Introducing the Target-Matrix Paradigm for Mesh Optimization via Node-movement

Patrick Knupp

Sandia National Laboratories, pknupp@sandia.gov

Abstract. A general-purpose algorithm for mesh optimization via node-movement, known as the Target-Matrix Paradigm, is introduced. The algorithm is general purpose in that it can be applied to a wide variety of mesh and element types, and to various commonly recurring mesh optimization problems such as shape improvement, and to more unusual problems like boundary-layer preservation with sliver removal, high-order mesh improvement, and edge-length equalization. The algorithm can be considered to be a direct optimization method in which weights are automatically constructed to enable definitions of application-specific mesh quality. The high-level concepts of the paradigm have been implemented in the Mesquite mesh-improvement library, along with a number of concrete algorithms that address mesh quality issues such as those shown in the examples of the present paper.

1 Motivation

A significant fraction of modeling and simulation software provides numerical solutions to partial differential equations via discretization methods. Not only do the equations themselves need to be discretized, but so does the domain on which the problem is defined. PDE software often uses mesh generation for this task. Sometimes the meshes are generated off-line before the PDE simulation and sometimes the meshes are generated dynamically or adaptively as the calculation proceeds. In either case, the quality of the mesh is an important consideration because poor quality can impact accuracy, efficiency, and in the worst case, can invalidate or prematurely terminate the calculation. There are a variety of mesh generation methods, some giving better quality than others, depending upon the circumstances. Unstructured hexahedral mesh generation is probably the least successful in terms of guaranteeing quality, but tetrahedral and structured methods can also create meshes with

quality issues. As a result, a considerable number of post-processing techniques have been devised to improve the quality of an existing mesh. Some of these methods change mesh topology, some change mesh vertex positions, and some change both. Because the methods for changing mesh topology and for changing vertex positions are quite different, it is wise, in the interest of progress, to investigate the best possible technologies in each of these areas, both separately and together. In the present work, a new paradigm for changing mesh vertex coordinates via numerical optimization is described.

The Target-matrix Paradigm has evolved gradually over the past four years; forerunners of the paradigm can be found in [25], [26], and [27]. With funding provided by DOE's Office of Science, the Paradigm has in the last four years been formalized and has undergone rapid elaboration to ensure that it is well-formulated, complete, and powerful. Unfortunately, there is not space in this paper to describe all of the mathematical details in the Paradigm; this is done elsewhere in a series of reports spanning over two hundred pages [16]-[24].[1] Development of the Paradigm is on-going, with additional emphasis now on demonstrations of its capabilities [28]. Our purpose in this paper is to acquaint the reader with the basic motivation for the Target-matrix Paradigm (this section) and with the high-level concepts it entails (next section). To illustrate the potential of the paradigm, this paper ends with some examples of canonical problems to which it has been applied, with some success.

Our research into mesh quality optimization takes into account several aspects of the mesh quality problem, beginning with the fact that the node-movement algorithms are intended to be permanent fixtures in the Mesquite mesh quality software library [1]. Since Mesquite must be able to deal with a wide variety of mesh types (e.g., structured, unstructured, 2D, 3D, surface, simplicial, non-simplicial, hybrid, meshes with hanging nodes, polygonal or polyhedral elements, and/or high-order nodes), we seek algorithms that are largely independent of mesh type. This avoids having to write a lot of special case code.

Moreover, Mesquite must also be able to handle a wide variety of meshing contexts (e.g., as a stand-alone post-processor, as a library linked to either mesh generation software or to a physics application code, both serial and parallel mesh optimization, meshes with or without associated geometry, and so on). It is thus desirable that the optimization algorithms be transparent to these contexts, as far as possible.

It is not claimed that the present algorithms (or the Mesquite software) have fully addressed all these mesh types and contexts as yet, but rather that these requirements have been kept in mind during the course of the research and that this has already enabled mesh optimization in a wider variety of contexts than has been previously achieved. Example 3.4, given later in the paper shows, for instance, that it is possible using the new approach to

[1] Eventually these reports will become either archive journal papers or part of a monograph on mesh optimization.

smooth meshes consisting of quadratic finite elements. The draw-back to all this flexibility, of course, is that the implementations in the Mesquite code, while efficient, are not as efficient as they could be if they were to address only a single mesh type or a single context. However, flexibility is consistent with Mesquite's dual mission to serve both as a vehicle for mesh optimization research and as a mesh improvement service for applications. In the latter case, it is convenient to have Mesquite available so that potential solutions can be tried quickly. If a Mesquite algorithm proves successful on the application problem, one can either accept the inefficiency and move forward (this happens surprisingly often), or, one can re-code the particular algorithm in their own code, in a more efficient, but less flexible manner (this is possible since Mesquite is open source and because many of its algorithms are or will be documented in various publications).

