Tangle-Free Finite Element Mesh Motion for Ablation Problems

Justin Droba*

NASA Johnson Space Center, Houston, TX, 77058, USA

JSC Engineering, Technology, and Science (JETS): Jacobs Technology and HX5, LLC

Mesh motion is the process by which a computational domain is updated in time to reflect physical changes in the material the domain represents. Such a technique is needed in the study of the thermal response of ablative materials, which erode when strong heating is applied to the boundary. Traditionally, the thermal solver is coupled with a linear elastic or biharmonic system whose sole purpose is to update mesh node locations in response to altering boundary heating. Simple mesh motion algorithms rely on boundary surface normals. In such schemes, evolution in time will eventually cause the mesh to intersect and "tangle" with itself, causing failure. Furthermore, such schemes are greatly limited in the problems geometries on which they will be successful. This paper presents a comprehensive and sophisticated scheme that tailors the directions of motion based on context. By choosing directions for each node smartly, the inevitable tangle can be completely avoided and mesh motion on complex geometries can be modeled accurately.

I. Introduction

Problems involving mesh motion—which are not to be confused with moving mesh methods, a class of adaptive mesh redistribution techniques designed to increase solution accuracy at low cost—arise naturally in the study of the thermal response of ablative materials. Ablation is the process by which a surface erodes by conversion of mass to another state of matter, most often from solid to gas (sublimination). While mass is conserved when the entire system is considered, from the perspective of the solid, the mass loss is absolute: no residue or accumulation is left, such as when a candle melts. Because the shape of the material changes in time, so does the computational domain. This is the goal of mesh motion: independent of the thermal solver, update the computational mesh to model its evolution in time.

A detailed discussion of the physics and modeling of ablation is beyond the scope of this paper. Instead, we assume that the total mass loss Δm is known. This quantity is related to the *mass loss rate due to ablation* $\dot{m}$ in the following manner:

$$\Delta m = \int_{t_0}^{t_f} \int_{\partial\Omega} \dot{m}\, dS\, dt \tag{1}$$

By solving a surface energy balance equation, the rate $\dot{m}$ can easily be obtained at surface quadrature points. Assuming a constant material density ρ, $\dot{m}$ can be converted to the surface recession rate $\dot{s}$:

$$\dot{s} = \frac{\dot{m}}{\rho}$$

$$\left[\dot{s}\right] = {}^{\mathrm{m}}/_{\mathrm{s}} \quad \left[\dot{m}\right] = {}^{\mathrm{kg}}/_{\mathrm{m\cdot s^2}}$$

Because the thermal response problem from which Δm is solved at discrete time steps, the time integral in (1) must be approximated. This paper assumes a fully explicit approach: the surface recession, determined at time level n, is taken constant until time level $n+1$. Under this assumption, the total change in surface position over a period of $\Delta t = t_{n+1} - t_n$ is

$$\Delta s = \dot{s}\Delta t$$

*Applied Aeroscience and CFD Branch, NASA Johnson Space Center, 2101 NASA Parkway, Houston, TX 77058; AIAA Member

With this quantity known, we can determine the overall mesh motion by assuming the solid responds to the strain of deformation by Δs linearly. The dynamics of displacement $\mathbf{u}$ of a position within the material (domain) Ω are governed by the linear elasticity equation

$$\left.\begin{aligned} \mathcal{L}\mathbf{u} &= \mathbf{f} & \mathbf{x} &\in \Omega \\ \mathbf{u} &= \big(\dot{s}\Delta t\big)\mathbf{d}(\mathbf{x}) & \mathbf{x} &\in \partial\Omega \end{aligned}\right\} \tag{2}$$

where $\mathbf{u}(\cdot) \in \mathbb{R}^d$ and the operator $\mathcal{L}$ is given by

$$\mathcal{L}\mathbf{u} \triangleq -\nabla\lambda(\nabla\cdot\mathbf{u}) - (\nabla\cdot\mu\nabla)\mathbf{u} - \nabla\cdot\mu(\nabla\mathbf{u})^T$$

In general, the homogenous case $\mathbf{f} \equiv 0$ is the only one of practical interest. The vector $\mathbf{d}$ indicates the direction of the surface motion. The variational problem corresponding to (2) is

$$a(\mathbf{u},\mathbf{v}) = \big\langle \mathbf{f},\mathbf{v}\big\rangle_{L^2(\Omega;\mathbb{R}^d)} \quad \forall\,\mathbf{v} \in H^1(\Omega;\mathbb{R}^d)$$

where the bilinear form is

$$\begin{aligned} a(\mathbf{u},\mathbf{v}) = {} & \lambda\int_\Omega (\nabla\cdot\mathbf{u})(\nabla\cdot\mathbf{v})\,d\mathbf{x} \quad + \mu\int_\Omega \big\langle \nabla\mathbf{u}^T,\nabla\mathbf{v}\big\rangle_{\mathbb{R}^{d\times d}}\,d\mathbf{x} + \mu\int_\Omega \big\langle \nabla\mathbf{u},\nabla\mathbf{v}\big\rangle_{\mathbb{R}^{d\times d}}\,d\mathbf{x} \\ & \lambda\int_{\partial\Omega} (\nabla\cdot\mathbf{u})\big\langle \mathbf{v},\boldsymbol{\nu}\big\rangle_{\mathbb{R}^d}\,dS - \mu\int_{\partial\Omega}\big\langle (\nabla\mathbf{u}^T)\mathbf{d},\mathbf{v}\big\rangle_{\mathbb{R}^d}\,dS - \mu\int_{\partial\Omega}\big\langle (\nabla\mathbf{u})\mathbf{d},\mathbf{v}\big\rangle_{\mathbb{R}^d}\,dS \end{aligned} \tag{3}$$

where we use the standard inner products

$$\begin{aligned} \big\langle \mathbf{A},\mathbf{B}\big\rangle_{\mathbb{R}^{d\times d}} = \operatorname{trace}(\mathbf{A}^T\mathbf{B}) \quad &= \sum_{i=1}^d\sum_{j=1}^d A_{ij}B_{ji} \\ \big\langle \mathbf{f},\mathbf{g}\big\rangle_{L^2(\Omega;\mathbb{R}^d)} = \int_\Omega \big\langle \mathbf{f},\mathbf{g}\big\rangle_{\mathbb{R}^d}\,d\mathbf{x} &= \sum_{i=1}^d \int_\Omega f_i(\mathbf{x})g_i(\mathbf{x})\,d\mathbf{x} \end{aligned}$$

Because the sole boundary condition in the problem is of Dirichlet type and will be enforced as an essential boundary condition, we require $\mathbf{v} \in H_0^1(\Omega;\mathbb{R}^d)$ so that the surface integrals of (3) all vanish.

We distinguish three disjoint subsets of $\partial\Omega$:

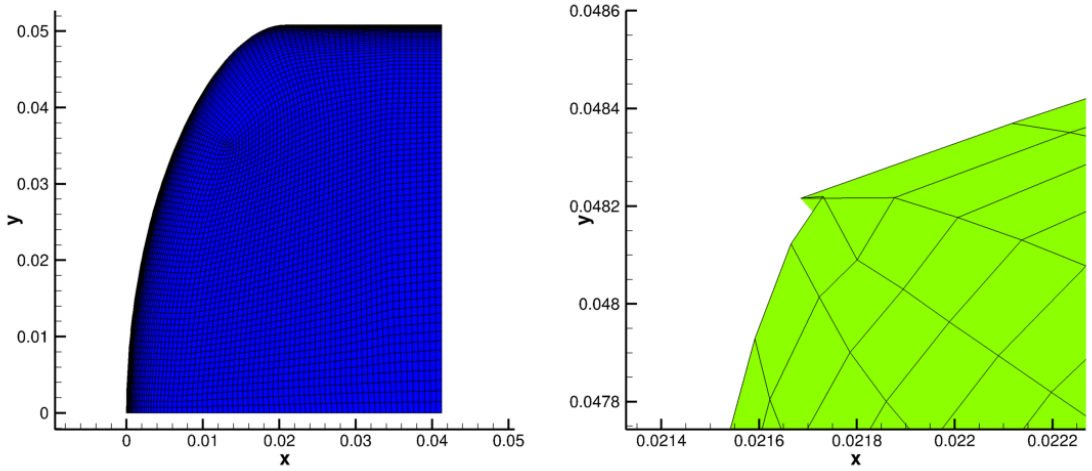
$\partial\Omega_R$:	Receding portion	Specified motion of Δs along $\mathbf{d}$
$\partial\Omega_S$:	Sliding portion	Motion in the normal direction forbidden
$\partial\Omega_F$:	Stationary (fixed) portion	No motion in any direction permitted

In a simple approach, the direction of motion $\mathbf{d}$ is taken to be the surface normal $\boldsymbol{\nu}$. While simple to implement, this has two unfortunate and undesirable consequences:

1. As we shall see in the coming section, it precludes sliding motion along any axis other than a coordinate one in a finite-element based formulation of (2).
2. On curved surfaces, the normal vectors from element to element are not parallel. This dooms the mesh to an inevitable collision with itself. An example is depicted in Figure 1 below.

This report presents a scheme for two- and three-dimensional meshes that simulatenously removes the first restriction while not suffering from the second drawback. We dub this method "tangle-free mesh motion."

In one-dimension, there is only one possible direction of motion: "normal" (left or right), always directed into the material. Tangling is impossible and sliding conditions do not exist, since prohibiting motion in the normal is equivalent to quashing all motion. As a result, mesh motion in one-dimension is fully resolved.



(a) **(b)**

Fig. 1: Collision Course. The starting geometry is depicted on the left in (a); here, $\partial\Omega_R$ is the curved part and the two flat sides (aligned with the x and y axes) comprise $\partial\Omega_S$. When aerodynamic heating is applied to $\partial\Omega_R$ of this ablating isoq, after sufficient simulation time, the result is (b). The material has receded and the mesh has become tangled. The thermal solve will fail shortly after this occurs.

II. Two-Dimensional Meshes

A. General Framework

Suppose Ω has been discretized with a finite element mesh $\mathfrak{T}$ and let $\mathcal{T} \in \mathfrak{T}$ be an element in this mesh such that $|\mathcal{T} \cap \partial\Omega|_{\mathbb{R}^{d-1}} > 0$; that is, the intersection with the boundary is nontrivial (more than a single point in two-dimensions). Let $\{\mathbf{x}_1, \cdots, \mathbf{x}_m\} \subset \partial\mathcal{T} \cap \partial\Omega$ and assume that $\{\mathbf{x}_{m+1}, \ldots, \mathbf{x}_N\} \cap \partial\Omega = \emptyset$. On $\mathcal{T}$, the finite element solution to (2) is a given by a representation in terms of local shape functions:

$$\mathbf{u}\big|_{\mathcal{T}} = \sum_{k=1}^{N} \alpha_k \psi_k(\mathbf{x})\mathbf{e}_1 + \beta_k \psi_k(\mathbf{x})\mathbf{e}_2 \tag{4}$$

As defined in finite element software packages (e.g., `libMesh`, `deal.II`), a shape function is the Lagrange interpolant at the nodes of a single element $\mathcal{T}^*$. It has the property $\psi_i(\mathbf{x}_j) = \delta_{ij}$ whenever $\mathbf{x}_j$ is a node of $\mathcal{T}^*$. The finite element basis functions are assembled from piecewise definitions involving these shape functions.

While representing vector solutions with scalar basis functions by decomposing into components as in (4) is advantageous in software, it is inconvenient for this discussion. We thus consider a reindexing scheme by creating vector-valued shape functions

$$\boldsymbol{\varphi}_k(\mathbf{x}) = \begin{cases} \psi_k(\mathbf{x})\,\mathbf{e}_1 & 1 \le k \le N \\ \psi_{k-N}(\mathbf{x})\,\mathbf{e}_2 & N+1 \le k \le 2N \end{cases}$$

so that $\mathbf{u}$ is more concisely written as

$$\mathbf{u}\big|_{\mathcal{T}} = \sum_{k=1}^{2N} \alpha_k \boldsymbol{\varphi}_k(\mathbf{x}) \tag{5}$$

It is this separation into Cartesian components that prohibits sliding along any axis other than a coordinate one. The boundary condition is a Dirichlet one of movement specified along $\mathbf{d}$. Suppose we enforce this condition by setting the appropriate degrees of freedom in the above representation. In $\mathbb{R}^2$, if $\mathbf{d} \neq \alpha\mathbf{e}_i$ for

some α and i, then for each k such that $\mathbf{x}_k \in \partial\Omega$, we must set the boundary conditions

$$\alpha_k = \frac{\dot{s}(\mathbf{x}_k)\Delta t}{d^{(1)}} \qquad\qquad \alpha_{k-N} = \frac{\dot{s}(\mathbf{x}_{k-N})\Delta t}{d^{(2)}}$$

whereby $d^{(i)}$ represents the i^{th} component[a] of $\mathbf{d}$, all of which are necessarily nonzero by assumption. On a sliding set, $\dot{s} \equiv 0$. All available degrees of freedom go to representing $\mathbf{d}$ in the standard basis so that the above quashes *all* motion when $\mathbf{d}$ is not aligned with a coordinate axis.

