number of iterations required to converge, the ILU(1) preconditioner with the MINRES solver (15 in Table 6) outperforms other combinations. In many cases, the MINRES solver requires fewer iterations to converge than does the CG solver although the CG solver has a faster solver time than the CG solver. Note also that the ILU(0) preconditioner has a faster solver time than the ILU(1) preconditioner although the ILU(0) requires more iterations to converge.

The least effective combination observed is the Jacobi preconditioner with the Bi-CGSTAB solver (4 in Table 6). In most cases, this combination diverges because the residual norm does not decrease. The Bi-CGSTAB solver shows irregular convergence behavior as was discussed in Section 5. The least efficient preconditioner-solver combination is the slowest combination which satisfies (5). In most cases, the least efficient combination is the Jacobi preconditioner with the GMRES solver (2 in Table 6). This combination sometimes does not converge due to the iteration limit.

In terms of the solver time and the number of iterations required to converge, the choice of quality metric matters most when a poor combination of preconditioner and solver are used. For example, if we compare the solver time on the circle mesh for the ILU(0) preconditioner with the GMRES solver (2 in Table 6) amongst quality metrics (i.e., IMR and SV) in Table 10(a), the solver time for SV is 40% higher than it is for IMR. However, if a better combination is chosen (e.g., 5, 9, or 13 in Table 6), the difference between mesh quality metrics is not that significant. This demonstrates that the choice of preconditioner and solver is more significant than the choice of mesh quality metric in the solution of Poisson's equation with Dirichlet boundary conditions.

Table 11 shows the solver time and total time for the most and least efficient preconditioner-solver combinations as a function of geometric domain and mesh quality metric. Combinations which did not converge according

(a) Circle. (1-16 denote the preconditioner-solver combinations.)

Metric	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
IMR	381	*	212	*	166	631	137	134	127	631	174	*	79	252	98	83
	1305	*	921	*	593	2052	357	361	530	1684	398	*	284	759	206	197
VCN	377	*	195	*	179	772	154	144	128	708	155	*	75	263	114	98
	1303	*	843	*	594	2349	402	361	530	2120	381	*	284	747	210	211
SV	446	*	262	*	165	773	165	156	146	911	157	*	87	350	198	99
	1441	*	1101	*	609	2322	392	407	541	2885	387	*	284	757	212	204
SI	350	*	187	*	155	613	139	160	139	795	216	*	87	252	86	107
	1307	*	860	*	594	1934	384	361	531	2010	376	*	284	756	208	212

(b) Gear. (1-16 denote the preconditioner-solver combinations.)

Metric	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
IMR	111	913	142	106	62	236	67	82	71	191	73	64	59	102	73	91
	606	2411	430	428	257	475	171	153	233	426	146	174	134	156	87	113
VCN	129	963	93	*	66	164	62	82	54	220	53	56	43	86	69	47
	606	2360	381	*	257	475	169	152	233	427	163	165	134	156	85	90
SV	118	542	101	*	61	143	67	33	53	161	61	*	40	64	42	45
	623	2198	426	*	257	477	176	166	239	440	162	*	134	156	93	87
SI	121	757	102	*	65	171	81	70	49	172	70	68	43	54	44	54
	604	2352	426	*	257	475	181	153	233	427	154	176	134	156	94	104

(c) Bevel. (1-16 denote the preconditioner-solver combinations.)

Metric	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
IMR	79	263	117	*	59	76	54	51	44	50	42	69	63	51	76	169
	190	343	126	*	72	88	54	48	62	67	47	49	43	39	27	35
VCN	113	239	106	*	64	65	91	71	37	48	39	77	59	55	61	85
	190	342	126	*	72	88	53	48	62	67	46	49	43	39	27	35
SV	64	169	63	*	39	60	47	47	32	54	41	44	49	46	52	62
	190	342	130	*	72	88	52	48	62	67	47	49	43	39	27	36
SI	66	140	79	*	37	54	57	42	32	42	42	49	46	45	55	135
	191	343	131	*	72	88	52	48	62	68	47	49	43	39	28	35

(d) Drill. (1-16 denote the preconditioner-solver combinations.)