Because meshes are used in many different physical applications such as heat transfer, fluid dynamics, structural mechanics, electro-magnetics, and many others, mesh quality issues can arise within each of these areas. Thus, mesh optimization is a cross-cutting technology in the sense that it can be applied to all of these areas, often with good effect. At the same time, mesh optimization is not a monolithic technology in the sense that there is only one mesh quality issue or only one issue that can potentially be addressed. In fact, mesh optimization can and has addressed a variety of mesh quality issues, either separately or together. For example, while each of the applications mentioned above may benefit from 'shape improvement,' they may also benefit from mesh optimization algorithms having different purposes such as 'smoothness', size-adaptivity, anisotropic-adaptivity, mesh updating on deforming geometric domains, ALE rezoning, sweep-smoothing, and other such *canonical* mesh quality issues. Thus, a second aspect of mesh quality that this research considers is how to address the wide variety of canonical mesh quality issues via mesh optimization techniques, within a unified paradigm.

Finally, it is evident from the previous paragraphs that to address all of these mesh types, contexts, and canonical problems within a single code such as Mesquite could potentially create a disparate collection of algorithms and data structures, resulting in both a maintenance headache and considerable conceptual uncertainty as to how the algorithms are related, not to mention whether they are all well-posed. The third aspect of this research is thus to alleviate this problem by developing and analyzing a mathematical framework for mesh optimization that develops this subject systematically, and in full generality, so that one has a coherent way of thinking about many of these contexts and problems. Moreover, the framework should enable relatively rapid delivery of solid new algorithms as they are needed. The framework that we propose is called the Target-Matrix (Optimization) Paradigm (or TMOP for

short)[2] As one might guess from the name, certain matrices called 'targets' play a central role in the theory.

The Target-Matrix Paradigm is closely related to the class of mesh optimization methods known as 'direct methods'. A 'direct optimization' method is one in which the formulation is made directly in terms of quantities representing mesh entities: length of edges, area/volume of elements, angles, and so on. Examples of other direct methods include [2], [3], [4], [5], and [6], among others. In contrast, indirect mesh optimization methods are formulated in the continuum, such as variational grid generation and partial differential equation methods (e.g., [7], [8], [9], [10]). There are some advantages to the direct methods which have attracted our attention to this approach. In practice, meshes are never taken to the limit of the continuum; one is always working with entities in a finite dimensional space. This suggests that one construct mesh optimization theories in the space in which the mesh itself lives. Moreover, finite dimensional spaces are much easier to deal with in mathematics than with infinite dimensional spaces; therefore theoretical progress can sometimes be much faster in the former case. As another consideration, while some indirect methods have associated proofs that show the solutions to the continuum equations give mappings which are everywhere invertible (and this is an excellent result) it does not necessarily mean in practice that the finite dimensional meshes that one works with will be non-inverted, because the proof only holds in the asymptotic limit [11]. On the other hand, if an analogous result were to be proved in a finite dimensional space, such as that provided by direct methods, then it would apply to the meshes that are used in practice. Although TMOP can be classified as a direct method, it differs strongly from other direct methods in that it aims at the full spectrum of mesh types, contexts, and canonical problems. As will be seen in the next section, to address this need for flexibility, TMOP has a number of unique features.

The term 'paradigm' is used here in the sense that a certain *world-view* or way-of-thinking about mesh optimization is taken that involves the use of Target-matrices to describe the optimal mesh. This approach is considerably different from other direct node-movement methods which, for the most part, have failed to fully exploit the idea of target or weighting functions.[3] However, there are other optimization methods which can be considered paradigms in the same sense as we use here. Most of these other paradigms come from early papers on the Indirect Methods. Prime examples of other mesh optimization paradigms include Harmonic Maps [13], the elliptic grid generators [14], and Laplace-Beltrami systems [15]. The idea in each of these paradigms is the same: address a variety of canonical mesh quality issues within a sin-

[2]Regrettably, the natural acronym for the Target-matrix Paradigm is TMP, which has the connotation of temporariness, which we hope is not the future of this method.

[3]There also exist methods of mesh *adaptation* via mesh modification which use metric tensor weightings (see [12] for a general discussion).

gle theory, through the use of weighting functions, scalars, and/or matrices. The emphasis in all these paradigms is not on quality metrics (they primarily use only one metric), but on the construction of the weighting functions. TMOP uses the same idea, but differs from these other paradigms in several respects. First, TMOP is a direct method, so that the results hold on discrete meshes, not just in the asymptotic limit. Second, the method is transparent to mesh type. Third, the theory of target-matrix construction is well-developed in TMOP and contains some new ideas on how to do the constructions. Since many of the older paradigms were proposed prior to unstructured meshing, and prior to advances in computer science, they are somewhat out of date with respect to current meshes, contexts, and canonical problems. In contrast, TMOP addresses many mesh quality issues in a modern setting.

2 Review of the Target-matrix Paradigm

The present section constitutes an introduction to the high-level concepts of TMOP, their purpose, how they are related, and how they meet the need for flexibility.