The solution, therefore, is to change the basis from the standard $\{\mathbf{e}_1, \mathbf{e}_2\}$ to one more appropriate for local conditions. Define, instead, the following vector basis functions:

$$\tilde{\boldsymbol{\varphi}}_k(\mathbf{x}) = \begin{cases} \psi_k(\mathbf{x})\,\mathbf{p}_k & 1 \le k \le m \\ \psi_k(\mathbf{x})\,\mathbf{e}_1 & m+1 \le k \le N \\ \psi_{k-N}(\mathbf{x})\,\mathbf{q}_{k-N} & N+1 \le k \le m+N \\ \psi_{k-N}(\mathbf{x})\,\mathbf{e}_2 & m+N+1 \le k \le 2N \end{cases}$$

For each k, we require $\mathbb{R}^2 = \text{span}\,\{\mathbf{p}_k, \mathbf{q}_k\}$ so that all solution vectors are representable in $\{\tilde{\boldsymbol{\varphi}}_k\}_{k=1}^N$. For the moment, suppose that each $\mathbf{p}_k$ and $\mathbf{q}_k$ is known *a priori*; defining these directions is a matter of circumstance and the focus of a future section. The new shape functions can be written in terms of the old:

$$\tilde{\boldsymbol{\varphi}}_k(\mathbf{x}) = \begin{cases} p_k^{(1)}\boldsymbol{\varphi}_k(\mathbf{x}) + p_k^{(2)}\boldsymbol{\varphi}_{k+N}(\mathbf{x}) & 1 \le k \le m \\ \boldsymbol{\varphi}_k(\mathbf{x}) & m+1 \le k \le N \\ q_{k-N}^{(1)}\boldsymbol{\varphi}_{k-N}(\mathbf{x}) + q_{k-N}^{(2)}\boldsymbol{\varphi}_k(\mathbf{x}) & N+1 \le k \le m+N \\ \boldsymbol{\varphi}_k(\mathbf{x}) & m+N+1 \le k \le 2N \end{cases} \tag{6}$$

We may encode the transformation as matrix multiplication

$$\tilde{\boldsymbol{\varphi}}_k(\mathbf{x}) = \sum_{l=1}^{2N} \mathcal{M}_{kl}\boldsymbol{\varphi}_l(\mathbf{x}) \tag{7}$$

where $\mathcal{M}_{kl}$ are the entries of the matrix

$$\mathcal{M} = \left[\begin{array}{ccc:ccc|ccc:ccc} p_1^{(1)} & & 0 & 0 & \cdots & 0 & p_1^{(2)} & & 0 & 0 & \cdots & 0 \\ & \ddots & & \vdots & \ddots & \vdots & & \ddots & & \vdots & \ddots & \vdots \\ 0 & & p_m^{(1)} & 0 & \cdots & 0 & 0 & & p_m^{(2)} & 0 & \cdots & 0 \\ 0 & \cdots & 0 & 1 & & 0 & 0 & \cdots & 0 & 0 & \cdots & 0 \\ \vdots & \ddots & \vdots & & \ddots & & \vdots & \ddots & \vdots & \vdots & \ddots & \vdots \\ 0 & \cdots & 0 & 0 & & 1 & 0 & \cdots & 0 & 0 & \cdots & 0 \\ \hline q_1^{(1)} & & 0 & 0 & \cdots & 0 & q_1^{(2)} & & 0 & 0 & \cdots & 0 \\ & \ddots & & \vdots & \ddots & \vdots & & \ddots & & \vdots & \ddots & \vdots \\ 0 & & q_m^{(1)} & 0 & \cdots & 0 & 0 & & q_m^{(2)} & 0 & \cdots & 0 \\ 0 & \cdots & 0 & 0 & \cdots & 0 & 0 & \cdots & 0 & 1 & & 0 \\ \vdots & \ddots & \vdots & \vdots & \ddots & \vdots & \vdots & \ddots & \vdots & & \ddots & \\ 0 & \cdots & 0 & 0 & \cdots & 0 & 0 & \cdots & 0 & 0 & & 1 \end{array}\right] \begin{array}{l} \left.\vphantom{\begin{array}{c}a\\a\\a\\a\\a\\a\end{array}}\right\} N-m \\ \left.\vphantom{\begin{array}{c}a\\a\\a\\a\\a\\a\end{array}}\right\} N \\ \left.\vphantom{\begin{array}{c}a\\a\\a\\a\\a\\a\end{array}}\right\} N-m \\ \left.\vphantom{\begin{array}{c}a\\a\\a\\a\\a\\a\end{array}}\right\} N \end{array}$$

$$\underbrace{\qquad\qquad}_{N} \quad \underbrace{\qquad\qquad}_{N-m} \qquad \underbrace{\qquad\qquad}_{N} \quad \underbrace{\qquad\qquad}_{N-m}$$

The element stiffness matrix $\mathbf{K}$ is the Gram matrix for the associated bilinear form $a(\,\cdot\,,\,\cdot\,)$ when the basis functions are restricted only to a single element. It has entries

$$K_{ij} = a(\boldsymbol{\varphi}_i, \boldsymbol{\varphi}_j)$$

[a] We follow standard convention: boldface for vectors quantities, normal weight for scalar ones.

Let $\tilde{\mathbf{K}}$ be the element stiffness matrix for the new basis functions $\tilde{\boldsymbol{\varphi}}_k$. Then using (7), we have

$$\begin{aligned}
\tilde{K}_{ij} &= a(\tilde{\boldsymbol{\varphi}}_i, \tilde{\boldsymbol{\varphi}}_j) \\
&= a\left(\sum_{l=1}^{2N} \mathcal{M}_{il}\boldsymbol{\varphi}_l(\mathbf{x}), \sum_{l'=1}^{2N} \mathcal{M}_{jl'}\boldsymbol{\varphi}_{l'}(\mathbf{x})\right) \\
&= \sum_{l=1}^{2N}\sum_{l'=1}^{2N} \mathcal{M}_{jl'}a(\boldsymbol{\varphi}_l(\mathbf{x}), \boldsymbol{\varphi}_{l'}(\mathbf{x}))\mathcal{M}_{il} \\
&= \sum_{l=1}^{2N}\left(\sum_{l'=1}^{2N} \mathcal{M}_{jl'}K_{ll'}\right)\mathcal{M}_{il}
\end{aligned}$$

which we recognize, though matrix multiplication, as the congruence relation

$$\tilde{\mathbf{K}} = \mathcal{M}^T\mathbf{K}\mathcal{M} \tag{8}$$

The element contribution to righthand-side vector, which we denote as the vector $\mathbf{E}$ which has entries

$$E_k = \left\langle \mathbf{f}, \boldsymbol{\varphi}_k \right\rangle_{L^2(\mathcal{T})}$$

can be transformed in a similar manner. Let $\tilde{\mathbf{E}}$ have entries

$$\tilde{E}_k = \left\langle \mathbf{f}, \tilde{\boldsymbol{\varphi}}_k \right\rangle_{L^2(\mathcal{T})}$$

Then using (7), we have

$$\tilde{E}_k = \sum_{l=1}^{2N} \mathcal{M}_{kl}\left\langle \mathbf{f}, \boldsymbol{\varphi}_l \right\rangle_{L^2(\mathcal{T})}$$

or in terms of matrix-vector multiplication

$$\tilde{\mathbf{E}} = \mathcal{M}^T\mathbf{E} \tag{9}$$

A bit of forward thinking reveals that that one-dimensional subspaces are sufficient to describe the motion of a node. Once we've determined the new location of a receding node, we can compute a single vector of motion by subtracting the new position from the old. For sliding sets, motion is forbidden in the normal direction but allowed in the orthogonal complement of the normal—this is again a one-dimensional subspace. Consequently, nonzero motion will occur in the direction of at most one of $\mathbf{p}_k$ and $\mathbf{q}_k$. We can therefore simplify matters considerably by choosing that $\mathbf{q}$ places $\mathcal{M}$ into a favorable form. By taking

$$\mathbf{q}_k = \left[p_k^{(2)}, -p_k^{(1)}\right]^T \tag{10}$$

$\mathcal{M}$ becomes symmetric:

$$\mathcal{M} = \left[\begin{array}{ccc|ccc|ccc:ccc}
p_1^{(1)} & & 0 & 0 & \cdots & 0 & p_1^{(2)} & & 0 & 0 & \cdots & 0 \\
 & \ddots & & \vdots & \ddots & \vdots & & \ddots & & \vdots & \ddots & \vdots \\
0 & & p_m^{(1)} & 0 & \cdots & 0 & 0 & & p_m^{(2)} & 0 & \cdots & 0 \\
\hline
0 & \cdots & 0 & 1 & & 0 & 0 & \cdots & 0 & 0 & \cdots & 0 \\
\vdots & \ddots & \vdots & & \ddots & & \vdots & \ddots & \vdots & \vdots & \ddots & \vdots \\
0 & \cdots & 0 & 0 & & 1 & 0 & \cdots & 0 & 0 & \cdots & 0 \\
\hline
p_1^{(2)} & & 0 & 0 & \cdots & 0 & -p_1^{(1)} & & 0 & 0 & \cdots & 0 \\
 & \ddots & & \vdots & \ddots & \vdots & & \ddots & & \vdots & \ddots & \vdots \\
0 & & p_m^{(2)} & 0 & \cdots & 0 & 0 & & -p_m^{(1)} & 0 & \cdots & 0 \\
\hdashline
0 & \cdots & 0 & 0 & \cdots & 0 & 0 & \cdots & 0 & 1 & & 0 \\
\vdots & \ddots & \vdots & \vdots & \ddots & \vdots & \vdots & \ddots & \vdots & & \ddots & \\
0 & \cdots & 0 & 0 & \cdots & 0 & 0 & \cdots & 0 & 0 & & 1
\end{array}\right] \tag{11}$$

If $|\mathbf{p}_k| = 1$, then $\mathcal{M}$ becomes an orthogonal matrix, making inversion trivial.

B. Applying the Corrections

As long as $\mathcal{M}$ is invertible, we can obtain the stiffness for the old basis vectors from that of the new ones. This is especially convenient in software because the component-decomposition of $\boldsymbol{\varphi}_k$ greatly simplifies storage and bookeeping requirements. For a true representation involving the $\tilde{\boldsymbol{\varphi}}_k$, we must store thousands of vector components and track what maps where. It is therefore more advantageous to use the $\tilde{\boldsymbol{\varphi}}$ as an intermediate local step and ultimately return to a representation in $\boldsymbol{\varphi}_k$.

As long as $\mathcal{M}$ is invertible, we can do exactly what we propose above by reversing (8) and (9):

$$\mathbf{K} = \mathcal{M}^{-T}\tilde{\mathbf{K}}\mathcal{M}^{-1} = \mathcal{M}\tilde{\mathbf{K}}\mathcal{M}$$
$$\mathbf{E} = \mathcal{M}^{-T}\tilde{\mathbf{E}} = \mathcal{M}\tilde{\mathbf{E}}$$

with the second equalities following from the orthorgonality of $\mathcal{M}$.

The symmetric $\mathcal{M}$ was constructed under the assertation that only a single direction would describe the motion, but this does not mean that we will always *enforce* a single direction of motion. When only one direction is enforced, the dynamics of the linear elastic system determine the motion in the reciprocal basis direction. This is precisely what we desire for sliding side sets but not receding ones, on which we wish to specify exactly the magnitude and direction of motion.

Analagous to (5), we can write the finite element solution in terms of the new basis as

$$\mathbf{u}\big|_{\mathcal{T}} = \sum_{k=1}^{2N} \tilde{\alpha}_k \tilde{\boldsymbol{\varphi}}_k(\mathbf{x})$$

Choose $\mathbf{p}_k = \mathbf{d}(\mathbf{x}_k)$, the direction of the boundary condition at $\mathbf{x}_k$. Then we can set the boundary condition exactly by equating this condition with the corresponding degree of freedom:

$$\tilde{\alpha}_k = \hat{s}(\mathbf{x}_k)$$

where $\hat{s}$ is the (perhaps nonphysical) prescribed value of recesssion, and when appropriate (i.e., motion in the direction of $\mathbf{q}_k$ should be quashed)

$$\tilde{\alpha}_{k-N} = 0$$

Ordinarily, when degrees of freedom are set in this manner, the corresponding variables are removed from the system and references to those quantities replaced by the predetermined value. Unfortunately, this is not possible here because the above sets degrees of freedom in $\tilde{\boldsymbol{\varphi}}_k$, which are local shape functions, not finite element basis functions. Accordingly, removal of system variables would require a complex and cumbersome bookeeping step that would negate much of the perceived value of reducing the system size.