Metric	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
IMR	169	260	101	*	51	84	65	66	49	72	74	128	75	90	53	97
	282	496	170	*	103	118	75	62	93	100	64	70	49	47	31	41
VCN	110	260	95	*	70	81	67	94	52	106	66	66	72	64	102	81
	281	495	167	*	103	118	77	62	93	100	58	69	49	47	32	41
SV	132	222	91	*	61	71	98	64	63	76	60	150	118	88	58	73
	281	495	179	*	103	118	75	62	93	99	68	69	49	47	30	41
SI	120	386	140	*	57	85	90	49	58	80	58	53	59	56	67	84
	282	496	181	*	103	118	76	62	93	100	57	69	49	47	32	41

Table 10. Linear solver time (sec) and number of iterations required to converge for problem (a) as a function of mesh quality metric for the 16 preconditioner-solver combinations (see Table 6) on the four geometric domains. A '*' denotes failure. For each quality metric, the numbers in the top and bottom rows represent the linear solver time and number of iterations to convergence, respectively.

to (5) are not considered here. In terms of the solver time, the best mesh quality metric varies with respect to the geometric domain, and there is no clear winner. The results also show the importance of choosing an appropriate combination of preconditioner and solver. For example on the gear domain, the least efficient combination (2 in Table 6) is 94% slower than it is for the most efficient combination (13 in Table 6).

We also observe that the rank ordering for mesh quality metrics described in Section 7.1 is consistent with the total time rank ordering. The table shows that IMR with the most efficient (best) combination of preconditioner and solver outperforms the other mesh quality metrics. The total time for IMR is 88% less than it is for SI, for the most efficient (i.e., ILU(1) with CG) combination. For the least efficient combination (i.e., Jacobi with GMRES), IMR is 51% faster than that of the least efficient quality metric, i.e., SI.

(a) Solver time

Mesh		Mesh quality metrics			
		IMR	VCN	SV	SI
Circle	Best	79	75	87	87
	worst	631	772	911	795
Gear	Best	59	43	40	43
	worst	913	963	542	757
Bevel	Best	42	37	32	32
	worst	263	239	169	140
Drill	Best	49	52	61	49
	worst	260	260	222	386

(b) Total time

Mesh		Mesh quality metrics			
		IMR	VCN	SV	SI
Circle	Best	120	142	604	842
	worst	672	839	1428	1550
Gear	Best	98	108	563	827
	worst	952	1028	1065	1541
Bevel	Best	188	245	302	398
	worst	409	447	439	506
Drill	Best	208	265	338	426
	worst	419	473	499	763

Table 11. Solver time (sec) and total time (sec) for problem (a) as a function of the best (most efficient) and worst (least efficient) combination of preconditioner and solver for various mesh quality metrics. Only combinations satisfying (5) are considered.

7.3 Modifying the PDE coefficients and boundary conditions

In this experiment, we modify both the PDE coefficients and boundary conditions and investigate whether or not the previous conclusions hold true for other elliptic PDE problems (i.e., problems (b)-(e) in Table 3).

We first investigate the effect that including the mass matrix, M, has on the best combination of mesh quality metric, preconditioner, and solver for problems (b) and (c) in Table 3. Problem (b) and (c) represent an elliptic PDE with a mass matrix and Dirichlet and generalized Neumann boundary conditions, respectively. Table 12 shows the solver time and the number of iterations required to converge for these two problems. For this experiment, the IMR metric on the bevel domain is employed. Tables 12(a) and 12(b) represent the linear solver time and the number of iterations required to converge for problems (b) and (c), respectively. Similar trends are observed when other mesh quality metrics are employed. Interestingly, in terms of the solver time, the worst combination of preconditioner and solver is different from the previous results obtained for Poisson's equation with Dirichlet boundary conditions. For Poisson's equation with Dirichlet boundary conditions, the worst combination was the Jacobi preconditioner with the Bi-CGSTAB or GMRES solver (4 and 2 in Table 6). However, after we add the mass matrix with a=100 in (1), the worst combinations are the ILU(1) preconditioner with the CG or GMRES solver (13 and 14 in Table 6). In most cases, the best combination is the ILU(0) preconditioner with the MINRES solver (11 in Table 6). Different from Poisson's equation, the solver time of the ILU(1) preconditioner with the CG or MINRES solver on problem (b) is up to 82% larger than it is for the most efficient combination of preconditioner and solver. This trend also occurs for other values of the PDE coefficients, such as $a = 10, 50$ in (1). This example shows that the best and worst combinations of the preconditioner and solver vary due to the addition of the mass matrix. However, in terms of the number of iterations required to converge, the worst combination (13 and 14 in Table 6) requires only two iterations to converge, mainly due to the slow generation time of the ILU(1) preconditioner for these cases. The observed trends are similar for problems (b) and (c).