As in the Finite Element Method, one begins the theory by defining a set of local mappings from points Ξ in the master element(s) to points $X(\Xi)$ belonging to the elements of the mesh that is to be optimized (the latter is called the *active* mesh). The mappings are most commonly those from linear finite elements, but can be more general if needed. If the active mesh consists of only one element type, then the mappings can all have the same functional form, e.g., the bilinear map from a square to a quadrilateral. If the active mesh contains more than one element type (e.g., triangles and quadrilaterals) then more than one mapping form is required. Although the form of the mapping may be the same from one element to another, the exact mapping on each element can differ because the mapping depends on the coordinates of the vertices which define the particular element. Non-linear mappings are also allowed, for example, in the case of high-order finite elements. Thus, every element in the mesh has an associated mapping; in Mesquite these mappings are defined by default so that usually the user has very little work to do. The fact that TMOP most often uses finite element mappings to measure and control mesh quality does not mean that the application must be a finite element calculation.

In addition to the mappings, TMOP requires that a set of *sample points* within the master element(s) be defined. Let the sample points within the master element be denoted by $\{\Xi_k\}$, $k = 0, 1, \ldots, K-1$. The corresponding points in the active mesh elements are $\{X_k\}$ where $X_k = X(\Xi_k)$. Typically, the sample points are located at the corners of the master element if the element is linear, otherwise they may also be located, for example, at mid-edges, mid-faces, and/or mid-elements. TMOP thus requires that, in the formulation

stage of the optimization, one define a set of mappings and sample points over all the elements of the mesh. This is not as daunting as it sounds because the mappings are usually of the same form for each element in the mesh (unless it contains more than one element type) and thus there is only one master element that is used for every element in the active mesh. The sample points are usually located at the corners of the master element if the mapping is linear, so most of the time, there is little need for user input here.

The mappings are required to be differentiable so that their Jacobian matrix $\partial X/\partial \Xi$ exists at the sample points. For short-hand, we denote this Jacobian matrix by the symbol A, which refers to the Jacobian of the map from the master element to an element in the *active* mesh. For non-simplicial elements, the Jacobian matrix varies from point to point within the master element, as a function of Ξ. We denote by A_k the Jacobian matrix evaluated at sample point k; thus $A_k = A(\Xi_k)$ is the active Jacobian matrix at a sample point. The matrix is indirectly a function of the coordinates of the vertices which define the mapping. The Jacobian matrix also varies from one element to the next; for clarity, the element dependence has been suppressed in the notation used above. Note that these matrices are very small; for a 2D mesh element, the active matrix is just 2×2. Mappings, sample points, and the local Jacobian matrix within TMOP are discussed more fully in [16].

Target matrices play a critical role in TMOP because they define the *desired* Jacobian matrices in the optimal mesh. Target matrices are not optional, but *must* be constructed prior to the mesh optimization step in order to obtain a well-posed problem. Targets force the user/customer to supply a high-level definition of quality, derived from one of the canonical mesh quality issues (like those mentioned in the previous section). For every sample point k in the mesh, the target paradigm thus requires two matrices, forming a pair: the Jacobian matrix A_k derived from the active mesh and the Target (or reference-Jacobian) matrix W_k. Because every sample point can have a different Target-matrix, supplying this information would seem to constitute an enormous burden on the user. However, TMOP describes various automatic *target construction* algorithms that take the high-level quality definition and, using other available data, create the low-level target-matrix datasets (see [22]). This is possible, in part, due to the use of the QR-factorization, which enables one to separate out quality into four matrix factors called Size, Orientation, Angle, and Aspect Ratio. As with other mesh optimization paradigms, construction of targets remains somewhat of an art, but in TMOP is made more tractable for two reasons. First, the targets are based on the Jacobian matrix of the desired optimal mesh and thus have a simple geometric interpretation. Second, the target construction method can make partial use of the initial mesh (the one to be optimized). The initial mesh is nearly always available, and, in most other optimization methods, is ignored, even though it often contains valuable information.

For clarity, the sample point indices are often suppressed in much of the remainder of this presentation. Let A and W at some sample point be defined.

Because the construction of targets is under our control, we can assume that, for every target, $det(W) \neq 0$ and thus W^{-1} exists. The *weighted Jacobian* matrix T, defined by $T = AW^{-1}$, is heavily used in TMOP because, if the target matrix W has units of length (as does A), then T is non-dimensional and provides a convenient scaling of A.

Next, a set of local quality metrics is needed. Let M_d be the set of $d \times d$ matrices with real numbers as elements. To this point the matrices A, W, and T in the paradigm are either 2×2 or 3×3, reflecting the dimension of the elements in the mesh. A local quality metric $\mu = \mu(T)$, which is a function from M_d to the non-negative numbers, measures the relationship between A and W. Several particular relationships are of interest