Instead, we exploit the nature of finite precision arithmetic: if we have

$$\varepsilon^{-1}a + b = \varepsilon^{-1}c$$

then $a \to c$ as $\varepsilon \to 0+$ if $\min\left\{\left|\frac{a}{b}\right|, \left|\frac{b}{a}\right|\right\} \ll \varepsilon^{-1}$. On a computer, by taking $\varepsilon^{-1} \sim 10^{15}$, for instance, it is possible to obtain $a \approx c$ to nearly machine precision. This technique is known as a penalty method.

To enforce the condition via the penalty method, we modify $\tilde{\mathbf{K}}$ and $\tilde{\mathbf{E}}$ by adding perturbations:

$$\tilde{\mathbf{K}} \mapsto \tilde{\mathbf{K}} + \mathbf{P}_i$$
$$\tilde{\mathbf{E}} \mapsto \tilde{\mathbf{E}} + \mathbf{B}_i$$

The subscripts on $\mathbf{P}$ and $\mathbf{B}$ match the number of degrees of freedom being set (single or double enforcement). With these modifications, the contributions from the element $\mathcal{T}$ to the global stiffness matrix $\mathbf{A}$ and right-hand vector $\mathbf{F}$ become, per (8),

$$\begin{aligned} \mathbf{A}\big|_{\mathcal{T}} = \mathbf{K} \hookleftarrow \mathcal{M}\big(\tilde{\mathbf{K}} + \mathbf{P}_i\big)\mathcal{M} \quad &\Longrightarrow \quad \mathbf{A}\big|_{\mathcal{T}} = \mathbf{K} + \hat{\mathbf{K}}_i \qquad \hat{\mathbf{K}}_i \triangleq \mathcal{M}\mathbf{P}_i\mathcal{M} \\ \mathbf{F}\big|_{\mathcal{T}} = \mathbf{E} \hookleftarrow \mathcal{M}\big(\tilde{\mathbf{E}} + \mathbf{B}_i\big) \quad &\Longrightarrow \quad \mathbf{F}\big|_{\mathcal{T}} = \mathbf{E} + \hat{\mathbf{E}}_i \qquad \hat{\mathbf{E}}_i \triangleq \mathcal{M}\mathbf{B} \end{aligned} \tag{12}$$

1. Single Enforcement

When only one degree of freedom is set, the perturbations take the form

$$\mathbf{P}_1 \triangleq \varepsilon^{-1} \left[\begin{array}{c|c} \mathbf{I} & \mathbf{0} \\ \hline \mathbf{0} & \mathbf{0} \end{array}\right] \begin{matrix} \}m \\ \}2N-m \end{matrix} \qquad \mathbf{B}_1 \triangleq \varepsilon^{-1} \left[\begin{array}{c} \hat{s}(\mathbf{x}_1) \\ \vdots \\ \hat{s}(\mathbf{x}_m) \\ \hline 0 \\ \vdots \\ 0 \end{array}\right] \tag{13}$$

in which $\mathbf{I}$ denotes the identity matrix. If we partition $\mathcal{M}$ as

$$\mathcal{M} = \begin{bmatrix} \mathbf{M}_{11} & \mathbf{M}_{12} \\ \mathbf{M}_{21} & \mathbf{M}_{22} \end{bmatrix}$$

with the blocks defined along the thick dividing lines of (11), we can exploit the block structure of $\mathbf{P}$ and perform the required multiplication easily:

$$\mathcal{M}\mathbf{P}_1\mathcal{M} = \begin{bmatrix} \mathbf{M}_{11}^2 & \mathbf{M}_{11}\mathbf{M}_{12} \\ \mathbf{M}_{21}\mathbf{M}_{11} & \mathbf{M}_{21}\mathbf{M}_{12} \end{bmatrix} \tag{14}$$

We thus obtain a nicely simplifed result for the corrections to the stiffness matrix and righthand-side vector:

$$\hat{\mathbf{K}}_1 = \frac{1}{\varepsilon} \left[\begin{array}{ccccc|ccccc} \left[p_1^{(1)}\right]^2 & & & & 0 & p_1^{(1)}p_1^{(2)} & & & & 0 \\ & \ddots & & & & & \ddots & & & \\ & & \left[p_m^{(1)}\right]^2 & & & & & p_m^{(1)}p_m^{(2)} & & \\ & & & 0 & & & & & 0 & \\ & & & & \ddots & & & & & \ddots \\ 0 & & & & 0 & 0 & & & & 0 \\ \hline p_1^{(1)}p_1^{(2)} & & & & 0 & \left[p_1^{(2)}\right]^2 & & & & 0 \\ & \ddots & & & & & \ddots & & & \\ & & p_m^{(1)}p_m^{(2)} & & & & & \left[p_m^{(2)}\right]^2 & & \\ & & & 0 & & & & & 0 & \\ & & & & \ddots & & & & & \ddots \\ 0 & & & & 0 & 0 & & & & 0 \end{array}\right] \qquad \hat{\mathbf{E}}_1 = \frac{1}{\varepsilon} \begin{bmatrix} \hat{s}(\mathbf{x}_1)d_1^{(1)} \\ \vdots \\ \hat{s}(\mathbf{x}_m)d_m^{(1)} \\ 0 \\ \vdots \\ 0 \\ \hat{s}(\mathbf{x}_1)d_1^{(2)} \\ \vdots \\ \hat{s}(\mathbf{x}_m)d_m^{(2)} \\ 0 \\ \vdots \\ 0 \end{bmatrix} \begin{matrix} \\ \\ \\ \\ \\ \}N-m \\ \\ \\ \\ \\ \}N-m \end{matrix}$$

2. Double Enforcement

When both degrees of freedom are set, we have

$$\mathbf{P}_2 \triangleq \varepsilon^{-1} \begin{bmatrix} \mathbf{I} & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{I} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} \end{bmatrix} \qquad \mathbf{B}_2 \triangleq \varepsilon^{-1} \begin{bmatrix} \hat{\mathbf{s}}(\mathbf{p}_1, \ldots, \mathbf{p}_m) \\ \mathbf{0} \\ \hat{\mathbf{s}}(\mathbf{p}_1^{\perp}, \ldots, \mathbf{p}_m^{\perp}) \\ \mathbf{0} \end{bmatrix} \tag{15}$$

The block sizes in $\mathbf{P}_2$ mirror those of $\mathcal{M}$ when partitioned along the dotted and thin black lines in (11):

$$\mathcal{M} = \begin{bmatrix} \mathbf{M}_{11} & \mathbf{0} & \mathbf{M}_{13} & \mathbf{0} \\ \mathbf{0} & \mathbf{I} & \mathbf{0} & \mathbf{0} \\ \mathbf{M}_{31} & \mathbf{0} & \mathbf{M}_{33} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{I} \end{bmatrix}$$

The vector $\mathbf{p}_k^{\perp}$ is $\mathbf{q}_k$ as defined by (10). Within $\mathbf{B}_2$, the block $\hat{\mathbf{s}}(\mathbf{d}_1, \ldots, \mathbf{d}_m) \in \mathbb{R}^m$ is defined as

$$\hat{\mathbf{s}}(\mathbf{d}_1, \ldots, \mathbf{d}_m) \triangleq \begin{bmatrix} \hat{s}(\mathbf{x}_1; \mathbf{d}_1) \\ \vdots \\ \hat{s}(\mathbf{x}_m; \mathbf{d}_m) \end{bmatrix} \tag{16}$$

The inclusion of a direction $\mathbf{d}_i$ indicates that the recession value $\hat{s}(\mathbf{x}_i)$ is in the direction $\mathbf{d}_i$. Performing the multiplication to form $\hat{\mathbf{K}}_2$, we have

$$\hat{\mathbf{K}}_2 = \begin{bmatrix} \mathbf{M}_{11}^2 + \mathbf{M}_{13}\mathbf{M}_{31} & \mathbf{0} & \mathbf{M}_{11}\mathbf{M}_{13} + \mathbf{M}_{13}\mathbf{M}_{33} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \mathbf{M}_{31}\mathbf{M}_{11} + \mathbf{M}_{33}\mathbf{M}_{31} & \mathbf{0} & \mathbf{M}_{31}\mathbf{M}_{13} + \mathbf{M}_{33}^2 & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} \end{bmatrix}$$

Each block $\mathbf{M}_{ij}$ is a diagonal matrix. Letting $\left[\mathbf{A}\right]_{ij}$ denote the (i,j) entry of the matrix $\mathbf{A}$, we can compute the entries in the above representation:

$$\begin{aligned} \left[\mathbf{M}_{11}^2 + \mathbf{M}_{13}\mathbf{M}_{31}\right]_{ii} &= \left[p_i^{(1)}\right]^2 + \left[p_i^{(2)}\right]^2 = 1 \\ \left[\mathbf{M}_{11}\mathbf{M}_{13} + \mathbf{M}_{13}\mathbf{M}_{33}\right]_{ii} &= p_i^{(1)} p_i^{(2)} - p_i^{(2)} p_i^{(1)} = 0 \\ \left[\mathbf{M}_{31}\mathbf{M}_{11} + \mathbf{M}_{33}\mathbf{M}_{31}\right]_{ii} &= p_i^{(2)} p_i^{(1)} - p_i^{(1)} p_i^{(2)} = 0 \\ \left[\mathbf{M}_{31}\mathbf{M}_{13} + \mathbf{M}_{33}^2\right]_{ii} &= \left[p_i^{(1)}\right]^2 + \left[p_i^{(2)}\right]^2 = 1 \end{aligned}$$

Then, we obtain an amazingly simple result for $\hat{\mathbf{K}}_2$ and slightly complicated one for $\hat{\mathbf{E}}_2$:

$$\hat{\mathbf{K}}_2 = \frac{1}{\varepsilon} \left[\begin{array}{ccc|ccc} 1 & & & 0 & & \\ & \ddots & & & \ddots & \\ & & 1 & & & 0 \\ \hline 0 & & & 1 & & \\ & \ddots & & & \ddots & \\ & & 0 & & & 1 \end{array}\right] \begin{matrix} \left.\begin{matrix} \\ \\ \\ \end{matrix}\right\} m \\ \left.\begin{matrix} \\ \\ \\ \end{matrix}\right\} N-m \\ \left.\begin{matrix} \\ \\ \\ \end{matrix}\right\} m \\ \left.\begin{matrix} \\ \\ \\ \end{matrix}\right\} N-m \end{matrix}$$

$$\begin{matrix} \underbrace{}_{m} & \underbrace{}_{m} \\ \underbrace{}_{N-m} & \underbrace{}_{N-m} \end{matrix}$$

$$\hat{\mathbf{E}}_2 = \frac{1}{\varepsilon} \begin{bmatrix} \left\langle \mathbf{p}_1, \hat{\mathbf{s}}(\mathbf{x}_1; \mathbf{p}_1, \mathbf{p}_1^{\perp}) \right\rangle \\ \vdots \\ \left\langle \mathbf{p}_m, \hat{\mathbf{s}}(\mathbf{x}_m; \mathbf{p}_m, \mathbf{p}_m^{\perp}) \right\rangle \\ 0 \\ \vdots \\ 0 \\ \left\langle \mathbf{p}_1^{\perp}, \hat{\mathbf{s}}(\mathbf{x}_1; \mathbf{p}_1, \mathbf{p}_1^{\perp}) \right\rangle \\ \vdots \\ \left\langle \mathbf{p}_m^{\perp}, \hat{\mathbf{s}}(\mathbf{x}_m; \mathbf{p}_m, \mathbf{p}_m^{\perp}) \right\rangle \\ 0 \\ \vdots \\ 0 \end{bmatrix} \begin{matrix} \left.\begin{matrix} \\ \\ \\ \\ \\ \\ \\ \\ \\ \end{matrix}\right\} N-m \\ \left.\begin{matrix} \\ \\ \\ \\ \\ \\ \\ \\ \\ \end{matrix}\right\} N-m \end{matrix}$$

We reuse the previous notation (16) for compactness; for clarity, the entries of $\hat{\mathbf{E}}_2$ are of the form

$$\begin{aligned} \left\langle \mathbf{p}, \hat{\mathbf{s}}(\mathbf{x}^*; \mathbf{p}, \mathbf{p}^{\perp}) \right\rangle &= \hat{s}(\mathbf{x}^*; \mathbf{p}) p^{(1)} + \hat{s}(\mathbf{x}^*; \mathbf{p}^{\perp}) p^{(2)} \\ \left\langle \mathbf{p}^{\perp}, \hat{\mathbf{s}}(\mathbf{x}^*; \mathbf{p}, \mathbf{p}^{\perp}) \right\rangle &= \hat{s}(\mathbf{x}^*; \mathbf{p}) p^{(2)} - \hat{s}(\mathbf{x}^*; \mathbf{p}^{\perp}) p^{(1)} \end{aligned}$$

C. Setting the Direction and Magnitude of Motion

The previous section supposes that the recession values and directions of motion are known *a priori*. This section describes how to determine them. First, we need to define a simple but critical concept:

Definition (Side set).