In terms of the smoothing time, the rank ordering amongst different quality metrics remains the same, i.e., IMR $<$ VCN $<$ SV $<$ SI. In terms of the solver time, the choice of quality metric does not affect the results significantly. The most efficient combination of the mesh quality metric, preconditioner, and solver for solution of these elliptic PDEs is the IMR quality metric with the ILU(0) preconditioner and the MINRES solver. This is consistent with those obtained when solving Poisson's equation with Dirichlet boundary conditions. However, the least efficient combination is now the ILU(1) preconditioner with the GMRES solver (14 in Table 6).

Second, problems (d) and (e) in Table 3, are solved to see the effect of modifying the boundary conditions. Table 12 shows the solver time and the number of iterations required to converge for the IMR quality metric on the

bevel domain. The other mesh quality metrics show similar trends. Tables 12(a) and 12(b) represent the solver time and the number of iterations required to solve problems (d) and (e) in Table 3, respectively. Experimental results show that the most efficient mesh quality metric is still IMR if smoothing and solver time are considered together. The rank ordering amongst the various quality metrics is as follows: IMR < VCN < SV < SI. The least efficient combination of preconditioner and solver is the Jacobi preconditioner with the GMRES solver (2 in Table 6), which is consistent with previous results. There are multiple best combinations, e.g., the SSOR preconditioner with the CG solver (5 in Table 6). The solver time of the best combination is up to 77% faster than it is for the least efficient combination. However, the best combinations are similar to those seen for the previous PDE problems.

PDE problem	Preconditioner-Solver Combinations															
	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
(b)	3	5	3	3	11	3	4	4	6	4	5	4	14	32	17	17
	10	9	6	6	5	5	3	4	3	3	2	2	2	2	2	2
(c)	6	10	9	8	8	5	5	5	7	8	8	8	19	17	18	21
	16	15	10	9	9	9	5	5	8	8	4	6	5	5	3	3

PDE problem	Preconditioner-Solver Combinations															
	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
(d)	155	193	74	*	44	162	93	41	57	64	52	85	96	98	50	58
	185	335	134	*	77	96	58	48	71	90	50	57	37	36	27	30
(e)	83	179	105	112	47	55	61	52	42	42	62	61	62	45	59	71
	188	349	131	133	72	88	54	48	60	72	46	51	34	34	23	32

Table 12. Linear solver time (sec) and number of iterations required to converge as a function of PDE coefficients and boundary conditions on the bevel domain. A '*' denotes failure. For this experiment, the IMR quality metric is used. The keys for the PDE problems and the preconditioner-solver combinations are given in Tables 3 and 6, respectively.

8 Conclusions and Future Work

For all elliptic PDEs considered, the IMR metric is the most efficient. The least efficient mesh quality metric is SI. The best preconditioner-solver combination varies with respect to the PDE. For Poisson's equation, the most efficient preconditioner-solver combinations, which have the fastest solver time, are the SSOR, ILU(0), and ILU(1) preconditioners with the CG solver. The ILU(0) or ILU(1) preconditioner with the MINRES solver shows comparable performance. The least effective combination observed is the Jacobi preconditioner with the Bi-CGSTAB solver, which always diverged. In many cases, the least

efficient combination is the Jacobi preconditioner with the GMRES solver. In terms of the total time, IMR is 88% faster than it is for SI if the most efficient preconditioner-solver combination is employed. For the least efficient combination, IMR is still 51% faster than it is for SI.

Modifying the coefficients and boundary conditions of the elliptic PDEs effects the efficiency ranking of the preconditioner-solver combinations. The efficiency rankings are more sensitive to modifications in the PDE coefficients than to modifications in the boundary conditions. Unlike the solution of Poisson's equation with Dirichlet boundary conditions, the ILU(1) preconditioner with the CG or MINRES solver shows inefficient performance, in terms of the solver time, on elliptic PDEs involving a mass matrix. In addition, the solver time of the ILU(1) preconditioner with the CG or MINRES solver is up to 82% greater than it is for the most efficient combination of preconditioner and solver. However, the rank ordering amongst quality metrics is the same as it is for Poisson's equation.

For future research, we will investigate the relationship between the choice of mesh quality metric and the efficient solution of parabolic and hyperbolic PDEs on anisotropic unstructured meshes.

Acknowledgements

The authors would like to thank Anirban Chatterjee and Padma Raghavan for interesting the third author in this area of research and Nicholas Voshell for helpful discussions. This work was funded in part by NSF grant CNS 0720749 and an Institute for Cyberscience grant from The Pennsylvania State University. This work was supported in part through instrumentation funded by the National Science Foundation through grant OCI-0821527.