- $A = W$ or, equivalently, $T = I$. When the two Jacobian matrices satisfy this relationship, the active matrix matches the target-matrix. In that case, the Size, Shape (i.e., Angle and Aspect Ratio), and Orientation properties of the Target-matrix also reside in the active-matrix. Local metrics which attempt to enforce this relationship, such as $\mu = |T - I|_F^2$, are thus called Size+Shape+Orientation metrics.[4] The relationship is enforced because $\mu \geq 0$, so that $\mu = 0$ is the ideal value. In fact, $\mu = 0$ if and only if $T = I$.
- $A = RW$ or, equivalently, $T = R$, with R a non-specified rotation matrix. When the Jacobians satisfy this relationship, the Size and Shape properties of the Target-matrix are also found in the active-matrix, while the orientation is not in general. Local metrics which attempt to enforce this relationship, such as $\mu = |T^t\, T - I|_F^2 + (det(T) - 1)^2$, are thus called Size+Shape metrics. The relationship is enforced because $\mu \geq 0$, so that $\mu = 0$ is the ideal value. In fact, $\mu = 0$ if and only if $T = R$.
- $A = sRW$ or, equivalently, $T = sR$, with R a non-specified rotation matrix, and s is a non-specified positive scalar. When the Jacobians satisfy this relationship, only the Shape properties of the Target-matrix reside in the active matrix, while Size and Orientation may not. Local metrics which attempt to enforce this relationship, such as $\mu = |T|_F\, |T^{-1}|_F$ (condition number), are thus called Shape metrics. The relationship is enforced because $\mu \geq d$ when $T \in M_d$, so that $\mu = d$ is the ideal value. In fact, $\mu = d$ if and only if $T = s\, R$.

Depending on which of the canonical mesh quality issues one wants to address, one may want to use a Shape Metric, a Size+Shape metric, or a Size+Shape+Orientation metric. In any case, these are the main quality metrics used in TMOP, along with their barrier forms (which are used to enforce positive Jacobian determinants). Thus TMOP focuses mainly on Target construction and not so much on devising different quality metrics. Properties of these local metrics, such as convexity, are studied in [17], [18], and [19].

Putting this all together, an objective function, typically of the form

[4] $|\cdot|_F$ is the Frobenius matrix norm.

$$F = \frac{1}{N} \sum_{n=1}^{N} \mu(T_n)$$

(where n is the sample point index and N is the total number of sample points), is minimized as a function of the coordinates of the free vertices to find the optimal mesh. The optimization is often constrained by fixing the position of some or all of the boundary vertices. In other cases, boundary vertices may be moved in directions tangential to the under-lying geometric curve or surface. The sum in the objective function can include all the sample points in the global mesh or only those within a local patch (if one is doing local patch smoothing).

A generalization of the objective function above, based on the Power Mean, permits better scaling, as compared to using an ℓ_p norm [16]. Part of the generalization includes *trade-off coefficients* which can be used to emphasize local quality in one location in the mesh more than in another [24].

Surface mesh optimization in TMOP requires special treatment since the active Jacobian matrices are not square (they are 3x2) and thus lack determinants, traces, and other properties used in the 2D and 3D element theories. Research on this topic is found in [23].

The above optimization problem, applied to most meshes, cannot usually be solved without resorting to iterative numerical methods. For the most part, standard methods for large-scale multi-variable optimization of continuous variables are used to solve the TMOP optimization problem.

This completes the high-level description of the paradigm. Comparing TMOP to other direct methods for mesh optimization, one sees that

1. The other methods do not use mappings or sample points. Therefore, they cannot easily deal with the wide variety of mesh and element types, nor can they avail themselves of the fundamental object in meshing, namely the Jacobian matrix.
2. The other methods fail to make significant use of target-matrices or weighting functions. Therefore, they cannot easily address more than one canonical mesh quality issue. Most of the previous direct methods were aimed at controlling mesh smoothness, area, and/or angles (or some combination thereof) and cannot be applied to the examples shown in the next section.
3. The other methods do not use matrices and thus cannot take advantage of the extensive mathematical and numerical theories of matrices which have been developed over the past two centuries. In contrast, TMOP uses ideas from matrix factorizations, matrix norms and inequalities, inverses, eigenpairs, special matrix types, and more.
4. The other methods fail to significantly use scaling methods to properly scale their metrics and objective functions, whereas TMOP does this via the Power Mean, the use of AW^{-1}, and trade-off coefficients.

5. While other direct methods must combine several objective functions together in order to achieve simultaneous control of length, area, and angles, TMOP can do this using only one objective function because it controls all these propoerties via a single target-matrix. Thus TMOP avoids the need to determine the proper weighted combinations of the different objective functions (which remains an open question) in order to properly scale the problem.
6. The other direct methods fail to use trade-off coefficients and thus cannot trade off quality in one part of the mesh against another part.
7. The set of canonical mesh quality issues addressed by TMOP is much broader than in the other direct methods. For example, while the latter mainly focus on smoothness, area, and angles, TMOP can additionally focus on mesh adaptivity, anisotropic smoothing, mesh alignment with vector fields, ALE rezone, and considerably more.

Although the other direct methods can sometimes be well-suited to particular problems, they are much less flexible and powerful than TMOP, as one would expect since they do not use the concepts described in this section.

3 Selected Application Examples

In this section, numerical examples are given to illustrate some of the diversity of mesh quality issues that TMOP can address. One can also consult [28] for an example involving deforming geometry.