A *side set* is a subset of $\partial\Omega$. Each of $\partial\Omega_R$, $\partial\Omega_S$, and $\partial\Omega_F$ is the union of side sets.

In implementation, the user will specify the side sets so that $\partial\Omega_R$, $\partial\Omega_S$, and $\partial\Omega_F$ are fully determined. For any two side sets $\mathcal{S}$ and $\mathcal{S}'$, it is *not* the case that $\mathcal{S} \cap \mathcal{S}' = \emptyset$; instead, we should have the property that $|\mathcal{S} \cap \mathcal{S}'|_{\mathbb{R}^{d-1}} = 0$ so that non-empty intersections can be no more than singletons in 2D and lines in 3D.

How the directions of motion are constructed depend on the problem geometry and the location of the node. Side sets allow for quick and easy classification that eliminates the need for complex geometric detection algorithms. With their help, we reduce the analysis to two node types in two-dimensions:

Definition (2D Node Classification).

In two-dimensional meshes, a node is a *corner node* if it is contained in two side sets. It is an *interior node* if it is contained in only one side set.

It is important to note the distinction between our definition of a corner and the geometric notion of a corner. Automatic detection based on the latter is precarious, as it is easy for an algorithm to misclassify coarse portions of the mesh, which can occur even in well-constructed meshes after extensive recession.

1. Corner Nodes

In two dimensions, a node is contained in exactly two boundary faces (edge). It is possible for a node to be contained in an arbitrary number of *elements* but our requirement that intersection with the boundary be non-trivial excludes from consideration all but one or two. It is beneficial to realize that the elements themselves are not important—it is the edges that give us the important information. We therefore simplify our analysis by disregarding the elements once we've identified the boundary edges.

The physical recession values $\dot{s}$ are known only at quadrature points along the edge, yet the results of the previous section require a value at the node itself. A potential solution is apparent:

1. Choose the value of $\dot{s}$ at the closest quadrature point for each edge.
2. Interpolate between these two values to construct an estimate for $\dot{s}$ at the node.

While seemingly sound, this method has a major deficiency: with only two values, we must use linear interpolation in step 2. Besides physical incorrectness, this effort is unable to consider curvature or other properties of the mesh and is likely to lead to poor mesh quality and skewed geometry. Instead, the solution is, in essence, to move the entire edge at once:

1. For each face, compute the "new quadrature points" by moving $\dot{s}$ along $\boldsymbol{\nu}$, both of which are known exactly, and then compute the best fit line through these new points to determine the new edge.
2. Compute the intersection of the new edges and make that the new location of the node.
3. The motion $\hat{s}$ and direction vector $\mathbf{p}$ are obtained by subtracting the old location from the new.

This idea is realized in Algorithm 1 below. Further explanation of the steps follows the presentation.

Algorithm 1 (Corner Motion).

1. Let $\mathcal{E}$ be the edge containing the node $\mathbf{x}^*$ from one side set and $\mathcal{E}'$ be the other.
2. Let $\mathbf{x}_1, \ldots, \mathbf{x}_q$ be the quadrature points of $\mathcal{E}$. Let $\boldsymbol{\nu}_k$ and $\dot{s}_k$ be the normal and recession values, respectively, specified at the quadrature point $\mathbf{x}_k$. Let $\mathbf{x}'_1, \ldots, \mathbf{x}'_{q'}$, $\boldsymbol{\nu}'_k$, and $\dot{s}'_k$ be the similarly named and corresponding quantities from $\mathcal{E}'$.

(continued).

3. Compute the centroids of the sets $\{\mathbf{x}_k + (\dot{s}_k\Delta t)\boldsymbol{\nu}_k\}_{k=1}^{q}$ and $\{\mathbf{x}'_k + (\dot{s}'_k\Delta t)\boldsymbol{\nu}'_k\}_{k=1}^{q'}$:

$$\mathbf{c} \triangleq \frac{1}{q}\sum_{k=1}^{q}\left[\mathbf{x}_k + (\dot{s}_k\Delta t)\boldsymbol{\nu}_k\right] \qquad \mathbf{c}' \triangleq \frac{1}{q'}\sum_{k=1}^{q'}\left[\mathbf{x}'_k + (\dot{s}'_k\Delta t)\boldsymbol{\nu}'_k\right]$$

4. Form the $2 \times q$ matrix $\mathbf{A}$ and $2 \times q'$ matrix $\mathbf{A}'$

$$\mathbf{A} \triangleq \left[\begin{array}{c|c|c} \mathbf{x}_1 + (\dot{s}_1\Delta t)\boldsymbol{\nu}_1 - \mathbf{c} & \cdots & \mathbf{x}_q + (\dot{s}_q\,\Delta t)\boldsymbol{\nu}_q - \mathbf{c} \end{array}\right]$$
$$\mathbf{A}' \triangleq \left[\begin{array}{c|c|c} \mathbf{x}'_1 + (\dot{s}'_1\Delta t)\boldsymbol{\nu}'_1 - \mathbf{c}' & \cdots & \mathbf{x}'_q + (\dot{s}'_{q'}\Delta t)\boldsymbol{\nu}'_q - \mathbf{c}' \end{array}\right]$$

5. Compute the singular value decompositions of $\mathbf{A}$ and $\mathbf{A}'$:

$$\mathbf{A} = \mathbf{U}\boldsymbol{\Sigma}\mathbf{V} \qquad \mathbf{A}' = \mathbf{U}'\boldsymbol{\Sigma}'\mathbf{V}'$$

   where $\mathbf{U} \in \mathbb{R}^{2\times 2}$, $\boldsymbol{\Sigma} \in \mathbb{R}^{2\times q}$, $\mathbf{V} \in \mathbb{R}^{q\times q}$, $\mathbf{U}' \in \mathbb{R}^{2\times 2}$, $\boldsymbol{\Sigma}' \in \mathbb{R}^{2\times q'}$, and $\mathbf{V}' \in \mathbb{R}^{q'\times q'}$. Let $\mathbf{n}_1 \triangleq \mathbf{u}_3$ and $\mathbf{n}_2 \triangleq \mathbf{u}'_3$, the third columns of $\mathbf{U}$ and $\mathbf{U}'$, respectively.

6. Solve the linear system

$$\begin{bmatrix} -n_1^{(2)} & n_2^{(2)} \\ n_1^{(1)} & -n_2^{(1)} \end{bmatrix}\begin{bmatrix} s^* \\ t^* \end{bmatrix} = \mathbf{c}' - \mathbf{c}$$

   and set $\mathbf{x}^*_{\text{new}} = \mathbf{c} + s^*\left[-n_1^{(2)}, n_1^{(1)}\right]$.

7. Set $\hat{s} = |\mathbf{x}^*_{\text{new}} - \mathbf{x}^*|$ and $\mathbf{p} = \hat{s}^{-1}\left(\mathbf{x}^*_{\text{new}} - \mathbf{x}^*\right)$.

8. Follow Section 2 with the above $\mathbf{p}$ and $\hat{s}$. Take $\hat{s}(\mathbf{x}^*; \mathbf{p}^\perp) = 0$.

The points $\tilde{\mathbf{x}}_k \triangleq \mathbf{x}_k + (\dot{s}_k\Delta t)\boldsymbol{\nu}_k$ are the "new quadrature points," merely the result of applying the motion at each quadrature point. They will not necessarily be the quadature points of the new edge. In fact, the $\tilde{\mathbf{x}}_k$ need not be colinear if $q > 2$. As a result, the new edge is determined by best-fit, the solution to

$$\min_{\hat{\mathbf{n}},\mathbf{y}\in\mathbb{R}^2}\sum_{k=1}^{q}\langle\tilde{\mathbf{x}}_k - \mathbf{y}, \hat{\mathbf{n}}\rangle^2$$

We can simplify our problem considerably with the following lemma:

Lemma 1.

The least-squares best-fit plane of $\{\mathbf{x}_k\}_{k=1}^{N} \subset \mathbb{R}^d$ contains the centroid $\mathbf{c} = \frac{1}{N}\sum_{k=1}^{N}\mathbf{x}_k$.

For a proof of this result, see the Appendix. We seek the "best one-dimensional subspace" spanned by the columns of $\mathbf{A}$. To find this subspace, we call upon the singular value decomposition

$$\mathbf{A} = \mathbf{U}\boldsymbol{\Sigma}\mathbf{V}^T$$

where $\mathbf{U} \in \mathbb{R}^{2\times 2}$ and unitary, $\mathbf{V} \in \mathbb{R}^{q\times q}$ and unitary, and $\boldsymbol{\Sigma}_k \in \mathbb{R}^{2\times q}$ of the form

$$\boldsymbol{\Sigma} = \begin{bmatrix} \boldsymbol{\Sigma}_r & \mathbf{0}_{r\times(q-r)} \\ \mathbf{0}_{(2-r)\times r} & \mathbf{0}_{(2-r)\times(q-r)} \end{bmatrix} \qquad \boldsymbol{\Sigma}_r = \begin{bmatrix} \sigma_1 & & \\ & \ddots & \\ & & \sigma_r \end{bmatrix}$$

in which $r = \operatorname{rank} \mathbf{A}$ and $\sigma_1 \geq \cdots \geq \sigma_r > 0$. This second lemma further aids us in our quest:

Lemma 2.

The first s columns of $\mathbf{U}$ represent the "biggest" s-dimensional subspace, the one that has the smallest representation error in Euclidean norm, contained in range($\mathbf{A}$).

The proof of Lemma 2 is again contained in the Appendix. These two lemma explain the mysterious subtraction of $\mathbf{c}$ from each column of $\mathbf{A}$ and $\mathbf{c}'$ from $\mathbf{A}'$. First, we note that range($\mathbf{A}$) = range$\begin{bmatrix}\mathbf{u}_1 & \cdots & \mathbf{u}_r\end{bmatrix}$:

$$\begin{aligned}\mathbf{A}\mathbf{x} &= \mathbf{U}\begin{bmatrix}\sigma_1\mathbf{V}^T\mathbf{x} & \cdots & \sigma_r\mathbf{V}^T\mathbf{x} & \mathbf{0} & \cdots & \mathbf{0}\end{bmatrix} \\ &= \sum_{i=1}^{r} \sigma_i \langle \mathbf{x}, \mathbf{v}_i \rangle \mathbf{u}_i\end{aligned}$$

Since the centroid is contained in the best-fit plane, by subtracting it from each data point creates a candidate for the best-fit line we seek. The task therefore becomes to find a single vector that best encapsulate the columnspaces of $\mathbf{A}$ (and similarly for $\mathbf{A}'$), a problem to which we can directly apply Lemma 2.

The above machinery explains how steps 1-5 of Algorithm 1 compute the new edges; we have $\mathcal{E} \mapsto (\mathbf{c}, \mathbf{n}_1)$ and $\mathcal{E}' \mapsto (\mathbf{c}', \mathbf{n}_2)$, where the pairing indicates the normal vector and point inside the new edge "plane." From the contents of these pairings we can write parametric expressions for the new edges:

$$\begin{aligned}\mathcal{E} \mapsto \ell(s) &= \mathbf{c} + s\begin{bmatrix}-n_1^{(2)}, n_1^{(1)}\end{bmatrix} \\ \mathcal{E}' \mapsto \ell'(t) &= \mathbf{c}' + t\begin{bmatrix}-n_2^{(2)}, n_2^{(1)}\end{bmatrix}\end{aligned}$$

Step 6, therefore, computes the intersection of ℓ and ℓ' in terms of the above parameterization and sets the location of $\mathbf{x}^*_{\text{new}}$ to be the point $\ell(s^*) = \ell'(t^*)$. As promised, the direction of motion $\mathbf{p}$ is the vector that takes us from the original location to $\mathbf{x}^*_{\text{new}}$; normalizing this vector and saving the original length gives us $\hat{s}$. Since this completely describes the desired displacement, we quash movement in the orthogonal complement.