References

1. I. Babuska and M. Suri, *The p and h-p versions of the finite element method, basic principles, and properties*, SIAM Rev., 35: 579-632, 1994.
2. M. Berzins, *Solution-based mesh quality for triangular and tetrahedral meshes*, in Proc. of the 6^{th} International Meshing Roundtable, Sandia National Laboratories, pp. 427-436, 1997.
3. M. Berzins, *Mesh quality - Geometry, error estimates, or both?*, in Proc. of the 7^{th} International Meshing Roundtable, Sandia National Laboratories, pp. 229-237, 1998.
4. E. Fried, *Condition of finite element matrices generated from nonuniform meshes*, AIAA Journal, 10: 219-221, 1972.
5. J. Shewchuk, *What is a good linear element? Interpolation, conditioning, and quality measures*, in Proc. of the 11^{th} International Meshing Roundtable, Sandia National Laboratories, pp. 115-126, 2002.

6. Q. Du, Z. Huang, and D. Wang, *Mesh and solver co-adaptation in finite element methods for anisotropic problems*, Numer. Methods Partial Differential Equations, 21: 859-874, 2005.
7. Q. Du, D. Wang, and L. Zhu, *On mesh geometry and stiffness matrix conditioning for general finite element spaces*, SIAM J. Numer. Anal., 47(2): 1421-1444, 2009.
8. A. Ramage and A. Wathen, *On preconditioning for finite element equations on irregular grids*, SIAM J. Matrix Anal. Appl., 15: 909-921, 1994.
9. A. Chatterjee, S.M. Shontz, and P. Raghavan, *Relating Mesh Quality Metrics to Sparse Linear Solver Performance*, SIAM Computational Science and Engineering, Costa Mesa, CA, 2007.
10. D. Mavripilis, *An assessment of linear versus nonlinear multigrid methods for unstructured mesh solvers*, J. Comput. Phys., 175: 302-325, 2002.
11. M. Batdorf, L. Freitag, and C. Ollivier-Gooch, *Computational study of the effect of unstructured mesh quality on solution efficiency*, in Proc. of the 13^{th} CFD Conference, AIAA, Reston, VA, 1997.
12. L. Freitag and C. Ollivier-Gooch, *A cost/benefit analysis of simplicial mesh improvement techniques as measured by solution efficiency*, Internat. J. Comput. Geom. Appl., 10: 361-382, 2000.
13. M. Brewer, L. Freitag Diachin, P. Knupp, T. Leurent, and D. Melander, *The Mesquite Mesh Quality Improvement Toolkit*, in Proc. of the 12^{th} International Meshing Roundtable, Sandia National Laboratories, pp. 239-250, 2003.
14. S. Balay, K. Buschelman, W.D. Gropp, D. Kaushik, M.G. Knepley, L.C. McInnes, B. F. Smith, and Hong Zhang, *PETSc Webpage, http://www.mcs.anl.gov/petsc*, 2009.
15. E.B. Becker, G.F. Carey, and J.T. Oden. *Finite Elements: An Introduction. Prentice-Hall*, Englewood Cliffs, New Jersey, 1981.
16. T. Munson, *Mesh Shape-Quality Optimization Using the Inverse Mean-Ratio Metric*, Mathematical Programming, 110: 561-590, 2007.
17. P. Knupp, *Achieving finite element mesh quality via optimization of the Jacobian matrix norm and associated quantities, Part II - A framework for volume mesh optimization and the condition number of the Jacobian matrix*, Internat. J. Numer. Methods Engrg., 48: 1165-1185, 2000.
18. J. Nocedal and S. Wright, *Numerical Optimization*, Springer-Verlag, 2^{nd} Edn., 2006.
19. A.H. Baker, E.R. Jessup, and Tz.V. Kolev, *A simple strategy for varying the restart parameter in GMRES(m)* , J. Comput. Appl. Math. 230: 751-761, 2009.
20. R. Barrett, M. Berry, T.F. Chan, J. Demmel, J. Donato, J. Dongarra, V. Eijkhout, R. Pozo, C. Romine, and H.Van der Vorst. *Templates for the Solution of Linear Systems: Building Blocks for Iterative Methods*, 2^{nd} Edn., SIAM, 1994.
21. Y. Saad, *Iterative Methods for Sparse Linear Systems*, Society for Industrial and Applied Mathematics, 2^{nd} Edn., 2003.
22. Cyberstar Webpage: http://www.ics.psu.edu/research/cyberstar/index.html.
23. J.R. Shewchuk, *Triangle: Engineering a 2D Quality Mesh Generator and Delaunay Triangulator,* Lecture Notes in Computer Science, vol. 1148, pp. 203-222, 1996.
24. H. Si, TetGen - A Quality Tetrahedral Mesh Generator and Three-Dimensional Delaunay Triangulator, http://tetgen.berlios.de/.