3.1 Shape Smoothing of a Surface Mesh

One of the simplest applications of TMOP is that of improving the shape of mesh elements, shape being a combination of element angles and aspect ratios. The initial mesh shown on the left side of Figure 1, consists of triangular elements on a sphere. The elements of the initial mesh vary in shape and size. The shape-improvement goal is to make all of the elements as close to equilateral as possible. This is accomplished in TMOP by (1) using a Shape metric (such as condition number) and (2) constructing the Target-matrix which represents an equilateral triangle.

$$W = \begin{pmatrix} 1 & \frac{1}{2} \\ 0 & \frac{\sqrt{3}}{2} \end{pmatrix}$$

The target-matrix in this particular problem is the same at each sample point of the mesh and the sample points are located at the centroid of each element. The linear triangle map is used. Because the Shape metric is Size+Orientation invariant, one needn't consider scaling of the target to achieve any particular

edge-length nor does one need to consider orientation. A series of transformations described in [23] converts the 3×2 Jacobian matrix A into a 2×2 matrix so that it can be combined with $W_{2\times 2}$ in order to form $T_{2\times 2}$. The optimal mesh on the right in Figure 1 shows the result of optimization with this target - the elements are nearly equilateral.

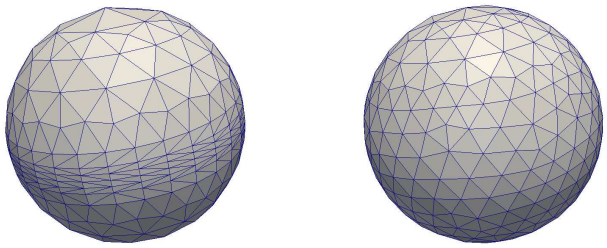

Fig. 1. *Triangle Mesh Shape Improvement on a Sphere:* Initial Mesh (Left), Optimal Mesh (Right)

3.2 Sliver-removal for a viscous CFD Mesh

Figure 2 shows a tetrahedral mesh created by the VGRID code [29] for a CFD problem involving a viscous boundary layer.[5] A closeup showing the boundary layer mesh is given on the right side of the picture. The mesh has good quality in the boundary layer, but unfortunately contains sliver elements in the far-field. The goal in this problem is to improve the shape of the sliver elements while preserving the boundary layer mesh, and retaining the good elements in the size-transition region between the boundary layer and far-field. Elements in the optimal mesh should be non-inverted. Boundary vertices are permitted to move while constrained to their geometry.

In this problem there are two goals, one to *preserve* part of the initial mesh and one to *create* better-shaped elements in another part of the initial mesh. Target-matrices for preserving the initial mesh are easy to construct: simply set $W_k = (A_k)_{initial}$ and use the metric $\mu(T) = |T - I|_F^2$. Likewise, Target-matrices corresponding to equilateral tetrahedra are easy to construct, and these, along with a Shape metric will tend to create well-shaped tetrahedral elements. The main optimization issue is then: how to blend these two different sets of Target-matrices in various portions of the mesh? Our solution is not to blend the matrices, but rather to blend two quality metrics. Thus, the objective function in this problem was

[5] All figures in this example courtesy of Jan-Renee Carlson, NASA-Langley.

$$
\begin{aligned}
F &= F_1 + F_2 \\
F_1 &= \sum_k c_k \left(\frac{|T_1|_F^3}{3\sqrt{3}\, det(T_1)} - 1 \right) \\
F_2 &= \sum_k (1 - c_k)\, |T_2 - I|_F^2 \\
T_1 &= A(W_1)^{-1} \\
T_2 &= A(W_2)^{-1}
\end{aligned}
$$

where F_1 is the shape-improvement term and F_2 is the preservation term. W_1 corresponds to the equilateral tetrahedron and W_2 is based on the initial mesh, as described above and c_k is a trade-off (or blending) coefficient. To be precise,

$$c_k = \frac{1}{1 + e^{-0.4394(d_k - 135)}}$$

where d_k is the maximum dihedral angle (in degrees) in the k^{th} tetrahedral element (and sample point) of the mesh.[6] The trade-off coefficient forms a logistic curve in the dihedral angle; it is nearly zero when the dihedral angle is less than 135 degrees and nearly one when the dihedral angle is greater than 135. Thus, shape improvement will tend to be emphasized in elements with large dihedral angles, while preservation will be emphasized on the others. Figure 3 (left side) explains why this particular functional form was chosen: a histogram plot showing the number of occurrences of the pair (maximum, minimum) dihedral angle within the mesh was created by Carlson. Examination of this plot showed that sliver elements tended to occur when the maximum dihedral angle exceeded 135 degrees, whereas the viscous elements occurred for the combination of small minimum and small maximum dihedral angle. Use of such a plot allowed us to avoid having to construct a blending function based on spatial coordinates of the meshed regions (a considerably more difficult endeavor).