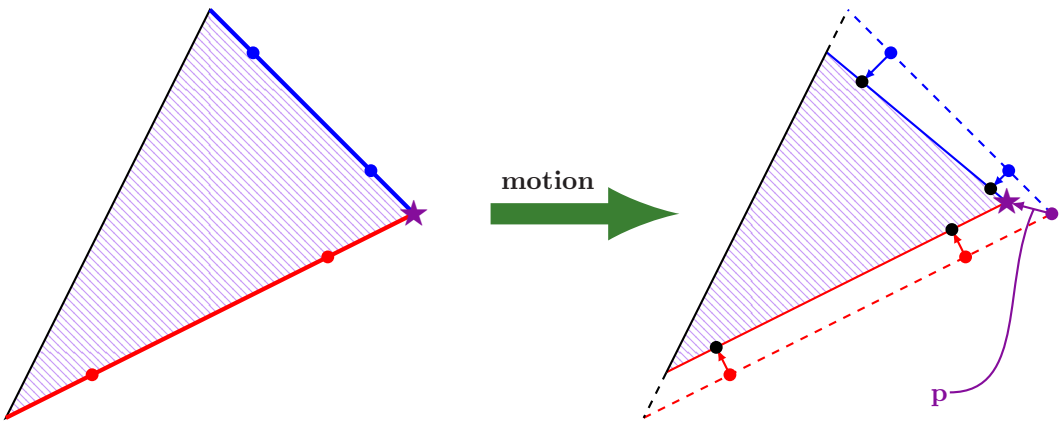


Fig. 2: Illustration of Algorithm 1. The left shows an element with two receding sides (in red and blue) along with the quadrature points on each edge. The corner node $\mathbf{x}^*$ is the purple star. Algorithm 1 computes the receded edges independently and then finds the location of the new node by intersecting the lines formed by a best-fit plane through the perturbed quadrature points, depicted on the left. Note that recession need not be uniform across an edge, as shown here.

While Figure 2 above depicts the case of a double receder, the algorithm cares not for the specific combination of sliding and receding edges. When one side is fixed, Algorithm 1 will not display the correct behavior when

the other side is receding; in implementation, it is advantageous to detect the presence of a fixed side set and immediately suppress all motion, eliminating the unnecessary overhead of the algorithm. When both sides are sliding, the end result is that the node will remain in its original location, so including logic to process that case quickly is also a boon to speed, although Algorithm 1 will also produce this effect.

2. Interior Nodes: Receders

This section will look very similar to the preceding one when written.

Summary of method: Recede each edge by best-fit line as we did in the corner case. However, the intersection between these edges may not be defined, if they were parallel to begin with and remained so after recession. Instead, we use the internal edge from the element home to the receding edges. The intersection of the new edge with the internal edge is computed. This gives two new points in total; the centroid of the two is taken and that becomes the new node location.

3. Interior Nodes: Sliders

A slider is characterized by forbidden motion in the normal direction. Up to this point, we have avoided asking for this elusive vector at nodes. If the boundary were smooth, there would be no issue. As elements are polygonal, the boundary of the computational domain is only Lipschitz, which means that the normal is only defined almost everywhere. The set of points where it is not defined is precisely the set of nodes—and precisely where we need it.

Because faces are straight lines between nodes, the normal is the same everywhere on a given face. On a particular face, the normal at the node can be defined via limiting procedure: for a fixed $\boldsymbol{\xi} \in \partial\Omega \cap \partial\mathcal{T}$:

$$\boldsymbol{\nu}(\boldsymbol{\xi} \to \mathbf{x}) = \lim_{h \to 0^+, h' \to 0^-} \boldsymbol{\nu}\big(x_1 + (\xi_1 - x_1)h, x_2 + (\xi_2 - x_2)h'\big) \tag{17}$$

Unfortunately, it is insufficient to consider only a single element, as shown in Figure 3.

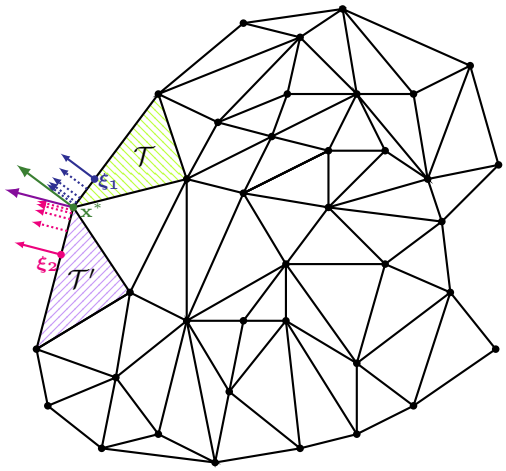


Fig. 3: Difficulties in Finding $\boldsymbol{\nu}$. In (a), for an "ordinary" $\mathcal{T}$ (shaded in green), (17) will always converge to the green vector independently of $\boldsymbol{\xi_1} \in \partial\mathcal{T} \cap \partial\Omega$. The steps of the limiting process are the dashed vectors. On $\mathcal{T}'$ (shaded in purple), the purple vector will be obtained regardless of the starting $\boldsymbol{\xi_2}$. This results in a node possessing a normal vector for each element in which it is contained.

On a per element basis, there is no issue in (a). However, the finite element basis functions $\boldsymbol{\psi}_k$ are constructed

from shape functions; using the new basis functions $\{\tilde{\varphi}_k\}$, we'd have something like

$$\psi_{k^*}(\mathbf{x}) = \begin{cases} \tilde{\varphi}_{k_1,\mathcal{T}}(\mathbf{x}) & \mathbf{x} \in \mathcal{T} \\ \tilde{\varphi}_{k_2,\mathcal{T}'}(\mathbf{x}) & \mathbf{x} \in \mathcal{T}' \\ \vdots \end{cases}$$

if the green node $\mathbf{x}^*$ is node number k_1 within $\mathcal{T}$, k_2 within $\mathcal{T}'$, and k^* overall. If different normals at $\mathbf{x}^*$ are allowed for different elements, we would have $\tilde{\boldsymbol{\varphi}}_{k_1,\mathcal{T}}(\mathbf{x}^*) \neq \tilde{\boldsymbol{\varphi}}_{k_2,\mathcal{T}'}(\mathbf{x}^*)$ so that $\boldsymbol{\psi}_{k^*}$ would no longer be continuous. Consequently, the "ordinary case" of (a) must be resolved by simultaneously considering all elements containing each boundary node. Fortunately, in $\mathbb{R}^2$, there can be only two such elements.

Let $\mathbf{x} \in \partial\mathcal{T} \cap \partial\mathcal{T}'$ and $\boldsymbol{\nu}_1 \triangleq \boldsymbol{\nu}(\boldsymbol{\xi}_{\mathcal{T}} \to \mathbf{x})$ and $\boldsymbol{\nu}_2 \triangleq \boldsymbol{\nu}(\boldsymbol{\xi}_{\mathcal{T}'} \to \mathbf{x})$ be the normals obtained from (17) applied within $\mathcal{T}$ and $\mathcal{T}'$, respectively. We then define the normal at $\mathbf{x}$ to be the "average" of the two normals:

$$\boldsymbol{\nu}(\mathbf{x}) = \frac{\boldsymbol{\nu}_1 + \boldsymbol{\nu}_2}{|\boldsymbol{\nu}_1 + \boldsymbol{\nu}_2|} \tag{18}$$

A visual depiction of (18) is shown below in Figure 4.

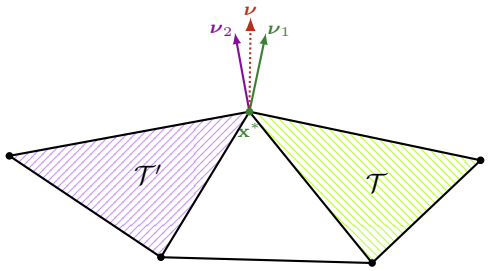


Fig. 4: The Solution to $\boldsymbol{\nu}$. $\boldsymbol{\nu}_1$ (in green, solid) and $\boldsymbol{\nu}_2$ (in purple, solid) are produced from the limiting process (17); $\boldsymbol{\nu}$ (in red, dashed) is the result of (18) applied to these two vectors.

D. Example Problems

This section will contain several examples of the two-dimensional scheme in action.

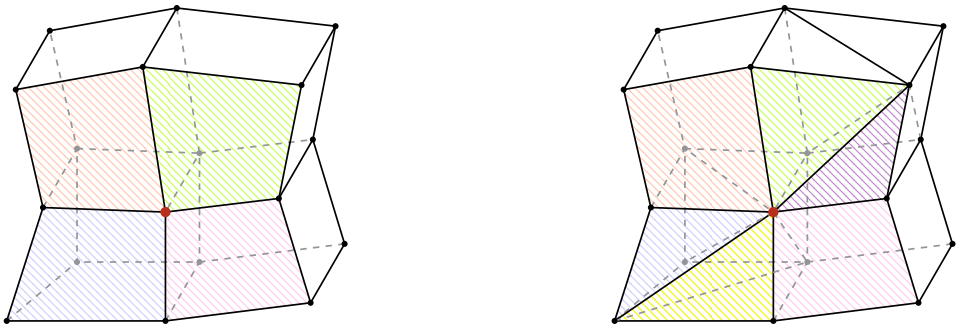
III. Three-Dimensional Meshes

The straightforward analysis possible in the 2D case can be attributed to two key facts:

1. Each node can be contained in at most two elements that intersect the boundary non-trivially.
2. Given a direction of motion, there are exactly two vectors orthogonal to it. Choosing appropriately allows us to construct a symmetric transformation matrix $\mathcal{M}$.

Unfortunately, neither of these hold in three dimensions. Consequently, the script for analysis diverges significantly from that 2D. An example of how (1) above may be violated is depicted in the figure below.

Another distinguishing feature of the 3D case is that there is more than one type of corner. In 2D, a node is a corner if it is either contained in two side sets or is contained in two faces within the same element. In 3D, because the number of containing elements may be arbitrary, attempts to characterize nodes based on elements is an exercise in futility. Instead, nodes are classified according to the user-specified side sets:

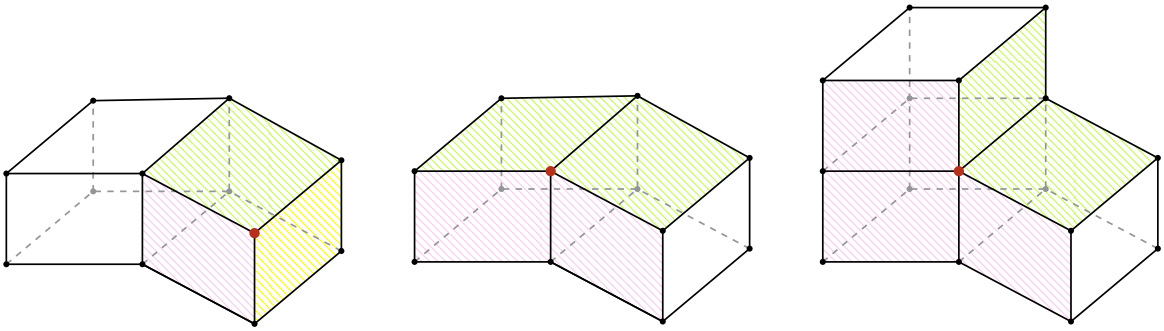


(a) **(b)**

Fig. 5: An Arbitrary Number of Boundary Faces: Consider the red node in each subfigure. Well-placed quadrilaterals lead to the "nicest" case of 4 containing elements, shown in **(a)**. Addition of mixed elements, such as tetrahedrons and pyramids in **(b)**, complicates things considerably.

Definition (Node Classification).

A boundary node is a *pure corner* node if it is contained in three side sets. It is an *edge corner* node if it is contained in two side sets. An *interior* node is contained in a single side set.



(a) **(b)** **(c)**

Fig. 6: Corners in 3D. Each side set is indicated by a different color. **(a)** depicts a pure corner. In this example, all three containing faces are in the same element, but this is not always the case. **(b)** and **(c)** depict edge corners. A bevy of configurations is possible, but there can be only two side sets involved.

The use of side sets to classify nodes shifts burden onto the user during grid generation. This is necessary because there is no way to tell from the mesh geometry alone how a node should be treated. By using side sets for identification, control remains with the user so that no errenous processing will occur.

A. General Framework

As we did in 2D, we begin with the vector-valued shape functions on an element $\mathcal{T}$

$$\boldsymbol{\varphi}_k(\mathbf{x}) = \begin{cases} \psi_k(\mathbf{x})\,\mathbf{e}_1 & \text{if } 1 \le k \le N \\ \psi_{k-N}(\mathbf{x})\,\mathbf{e}_2 & \text{if } N+1 \le k \le 2N \\ \psi_{k-2N}(\mathbf{x})\,\mathbf{e}_3 & \text{if } 2N+1 \le k \le 3N \end{cases}$$

and define the new shape functions

$$\tilde{\boldsymbol{\varphi}}_k(\mathbf{x}) = \begin{cases} \psi_k(\mathbf{x})\,\mathbf{p}_k & 1 \le k \le m \\ \psi_k(\mathbf{x})\,\mathbf{e}_1 & m+1 \le k \le N \\ \psi_{k-N}(\mathbf{x})\,\mathbf{q}_{k-N} & N+1 \le k \le m+N \\ \psi_{k-N}(\mathbf{x})\,\mathbf{e}_2 & m+N+1 \le k \le 2N \\ \psi_{k-2N}(\mathbf{x})\,\mathbf{r}_{k-2N} & 2N+1 \le k \le m+2N \\ \psi_{k-2N}(\mathbf{x})\,\mathbf{e}_3 & m+2N+1 \le k \le 3N \end{cases}$$

The directions $\mathbf{p}_k$, $\mathbf{q}_k$, and $\mathbf{r}_k$ should be considered arbitrary at the moment. We mirror the procedure that led to (6) and seek a matrix that encodes the transformation from old to new elements via

$$\tilde{\boldsymbol{\varphi}}_k(\mathbf{x}) = \sum_{l=1}^{3N} \mathcal{M}_{kl}\boldsymbol{\varphi}_l(\mathbf{x})$$

To extend the previous result (**??**) to three dimensions, we need to construct $\mathcal{M}$ to be symmetric. In 2D, restriction to a single direction of motion makes this task straightforward, as creating an orthogonal basis yielding a symmetric $\mathcal{M}$ is merely a matter of choosing the correct sign. In 3D, it is necessary to enforce multiple directions of motion. In such cases, a symmetric $\mathcal{M}$ is beyond the realm of possibility, as that would require that every invertible matrix be similar to a symmetric one. Consequently, developing the framework for obtaining the correction to the stiffness matrix $\mathcal{A}\big|_{\mathcal{T}}$ when $\mathcal{M}$ is not symmetric is our foremost task.

B. Applying the Corrections

1. Element Stiffness Matrix

Suppose the directions of motion $\mathbf{p}_k$, $\mathbf{q}_k$, and $\mathbf{r}_k$ have been predetermined and that these directions are linearly independent. Furthermore, suppose that $p_k^{(1)} \neq 0$, $q_k^{(2)} \neq 0$, and $r_k^{(3)} \neq 0$. Given a set of vectors that do not meet this criterion, we can always relabel them according to the following:

Lemma 3 (Rearrangement Algorithm).

Let $\mathbf{v}_{k_1}$, $\mathbf{v}_{k_2}$, and $\mathbf{v}_{k_3}$ be linearly independent vectors in $\mathbb{R}^3$. Let

$$(k_1', k_2', k_3') \in \underset{\pi \in \Pi(k_1,k_2,k_3)}{\arg\min} \operatorname{trace}(\operatorname{diag}\,\mathbf{Q}_\pi)^{-1} \tag{19}$$

where $\Pi(k_1, k_2, k_3)$ denotes the set of permutations of $\{k_1, k_2, k_3\}$, $\operatorname{diag}(\cdot)$ is the operator that extracts diagonal elements from a matrix, and for $\pi = (k_1'', k_2'', k_3'')$ the matrix $\mathbf{Q}_\pi$ is

$$\mathbf{Q}_\pi \triangleq \begin{bmatrix} \left|v_{k_1''}^{(1)}\right| & \left|v_{k_1''}^{(2)}\right| & \left|v_{k_1''}^{(3)}\right| \\ \left|v_{k_2''}^{(1)}\right| & \left|v_{k_2''}^{(2)}\right| & \left|v_{k_2''}^{(3)}\right| \\ \left|v_{k_3''}^{(1)}\right| & \left|v_{k_3''}^{(2)}\right| & \left|v_{k_3''}^{(3)}\right| \end{bmatrix}$$

Then $v_{k_1'}^{(1)} \neq 0$, $v_{k_2'}^{(2)} \neq 0$, and $v_{k_3'}^{(3)} \neq 0$.

Proof.

The result can be seen clearly by writing out the trace (19) explicitly:

$$\operatorname{trace}(\operatorname{diag} \mathbf{Q}_\pi)^{-1} = \frac{1}{\left|v_{k_1''}^{(1)}\right|} + \frac{1}{\left|v_{k_2''}^{(2)}\right|} + \frac{1}{\left|v_{k_3''}^{(3)}\right|} \tag{20}$$

If one of $v_{k_1''}^{(1)}$, $v_{k_2''}^{(2)}$, or $v_{k_3''}^{(3)}$ is zero, (20) will be infinite, and thus cannot be the minimum unless all permutations result in this degeneracy. We now show that since span $\{\mathbf{v}_{k_1}, \mathbf{v}_{k_2}, \mathbf{v}_{k_3}\} = \mathbb{R}^3$, at least one arrangement results in finite (20). Let $n_z(\mathbf{v})$ count the number of zero components of a vector $\mathbf{v} \in \mathbb{R}^3$. WLOG, we may suppose that $n_z(\mathbf{v}_{k_1}) \geq n_z(\mathbf{v}_{k_2}) \geq n_z(\mathbf{v}_{k_3})$.

Case I: $n_z(\mathbf{v}_{k_1}) = 2$.
Because span $\{\mathbf{v}_{k_1}, \mathbf{v}_{k_2}, \mathbf{v}_{k_3}\} = \mathbb{R}^3$, we have

$$\det \begin{bmatrix} \mathbf{v}_{k_1} & \mathbf{v}_{k_2} & \mathbf{v}_{k_3} \end{bmatrix} \neq 0$$

Let i be the nonzero component of $\mathbf{v}_{k_1}$ and i' amd i'' be the zero components. Then performing a Laplace expansion on the m^{th} column to evaluate the above determinant, we have

$$v_{k_1}^{(i)} \det \underbrace{\begin{bmatrix} v_{k_2}^{(i')} & v_{k_3}^{(i')} \\ v_{k_2}^{(i'')} & v_{k_3}^{(i'')} \end{bmatrix}}_{\triangleq \mathcal{A}} \neq 0$$

or that $\det \mathcal{A} \neq 0$. Consequently, each column of $\mathcal{A}$ has at least one nonzero entry; hence, at least one of $\pi = (k_1, k_2, k_3)$ or $\pi = (k_1, k_3, k_2)$ makes (20) finite.

Case II: $n_z(\mathbf{v}_{k_1}) = 1$.
At least one of $v_{k_1}^{(1)} \neq 0$, $v_{k_2}^{(1)} \neq 0$, or $v_{k_3}^{(1)} \neq 0$ holds; suppose one is $v_m^{(1)} \neq 0$. Set $k_1'' = m$. It then follows that only one of $v_m^{(2)} \neq 0$ or $v_m^{(3)} \neq 0$ is true.

Subcase II.1: $\boldsymbol{v_m^{(2)} = 0}$.
At most one of $v_{m'}^{(2)} = 0$ or $v_{m''}^{(2)} = 0$ is true, where $\{m', m''\} = \{k_1, k_2, k_3\} \backslash \{m\}$.

Subcase II.1.1: $\boldsymbol{v_{m'}^{(2)} \neq 0}$ **and** $\boldsymbol{v_{m''}^{(2)} \neq 0}$. At most one of $v_{m'}^{(3)} = 0$ or $v_{m''}^{(3)} = 0$. Put

$$k_3'' = \begin{cases} m' & v_{m''}^{(3)} = 0 \\ m'' & v_{m'}^{(3)} = 0 \end{cases} \qquad k_2'' = \begin{cases} m'' & v_{m''}^{(3)} = 0 \\ m' & v_{m'}^{(3)} = 0 \end{cases}$$

If $v_{m''}^{(3)} v_{m'}^{(3)} \neq 0$, then we may assign m' and m'' to k_2'' and k_3'' indiscriminately.

Subcase II.1.2: $\boldsymbol{v_{m'}^{(2)} = 0}$. Because $n_z(\mathbf{v}_{m'}) \leq 1$, it must be that $v_{m'}^{(3)} \neq 0$. Put $k_3'' = m'$. Then $v_{m''}^{(2)} \neq 0$, so put $k_2'' = m''$ to give a permutation that makes (20) finite.

Subcase II.2: $\boldsymbol{v_m^{(3)} = 0}$. Repeat Subcase II.1 with the symbol swap (2) $\leftrightarrows$ (3).

Case III: $n_z(\mathbf{v}_{k_1}) = 0$.
All components of $\mathbf{v}_{k_1}$, $\mathbf{v}_{k_2}$, and $\mathbf{v}_{k_3}$ are nonzero. All permutations yield a finite (20). ■

With the directions properly arranged by the Rearrangement Algorithm, we put

$$\mathcal{M} = \left[\begin{array}{ccc|ccc:ccc:ccc:ccc}
p_1^{(1)} & & 0 & 0 & \cdots & 0 & p_1^{(2)} & & 0 & 0 & \cdots & 0 & p_1^{(3)} & & 0 & 0 & \cdots & 0 \\
 & \ddots & & \vdots & \ddots & \vdots & & \ddots & & \vdots & \ddots & \vdots & & \ddots & & \vdots & \ddots & \vdots \\
0 & & p_m^{(1)} & 0 & \cdots & 0 & 0 & & p_m^{(2)} & 0 & \cdots & 0 & 0 & & p_m^{(3)} & 0 & \cdots & 0 \\
\hline
0 & \cdots & 0 & 1 & & 0 & 0 & \cdots & 0 & 0 & \cdots & 0 & 0 & \cdots & 0 & 0 & \cdots & 0 \\
\vdots & \ddots & \vdots & & \ddots & & \vdots & \ddots & \vdots & \vdots & \ddots & \vdots & \vdots & \ddots & \vdots & \vdots & \ddots & \vdots \\
0 & \cdots & 0 & 0 & & 1 & 0 & \cdots & 0 & 0 & \cdots & 0 & 0 & \cdots & 0 & 0 & \cdots & 0 \\
\hline
q_1^{(1)} & & 0 & 0 & \cdots & 0 & q_1^{(2)} & & 0 & 0 & \cdots & 0 & q_1^{(3)} & & 0 & 0 & \cdots & 0 \\
 & \ddots & & \vdots & \ddots & \vdots & & \ddots & & \vdots & \ddots & \vdots & & \ddots & & \vdots & \ddots & \vdots \\
0 & & q_m^{(1)} & 0 & \cdots & 0 & 0 & & q_m^{(2)} & 0 & \cdots & 0 & 0 & & q_m^{(3)} & 0 & \cdots & 0 \\
0 & \cdots & 0 & 0 & \cdots & 0 & 0 & \cdots & 0 & 1 & & 0 & 0 & \cdots & 0 & 0 & \cdots & 0 \\
\vdots & \ddots & \vdots & \vdots & \ddots & \vdots & \vdots & \ddots & \vdots & & \ddots & & \vdots & \ddots & \vdots & \vdots & \ddots & \vdots \\
0 & \cdots & 0 & 0 & \cdots & 0 & 0 & \cdots & 0 & 0 & & 1 & 0 & \cdots & 0 & 0 & \cdots & 0 \\
\hline
r_1^{(1)} & & 0 & 0 & \cdots & 0 & r_1^{(2)} & & 0 & 0 & \cdots & 0 & r_1^{(3)} & & 0 & 0 & \cdots & 0 \\
 & \ddots & & \vdots & \ddots & \vdots & & \ddots & & \vdots & \ddots & \vdots & & \ddots & & \vdots & \ddots & \vdots \\
0 & & r_m^{(1)} & 0 & \cdots & 0 & 0 & & r_m^{(2)} & 0 & \cdots & 0 & 0 & & r_m^{(3)} & 0 & \cdots & 0 \\
0 & \cdots & 0 & 0 & \cdots & 0 & 0 & \cdots & 0 & 0 & \cdots & 0 & 0 & \cdots & 0 & 1 & & 0 \\
\vdots & \ddots & \vdots & \vdots & \ddots & \vdots & \vdots & \ddots & \vdots & \vdots & \ddots & \vdots & \vdots & \ddots & \vdots & & \ddots & \\
0 & \cdots & 0 & 0 & \cdots & 0 & 0 & \cdots & 0 & 0 & \cdots & 0 & 0 & \cdots & 0 & 0 & & 1
\end{array}\right] \tag{21}$$

By the Rearrangement Lemma, each diagonal block of $\mathcal{M}$ is nonsingular. A secondary secondary benefit of assigning the directions in this manner is that the matrices

$$\mathbf{M}_{11} \triangleq \begin{bmatrix} p_1^{(1)} & & \\ & \ddots & \\ & & p_m^{(1)} \end{bmatrix} \qquad \mathbf{M}_{33} \triangleq \begin{bmatrix} q_1^{(2)} & & \\ & \ddots & \\ & & q_m^{(2)} \end{bmatrix} \qquad \mathbf{M}_{55} \triangleq \begin{bmatrix} r_1^{(3)} & & \\ & \ddots & \\ & & r_m^{(3)} \end{bmatrix}$$

should be reasonably well-conditioned: if one of $\left|v_{k_i'}^{(i)}\right| \ll 1$, then per (20), we will have trace(diag $\mathbf{Q}_\pi)^{-1} \gg 1$. Such a permutation is unlikely to yield the minumum trace, with one exception: a poor quality grid may yield a column in $\mathbf{Q}_\pi$ with nothing but small entries. In this case, there will be no avoiding poor conditioning.