With this construction, F_1 (shape-improvement) is emphasized on the large max. dihedral angle elements, and F_2 (preservation of initial mesh) elsewhere. Figure 3 (right side) shows two scatter plots overlaid, one for the initial mesh, and one for the optimal mesh. The plot shows that the sliver element region is nearly vacant in the optimal mesh. Moreover, the boundary layer mesh was preserved during the optimization. This solution was not hit upon immediately, but the flexibility of TMOP allowed us to try out a number of ideas fairly quickly as the described approach was developed. As this example shows, choosing the blending functions is as much of an art as is constructing target-matrices. Never-the-less, it was possible to find a suitable function after a modest amount of experimentation.

[6] The constant 0.4394 was determined by requiring that $c_k = 0.9$ when $d_k = 130$ degrees.

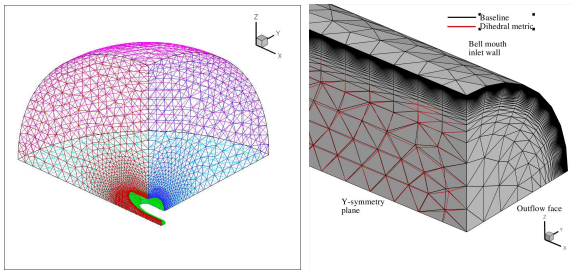

Fig. 2. *Initial Tetrahedral CFD Mesh from VGRID:*
Whole Domain (Left), Zoom to Boundary Layer Mesh (Right)

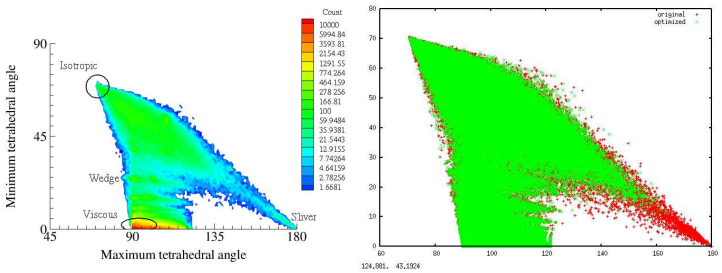

Fig. 3. *Maximum vs. Minimum Dihedral Angle Scatter Plots:*
Initial Mesh Showing Viscous and Sliver Regions (Left), Overlay of Initial and Optimal Meshes (Right)

3.3 Optimizing the Quality of Higher-order Node Meshes

Higher-order finite elements are defined via nodes/vertices located not only at the usual element corners, but also at mid-edge, mid-face, and mid-element locations. The addition of these extra nodes enables the use of quadratic and higher-order mappings that allow greater accuracy for the same number of elements. In terms of mesh generation, higher-order finite element meshes are frequently generated by first creating linear element meshes and subsequently adding the extra nodes. For interior elements, the extra nodes are placed at the geometric mid-point of an edge, face, or element. For boundary elements, however, the extra nodes must conform to the under-lying geometry in order to maintain higher-order accuracy. To do this, mesh generation codes often 'snap' the extra nodes near the boundary to the geometry via a normal projection operation. Usually this works fine but, if the mesh is coarse compared to the local geometric curvature, so-called 'inverted' elements can be created by the projection step. An example of this is shown in Figure 4 (left side): all of the triangles have straight sides except those on the boundary. Some of the boundary triangles have curved edges which cross the domain boundary

and are thus inverted (see triangle on bottom boundary just to the right of center). Such inverted elements must be removed or fixed before the finite element simulation can proceed.

There are various ways to fix the inverted elements in higher-order meshes, including re-meshing, refinement, node-insertion, and element-swapping (see, for example, [30]). Missing from this list, until now, is mesh optimization via higher-order node-movement. In this context, TMOP can attack this problem via the use of sample points. In the example given here, sample points were placed both at the three element corners and at the three mid-edge locations of every element in order to control the active Jacobian matrix at these locations. The goal is to optimize the initial mesh via movement of the mesh node and vertex coordinates at all of the sample points. The result, of course, will be a triangle mesh having curved sides, even in the interior.

A two-stage optimization procedure was used. In the first stage we sought to produce a non-inverted mesh. This was accomplished using a Target-matrix corresponding to an equilateral triangle whose area was the same as the area of a triangle in the initial mesh (they all have roughly the same area). The local quality metric, corresponding to equal-area elements, was $\mu(T) = [det(T) - 1]^2$. This metric can often, but not always, untangle a mesh. Luckily, on this problem, the optimization in the first stage did produce a non-inverted mesh (see Figure 4 - Center). The elements in the non-inverted mesh have highly-curved edges and the mesh is probably unsuitable for a finite element calculation. However, since the mesh is non-inverted, this mesh could be used as the input to the second stage of the optimization, in which we used a barrier-based Shape metric (condition number) to improve shape while keeping the mesh untangled. The result is shown on the Right of the Figure. Elements in the optimal mesh in the second stage are well-shaped and the mesh is non-inverted; element sides are slightly curved.

This example shows the potential of TMOP to play a role in improving quadratic and higher-order finite element quality. More details on this work-in-progress can be found in [31].