Invertibility of $\mathcal{M}$ is necessary to return to the original finite element basis $\boldsymbol{\varphi}_k$ from $\tilde{\boldsymbol{\varphi}}_k$. First, observe that we may permute $\mathcal{M}$ into the following form:

$$\mathcal{M}' = \left[\begin{array}{ccc|c} \mathbf{C}_1 & & \mathbf{0} & \mathbf{0} \\ & \ddots & & \vdots \\ \mathbf{0} & & \mathbf{C}_m & \mathbf{0} \\ \hline \mathbf{0} & \cdots & \mathbf{0} & \mathbf{I} \end{array}\right]$$

where $\mathbf{C}_k$ is the change of basis matrix

$$\mathbf{C}_k \triangleq \begin{bmatrix} \mathbf{p}_k^T \\ \mathbf{q}_k^T \\ \mathbf{r}_k^T \end{bmatrix}$$

The directions $\mathbf{p}_k$, $\mathbf{q}_k$, and $\mathbf{r}_k$ are assumed to be linearly independent. Thus, each $\mathbf{C}_k$ is invertible, and so is $\mathcal{M}'$. As there exists a nonsingular matrix $\mathcal{P}$ such that $\mathcal{M} = \mathcal{P}\mathcal{M}'$, it follows that $\mathcal{M}$ is also nonsingular.

Partition $\mathcal{M}$ along the thick lines of (21) as follows:

$$\mathcal{M} = \left[\begin{array}{c|c|c|c|c|c} \mathbf{M}_{11} & \mathbf{0} & \mathbf{M}_{13} & \mathbf{0} & \mathbf{M}_{15} & \mathbf{0} \\ \hline \mathbf{0} & \mathbf{I} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \mathbf{M}_{31} & \mathbf{0} & \mathbf{M}_{33} & \mathbf{0} & \mathbf{M}_{35} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{I} & \mathbf{0} & \mathbf{0} \\ \mathbf{M}_{51} & \mathbf{0} & \mathbf{M}_{53} & \mathbf{0} & \mathbf{M}_{55} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{I} \end{array}\right] \triangleq \left[\begin{array}{c|c} \mathcal{A} & \mathcal{B} \\ \hline \mathcal{C} & \mathcal{D} \end{array}\right]$$

where the blocks of the rightmost side correspond to the dividing lines on the left side. By construction, $\mathcal{A} = \mathbf{M}_{11}$ is invertible. To see that $\mathcal{D}^{-1}$ exists, we permute $\mathcal{M}$ into the form

$$\mathcal{M}'' = \left[\begin{array}{c|c|c|c|c|c} \mathbf{I} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{M}_{33} & \mathbf{0} & \mathbf{M}_{35} & \mathbf{0} & \mathbf{M}_{31} \\ \mathbf{0} & \mathbf{0} & \mathbf{I} & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{M}_{53} & \mathbf{0} & \mathbf{M}_{55} & \mathbf{0} & \mathbf{M}_{51} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{I} & \mathbf{0} \\ \hline \mathbf{0} & \mathbf{M}_{13} & \mathbf{0} & \mathbf{M}_{15} & \mathbf{0} & \mathbf{M}_{11} \end{array}\right] = \left[\begin{array}{c|c} \mathcal{D} & \mathcal{C} \\ \hline \mathcal{B} & \mathcal{A} \end{array}\right]$$

and confirm by direct computation (which we omit here) that $\mathcal{M}''$ has an LU-decomposition because $\mathbf{M}_{33}$ and $\mathbf{M}_{55}$ are invertible. We then recall that a matrix has an LU-decomposition if and only if all leading principle submatrices are nonsingular [Bernstein, p.260]. Because $\mathcal{D}$ is such for $\mathcal{M}''$, it must be nonsingular.

Recall the Schur complement $\mathcal{S}$ and "Haynsworth complement" $\mathcal{H}$ of $\mathcal{M}$:

$$\mathcal{S} \triangleq \mathcal{A} - \mathcal{B}\mathcal{D}^{-1}\mathcal{C}$$
$$\mathcal{H} \triangleq \mathcal{D} - \mathcal{C}\mathcal{A}^{-1}\mathcal{B}$$

We can further compute the determinant of $\mathcal{M}$ by block:

$$\begin{aligned} \det(\mathcal{M}) &= \det(\mathcal{A})\det(\mathcal{H}) \\ &= \det(\mathcal{D})\det(\mathcal{S}) \end{aligned}$$

which gives that both $\mathcal{S}^{-1}$ and $\mathcal{H}^{-1}$ exist. Knowing this, we recall the following result [B., p.44-45]:

Lemma 4 (Block Inversion).

Let $\mathcal{A} \in \mathbb{C}^{n\times n}$, $\mathcal{B} \in \mathbb{C}^{n\times m}$, $\mathcal{C} \in \mathbb{C}^{m\times n}$, and $\mathcal{D} \in \mathbb{C}^{m\times m}$. If $\mathcal{A}$ and $\mathcal{S}$ are nonsingular, then

$$\left[\begin{array}{c|c} \mathcal{A} & \mathcal{B} \\ \hline \mathcal{C} & \mathcal{D} \end{array}\right]^{-1} = \left[\begin{array}{c|c} \mathcal{S}^{-1} & -\mathcal{S}^{-1}\mathcal{B}\mathcal{D}^{-1} \\ \hline -\mathcal{D}^{-1}\mathcal{C}\mathcal{S}^{-1} & \mathcal{D}^{-1} + \mathcal{D}^{-1}\mathcal{C}\mathcal{S}^{-1}\mathcal{B}\mathcal{D}^{-1} \end{array}\right]$$

If $\mathcal{D}$ and $\mathcal{H}$ are nonsingular, then

$$\left[\begin{array}{c|c} \mathcal{A} & \mathcal{B} \\ \hline \mathcal{C} & \mathcal{D} \end{array}\right]^{-1} = \left[\begin{array}{c|c} \mathcal{A}^{-1} + \mathcal{A}^{-1}\mathcal{B}\mathcal{H}^{-1}\mathcal{C}\mathcal{A}^{-1} & -\mathcal{A}^{-1}\mathcal{B}\mathcal{H}^{-1} \\ \hline -\mathcal{H}^{-1}\mathcal{C}\mathcal{A}^{-1} & \mathcal{H}^{-1} \end{array}\right]$$

Each $\mathbf{M}_{kl} \in \mathbb{R}^{m\times m}$ is diagonal and $\mathbf{I}$ denotes the identity matrix of size $N-m$; the sizes of the zero blocks follow from context. To the return to stiffness matrix in terms of the old shape functions from the new, we must use equation (8) and **not** (**??**) because $\mathcal{M}$ is not symmetric in this case. As in 2D, we have

$$\begin{aligned}\mathbf{A}\big|_{\mathcal{T}} &= \mathcal{M}^{-T}\tilde{\mathbf{K}}\mathcal{M}^{-1} \\ &= \mathbf{K} + \mathcal{M}^{-T}\mathbf{P}\mathcal{M}^{-1}\end{aligned} \tag{22}$$

Depending on the classification of the node, one or more directions may be unenforced. That is, we allow motion in that direction to be determined by the dynamics of the system. Because the order of directions is set by (19), $\mathbf{P}$ is dependent on its classification. Fortunately, for each boundary node $\mathbf{x}_k$, we can write the general form

$$\mathbf{P}_k = \varepsilon^{-1}\Big[f_{0,k}\,\mathbf{e}_{i_0}\mathbf{e}_{i_0}^T + f_{1,k}\,\mathbf{e}_{i_1}\mathbf{e}_{i_1}^T + f_{2,k}\,\mathbf{e}_{i_2}\mathbf{e}_{i_2}^T\Big]$$

where $i_0 \triangleq k$, $i_1 \triangleq k+N$, and $i_2 \triangleq k+2N$, and

$$f_{0,k} = \begin{cases} 1 & \mathbf{p}_k \text{ enforced} \\ 0 & \text{otherwise}\end{cases} \qquad f_{1,k} = \begin{cases} 1 & \mathbf{q}_k \text{ enforced} \\ 0 & \text{otherwise}\end{cases} \qquad f_{2,k} = \begin{cases} 1 & \mathbf{r}_k \text{ enforced} \\ 0 & \text{otherwise}\end{cases}$$

so that the full form of $\mathbf{P}$ is given by

$$\mathbf{P} = \sum_{k=1}^{m}\mathbf{P}_k \tag{23}$$

Consequently, we need only compute

$$\mathcal{M}^{-T}\big(\mathbf{e}_j\,\mathbf{e}_j^T\big)\mathcal{M}^{-1} = \big(\mathcal{M}^{-T}\mathbf{e}_j\big)\big(\mathcal{M}^{-T}\mathbf{e}_j\big)^T$$

for $j = i_0, i_1, i_2$ to generate everything we will need to assemble $\mathbf{A}\big|_{\mathcal{T}}$ on the fly.

Ordinarily, inversion of a matrix is a tall order and recommended only if it can be done in exact arithmetic. Fortituous for us, because the blocks of $\mathcal{M}$ consist of entirely diagonal matrices, $\mathcal{M}^{-T}$ is not nearly so daunting to construct. Observe that within the Block Inversion Lemma, the blocks are the same size so that

$$\begin{aligned}\mathcal{S}^{-1} &= \mathcal{A}^{-1} + \mathcal{A}^{-1}\mathcal{B}\mathcal{H}^{-1}\mathcal{C}\mathcal{A}^{-1} & \mathcal{S}^{-1}\mathcal{B}\mathcal{D}^{-1} &= \mathcal{A}^{-1}\mathcal{B}\mathcal{H}^{-1} \\ \mathcal{H}^{-1} &= \mathcal{D}^{-1} + \mathcal{D}^{-1}\mathcal{C}\mathcal{S}^{-1}\mathcal{B}\mathcal{D}^{-1} & \mathcal{D}^{-1}\mathcal{C}\mathcal{S}^{-1} &= \mathcal{H}^{-1}\mathcal{C}\mathcal{A}^{-1}\end{aligned}$$

as long as all these quantities exist. We previously established that they do; consequently, we may "pick and choose" which among these is most convenient to compute. Because $\mathcal{S}$ is $m\times m$ and diagonal, it is trivial to invert. Formation of $\mathcal{H}$ requires inverting $\mathcal{A}$, again $m\times m$, but then we must invert $\mathcal{H}$, which is $(N-m)\times(N-m)$, the same size as $\mathcal{D}$. Hence, there is no advantage offered by the alternate form of the Block Inversion Lemma. As $\mathcal{D}$ is already known, we choose to invert it. We begin by partitioning $\mathcal{D}$ as

$$\mathcal{D} = \left[\begin{array}{c|ccc|c} \mathbf{I} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \hline \mathbf{0} & \mathbf{M}_{33} & \mathbf{0} & \mathbf{M}_{35} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{I} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{M}_{53} & \mathbf{0} & \mathbf{M}_{55} & \mathbf{0} \\ \hline \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{I} \end{array}\right] \triangleq \left[\begin{array}{c|c|c} \mathbf{I} & \mathbf{0} & \mathbf{0} \\ \hline \mathbf{0} & \mathbf{D}_r & \mathbf{0} \\ \hline \mathbf{0} & \mathbf{0} & \mathbf{I} \end{array}\right] \tag{24}$$

so that we need only compute $\mathbf{D}_r^{-1}$. This time, however, is *is* convenient to apply the "pick and choose" version of the Block Inversion Lemma. Doing so, we can obtain an expression for $\mathbf{D}_r^{-1}$:

$$\mathbf{D}_r^{-1} = \begin{bmatrix} \mathbf{S}^{-1} & \mathbf{0} & -\mathbf{S}^{-1}\mathbf{M}_{35}\mathbf{M}_{55}^{-1} \\ \mathbf{0} & \mathbf{I} & \mathbf{0} \\ -\mathbf{H}_r^{-1}\mathbf{M}_{53}\mathbf{M}_{33}^{-1} & \mathbf{0} & \mathbf{H}_r^{-1} \end{bmatrix} \tag{25}$$

where $\mathbf{S}$ is the Schur complement of $\mathbf{D}_r$ and $\mathbf{H}_r$ is the reduced Haynsworth complement of $\mathbf{D}_r$,

$$\mathbf{S} = \mathbf{M}_{33} - \mathbf{M}_{35}\mathbf{M}_{55}^{-1}\mathbf{M}_{53}$$
$$\mathbf{H}_r = \mathbf{M}_{55} - \mathbf{M}_{53}\mathbf{M}_{33}^{-1}\mathbf{M}_{35}$$