Fig. 4. *Optimization of Triangle Meshes with Quadratic Finite Elements:* Left: Initial 'Inverted'; Center: 'Non-inverted'; Right: Well-shaped

3.4 Rapid Delivery of a Custom-Built Smoother

The Target-matrix Paradigm contains a number of high-level concepts such as active and target matrices, local quality metrics, objective functions and such, that translate nicely into objects and classes within the Mesquite code. These have been implemented in Mesquite without having to provide the low-level details needed to complete any specific mesh optimization algorithm such as the ones mentioned in the examples here. As a result, it is possible to create new mesh optimization methods quite rapidly in response to requests from different application groups. Here, an example is given which came from the Cubit meshing group. Cubit frequently is used to create unstructured quadrilateral meshes via a paving algorithm (see initial mesh on the left of Figure 5). Although this mesh has nice shape-quality, in which the elements are nearly square, the application group complained that the mesh contained unnecessarily short edges. That is, some edges were about half as long as other edges. Because the application group was using an explicit simulation code, these short edges were determining the time-step that was used in the simulations and thus, the time-step was needlessly small (the Courant condition, which was used to determine the time-step, is proportional to the minimum edge-length in the mesh). To fix this issue, Mesquite was asked to optimize the initial mesh in such a way that these overly short edges were lengthened.

No such smoother was available in Mesquite at the time of the request. What was clearly needed was a smoother that created equal-length edges in the mesh (so none would be needlessly small) and, at the same time, maintained the near-square shape of the elements. TMOP was able to provide this capability quite quickly by implementing a concrete target construction algorithm that created the appropriate set of Target-matrices and by selecting the right local quality metric. Sample points were located at the four corners of each quadrilateral and the bilinear map applied. Because each quadrilateral element in the optimal mesh was to be identical (except for orientation), the set of Target-matrices needed to be the same at each sample point. To meet the problem requirements, the Target was selected to have the form $W = \Lambda I$, where I is the 2×2 identity matrix and Λ is a positive scalar. The identity matrix represents an element whose aspect ratio (length/width) is 1.0 and whose angles are ninety degrees; this corresponds to the shape of a square quadrilateral. The scalar Λ is related to edge-length in the theory of Target-matrix construction (see [22]); by making it the same at all sample points, the optimal mesh should have equal-length edges. Specifically, the scalar was set to the average edge-length in the initial mesh (this computational capability was already available in Mesquite). Finally, because we did not wish to control the Orientation of the quadrilaterals in the mesh, we used an Orientation-invariant quality metric, namely Size+Shape. A concrete target construction algorithm was added, with minimal effort, to the Mesquite code to create a target of this particular form. The optimal mesh resulting from these choices is shown on the right side of the Figure; as one can see, the edge-lengths in the

optimal mesh are nearly equal, the elements nearly square, and the mesh is non-inverted. The new capability was delivered to the customer as a Mesquite wrapper (called from the Cubit code) in a matter of two weeks (of course, it won't always be this easy).

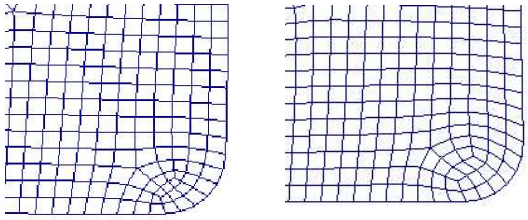

Fig. 5. *Optimization of Paved Mesh To Increase Time-Step:*

Left: Initial Mesh with 'Short' Edges; *Right*: Optimized Mesh with Equal Edges

4 Summary

The motivation for the Target-matrix paradigm is derived principally from the desire to create a general purpose mesh optimization library known as Mesquite. A general purpose mesh optimization library must be able to handle the wide variety of mesh types one encounters in computational simulations. It must also be able to provide solutions to the canonical problems in mesh quality, for example, mesh untangling, shape-improvement, size-adaptation, mesh alignment, and more. Without a general mesh optimization paradigm, these requirements could only be met via an incompatible collection of algorithms that might only be loosely related, resulting in a loss of flexibility and a software maintenance challenge. The Target-matrix paradigm unifies many of these requirements through the sample point concept (to handle mesh type) and via the target-matrix concept (to address different canonical problems). High-level concepts in TMOP were summarized. A rich mathematical theory underlying these concepts is under development. The high-level concepts have been clarified and made more precise, and many important details in target-construction and the like are described in a series of reports. The numerical examples given here illustrate the potential of TMOP to impact many canonical problems in mesh quality improvement via node movement. Future work will expand the list of canonical problems, mesh types, numerical examples, concrete automatic target-construction algorithms, and the set of high-level wrappers in Mesquite. Moreover, important extensions of the mathematical

theory will be developed.

Acknowledgments Many thanks to Jan-Renee Carlson, Jason Kraftcheck, and Nick Voshell for their important contributions to Mesquite and to the numerical examples.

References

1. M. Brewer, L. Diachin, P. Knupp, and D. Melander, *The Mesquite Mesh Quality Improvement Toolkit*, p239-250, Proceedings of the 12th International Meshing Roundtable, Santa Fe NM, 2003.
2. Castillo, J.E. *A discrete variational grid generation method*, SIAM J. Sci. Stat. Comp., Vol. 12, No. 2, p454-468, 1991.
3. Tinoco-Ruiz, J. and Barrera-Sanchez, P. *Area functionals in Plane Grid Generation*, in Numerical Grid Generation in Computational Field Simulations, M. Cross et. al. (eds.), p293-302, Greenwhich UK, 1998.
4. Kennon, S. and Dulikravich, G. *Generation of computational grids using optimization*, AIAA Journal, Vol. 24, No. 7, p1069-1073, 1986.
5. Freitag, L. *On combining Laplacian and optimization-based mesh smoothing techniques*, p37-43, AMD-Vol. 220, Trends in Unstructured Mesh Generation, ASME 1997.
6. Zhou, T. and Shimada, K. *An angle-based approach to two-dimensional mesh smoothing*, p373-384, Proceedings of the 9th International Meshing Roundtable, 2000.
7. Brackbill, J. and Saltzman, J. *Adaptive zoning for singular problems in two dimensions*, J. Comp. Phys., Vol. 46, p342-368, 1982.
8. Steinberg, S. and Roache, P. *Variational grid generation*, Num. Meth. for P.D.E., Vol. 2, p71-96, 1986.
9. Liseikin, V. *On a variational method of generating adaptive grids on n-dimensional surfaces*, Soviet Math. Docl., Vol. 44, No. 1, p149-152, 1992.
10. Winslow, A. *Numerical solution of the quasilinear Poisson equations in a nonuniform triangle mesh*, J. Comp. Phys., Vol. 2, p149-172, 1967.
11. Knupp, P. and Luczak, R. *Truncation error in grid generation*, Num. Meth. P.D.E., p561-571, Vol. 11, 1995.
12. Frey, P. and George, P. *Mesh Generation: Application to Finite Elements*, Wiley, 2008.
13. Dvinsky, A. *Adaptive grid generation from harmonic maps on Riemannian manifolds*, J. Comp. Phys., Vol. 95, p450-476, 1991.
14. Thompson, J. Warsi, Z. Mastin, C. *Automatic numerical generation of body-fitted curvilinear coordinate systems*, J. Comp. Phys., Vol. 24, p274-302, 1977.
15. Liseikin, V. *A computational differential geometry approach to grid generation*, Springer-Verlag, 2004.
16. P. Knupp, *Formulation of a Target-Matrix Paradigm for Mesh Optimization*, SAND2006-2730J, Sandia National Laboratories, 2006.
17. P. Knupp, *Local 2D Metrics for Mesh Optimization in the Target-matrix Paradigm*, SAND2006-7382J, Sandia National Laboratories, 2006.

18. P. Knupp and E. van der Zee, *Convexity of Mesh Optimization Metrics Using a Target-matrix Paradigm*, SAND2006-4975J, Sandia National Laboratories, 2006.
19. P. Knupp, *Analysis of 2D Rotational-invariant Non-barrier Metrics in the Target-Matrix Paradigm*, SAND2008-8219P, Sandia National Laboratories, 2008.
20. P. Knupp, *Measuring Quality Within Mesh Elements*, SAND2009-3081J, Sandia National Laboratories, 2009.
21. P. Knupp, *Label-invariant Mesh Quality Metrics*, pp139-155, Proceedings of the 18th International Meshing Roundtable, Springer, 2009.
22. P. Knupp, *Target-matrix Construction Algorithms*, SAND2009-7003P, Sandia National Laboratories, 2009.
23. P. Knupp and J. Kraftcheck, *Surface Mesh optimization in the Target-Matrix Paradigm*, manuscript.
24. P. Knupp, *Tradeoff-coefficient and Binary Metric Construction Algorithms within the Target-Matrix Paradigm*, manuscript.
25. P. Knupp, *Algebraic Mesh Quality Measures*, SIAM J. Sci. Comput., Vol. 23, No. 1, pp193-218, 2001.
26. L. Freitag, P. Knupp, *Tetrahedral mesh improvement via optimization of the element condition number*, Intl. J. Numer. Meth. Engr., 53:1377-1391, 2002.
27. P. Knupp, L. Margolin, and M. Shashkov, *Reference-Jacobian Optimization-based Rezone Strategies for Arbitrary Lagrangian Eulerian Methods*, J. Comp. Phys., p93-128, Vol. 176, No. 1, 2002.
28. P. Knupp, *Updating Meshes on Deforming Domains*, Communications in Numerical Methods in Engineering, 24:467-476, 2008.
29. http://tetruss.larc.nasa.gov/vgrid/
30. X. Luo, M. Shephard, L. Lee, L. Ge, and C. Ng, *Moving Curved Mesh Adaptatio for Higher Order Finite Element Simulations*, Engr. w/Cmptrs., published online, 27 Feb. 2010.
31. P. Knupp, N. Voshell, and J. Kraftcheck, *Quadratic Triangle Mesh Untanglng and Optimization via the Target-matrix Paradigm*, CSRI Summer Proceedings, SAND2010-3083P, Sandia National Laboratories, 2009.