With $\mathcal{D}^{-1}$ in hand, we can write expressions for $\mathcal{S}$ and $\mathcal{S}^{-1}\mathcal{B}\mathcal{D}^{-1}$:

$$\mathcal{S} = \mathbf{M}_{11} - \mathbf{M}_{13}\mathbf{S}^{-1}\big(\mathbf{M}_{31} - \mathbf{M}_{35}\mathbf{M}_{55}^{-1}\mathbf{M}_{51}\big) - \mathbf{M}_{15}\mathbf{H}_r^{-1}\big(\mathbf{M}_{51} - \mathbf{M}_{53}\mathbf{M}_{33}^{-1}\mathbf{M}_{31}\big)$$
$$\mathcal{S}^{-1}\mathcal{B}\mathcal{D}^{-1} = \begin{bmatrix}\mathbf{0} & -\mathcal{S}^{-1}\mathcal{P}_1 & \mathbf{0} & -\mathcal{S}^{-1}\mathcal{P}_2 & \mathbf{0}\end{bmatrix}$$

where

$$\mathcal{P}_1 \triangleq \mathbf{M}_{15}\mathbf{H}_r^{-1}\mathbf{M}_{53}\mathbf{M}_{33}^{-1} - \mathbf{M}_{13}\mathbf{S}^{-1} \tag{26}$$
$$\mathcal{P}_2 \triangleq \mathbf{M}_{13}\mathbf{S}^{-1}\mathbf{M}_{35}\mathbf{M}_{55}^{-1} - \mathbf{M}_{15}\mathbf{H}_r^{-1} \tag{27}$$

To get $\mathcal{M}^{-T}$ from here, it is simply a matter of multiplying everything out and transposing:

$$\mathcal{M}^{-T} = \begin{bmatrix}
\mathcal{S}^{-1} & \mathbf{0} & \mathcal{S}^{-1}\mathcal{P}_3 & \mathbf{0} & \mathcal{S}^{-1}\mathcal{P}_4 & \mathbf{0} \\
\mathbf{0} & \mathbf{I} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} \\
\mathcal{S}^{-1}\mathcal{P}_1 & \mathbf{0} & \mathbf{S}^{-1} + \mathcal{S}^{-1}\mathcal{P}_3\mathcal{P}_5 & \mathbf{0} & \mathcal{S}^{-1}\mathcal{P}_4\mathcal{P}_5 - \mathbf{H}_r^{-1}\mathbf{M}_{53}\mathbf{M}_{33}^{-1} & \mathbf{0} \\
\mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{I} & \mathbf{0} & \mathbf{0} \\
\mathcal{S}^{-1}\mathcal{P}_2 & \mathbf{0} & \mathcal{S}^{-1}\mathcal{P}_3\mathcal{P}_6 - \mathbf{S}^{-1}\mathbf{M}_{35}\mathbf{M}_{55}^{-1} & \mathbf{0} & \mathbf{H}_r^{-1} + \mathcal{S}^{-1}\mathcal{P}_4\mathcal{P}_6 & \mathbf{0} \\
\mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{I}
\end{bmatrix} \tag{28}$$

where

$$\mathcal{P}_3 \triangleq -\mathbf{S}^{-1}\left(\mathbf{M}_{31} - \mathbf{M}_{35}\mathbf{M}_{55}^{-1}\mathbf{M}_{51}\right) \tag{29}$$
$$\mathcal{P}_4 \triangleq -\mathbf{H}_r^{-1}\big(\mathbf{M}_{51} - \mathbf{M}_{53}\mathbf{M}_{33}^{-1}\mathbf{M}_{31}\big) \tag{30}$$
$$\mathcal{P}_5 \triangleq \mathbf{M}_{15}\mathbf{H}_r^{-1}\mathbf{M}_{53}\mathbf{M}_{33}^{-1} - \mathbf{M}_{13}\mathbf{S}^{-1} \tag{31}$$
$$\mathcal{P}_6 \triangleq \mathbf{M}_{13}\mathbf{S}^{-1}\mathbf{M}_{35}\mathbf{M}_{55}^{-1} - \mathbf{M}_{15}\mathbf{H}_r^{-1} \tag{32}$$

In implementation, due to the diagonality of all matrices invovled, full matrix multiplication may be avoided entirely by working node-by-node. To this end, we compute the ith diagonal entry of all the ingredients needed to create $\mathcal{M}^{-T}$:

$$S^{(i,i)} = q_i^{(2)} - \frac{q_i^{(3)} r_i^{(2)}}{r_i^{(3)}} \qquad P_3^{(i,i)} = -\frac{1}{S^{(i,i)}}\left(q_i^{(1)} - \frac{q_i^{(3)} r_i^{(1)}}{r_i^{(3)}}\right)$$

$$H_r^{(i,i)} = r_i^{(3)} - \frac{q_i^{(3)} r_i^{(2)}}{q_i^{(2)}} \qquad P_4^{(i,i)} = -\frac{1}{H_r^{(i,i)}}\left(r_i^{(1)} - \frac{r_i^{(3)} q_i^{(1)}}{q_i^{(2)}}\right)$$

$$P_1^{(i,i)} = \frac{p_i^{(3)} r_i^{(2)}}{H_r^{(i,i)} q_i^{(2)}} - \frac{p_i^{(2)}}{S^{(i,i)}} \qquad P_5^{(i,i)} = \frac{p_i^{(3)} r_i^{(2)}}{H_r^{(i,i)} q_i^{(2)}} - \frac{p_i^{(2)}}{S^{(i,i)}}$$

$$P_2^{(i,i)} = \frac{p_i^{(2)} q_i^{(3)}}{S^{(i,i)} r_i^{(3)}} - \frac{p_i^{(3)}}{H_r^{(i,i)}} \qquad P_6^{(i,i)} = \frac{p_i^{(2)} q_i^{(3)}}{S^{(i,i)} r_i^{(3)}} - \frac{p_i^{(3)}}{H_r^{(i,i)}}$$

$$\mathcal{S}^{(i,i)} = p_i^{(1)} - \frac{p_i^{(2)}}{S^{(i,i)}}\left(q_i^{(1)} - \frac{q_i^{(3)} r_i^{(1)}}{r_i^{(3)}}\right) - \frac{p_i^{(3)}}{H_r^{(i,i)}}\left(r_i^{(1)} - \frac{r_i^{(2)} q_i^{(1)}}{q_i^{(2)}}\right)$$

Noting that when a matrix multiplies $\mathbf{e}_k$, the result is the extraction of the k^{th} column, we have

$$\mathcal{M}^{-T}\mathbf{e}_{i_0} = \Big[\underbrace{0,\ldots,0}_{k-1},\beta_k^{(1)},\underbrace{0,\ldots,0}_{N-k},\beta_k^{(2)},\underbrace{0,\ldots,0}_{N-1},\beta_k^{(3)},\underbrace{0,\ldots,0}_{N-k}\Big]^T \tag{33}$$

$$\mathcal{M}^{-T}\mathbf{e}_{i_1} = \Big[\underbrace{0,\ldots,0}_{k-1},\gamma_k^{(1)},\underbrace{0,\ldots,0}_{N-k},\gamma_k^{(2)},\underbrace{0,\ldots,0}_{N-1},\gamma_k^{(3)},\underbrace{0,\ldots,0}_{N-k}\Big]^T \tag{34}$$

$$\mathcal{M}^{-T}\mathbf{e}_{i_2} = \Big[\underbrace{0,\ldots,0}_{k-1},\delta_k^{(1)},\underbrace{0,\ldots,0}_{N-k},\delta_k^{(2)},\underbrace{0,\ldots,0}_{N-1},\delta_k^{(3)},\underbrace{0,\ldots,0}_{N-k}\Big]^T \tag{35}$$

where

$$\begin{aligned}
\beta_k^{(1)} &= \frac{1}{\mathcal{S}^{(k,k)}} & \gamma_k^{(1)} &= \frac{P_3^{(k,k)}}{\mathcal{S}^{(k,k)}} & \delta_k^{(1)} &= \frac{P_4^{(k,k)}}{\mathcal{S}^{(k,k)}} \\
\beta_k^{(2)} &= \frac{P_1^{(k,k)}}{\mathcal{S}^{(k,k)}} & \gamma_k^{(2)} &= \frac{1}{S^{(k,k)}} + \frac{P_3^{(k,k)}P_5^{(k,k)}}{\mathcal{S}^{(k,k)}} & \delta_k^{(2)} &= \frac{P_4^{(k,k)}P_5^{(k,k)}}{\mathcal{S}^{(k,k)}} - \frac{r_k^{(2)}}{H_r^{(k,k)}q_k^{(2)}} \\
\beta_k^{(3)} &= \frac{P_2^{(k,k)}}{\mathcal{S}^{(k,k)}} & \gamma_k^{(3)} &= \frac{P_3^{(k,k)}P_6^{(k,k)}}{\mathcal{S}^{(k,k)}} - \frac{q_k^{(3)}}{S^{(k,k)}r_k^{(3)}} & \delta_k^{(3)} &= \frac{1}{H_r^{(k,k)}} + \frac{P_4^{(k,k)}P_6^{(k,k)}}{\mathcal{S}^{(k,k)}}
\end{aligned}$$

Finally, to assist in assembling (22), let

$$\begin{aligned}
\mathbf{R}_k &\triangleq \mathcal{M}^{-T}\mathbf{P}_k\mathcal{M}^{-1} \\
&= \varepsilon^{-1}\sum_{j=1}^{3} f_{j,k}\big(\mathcal{M}^{-T}\mathbf{e}_{i_j}\big)\big(\mathcal{M}^{-T}\mathbf{e}_{i_j}\big)^T
\end{aligned}$$

Per (34)–(35), there are only nine nonzero entries in $\mathbf{R}_k$:

$$\left.\begin{aligned}
R_k^{(i_0,i_0)} &= \varepsilon^{-1}\left[f_{0,k}\beta_k^{(1)}\beta_k^{(1)} + f_{1,k}\gamma_k^{(1)}\gamma_k^{(1)} + f_{2,k}\delta_k^{(1)}\delta_k^{(1)}\right] \\
R_k^{(i_0,i_1)} &= \varepsilon^{-1}\left[f_{0,k}\beta_k^{(2)}\beta_k^{(1)} + f_{1,k}\gamma_k^{(2)}\gamma_k^{(1)} + f_{2,k}\delta_k^{(2)}\delta_k^{(1)}\right] \\
R_k^{(i_0,i_2)} &= \varepsilon^{-1}\left[f_{0,k}\beta_k^{(3)}\beta_k^{(1)} + f_{1,k}\gamma_k^{(3)}\gamma_k^{(1)} + f_{2,k}\delta_k^{(3)}\delta_k^{(1)}\right] \\
\\
R_k^{(i_1,i_0)} &= \varepsilon^{-1}\left[f_{0,k}\beta_k^{(1)}\beta_k^{(2)} + f_{1,k}\gamma_k^{(1)}\gamma_k^{(2)} + f_{2,k}\delta_k^{(1)}\delta_k^{(2)}\right] \\
R_k^{(i_1,i_1)} &= \varepsilon^{-1}\left[f_{0,k}\beta_k^{(2)}\beta_k^{(2)} + f_{1,k}\gamma_k^{(2)}\gamma_k^{(2)} + f_{2,k}\delta_k^{(2)}\delta_k^{(2)}\right] \\
R_k^{(i_1,i_2)} &= \varepsilon^{-1}\left[f_{0,k}\beta_k^{(3)}\beta_k^{(2)} + f_{1,k}\gamma_k^{(3)}\gamma_k^{(2)} + f_{2,k}\delta_k^{(3)}\delta_k^{(2)}\right] \\
\\
R_k^{(i_2,i_0)} &= \varepsilon^{-1}\left[f_{0,k}\beta_k^{(1)}\beta_k^{(3)} + f_{1,k}\gamma_k^{(1)}\gamma_k^{(3)} + f_{2,k}\delta_k^{(1)}\delta_k^{(3)}\right] \\
R_k^{(i_2,i_1)} &= \varepsilon^{-1}\left[f_{0,k}\beta_k^{(2)}\beta_k^{(3)} + f_{1,k}\gamma_k^{(2)}\gamma_k^{(3)} + f_{2,k}\delta_k^{(2)}\delta_k^{(3)}\right] \\
R_k^{(i_2,i_2)} &= \varepsilon^{-1}\left[f_{0,k}\beta_k^{(3)}\beta_k^{(3)} + f_{1,k}\gamma_k^{(3)}\gamma_k^{(3)} + f_{2,k}\delta_k^{(3)}\delta_k^{(3)}\right]
\end{aligned}\right\} \tag{36}$$

Finally, recalling (23), the contribution to the global element stiffness matrix from $\mathcal{T}$ is then

$$\mathbf{A}\big|_{\mathcal{T}} = \mathbf{K} + \sum_{k=1}^{m}\mathbf{R}_k \tag{37}$$

2. Right-side Vector

The final piece to the puzzle is the contribution to the right-side vector $\mathbf{F}\big|_{\mathcal{T}}$; for this, we need only compute the perturbation $\mathcal{M}\mathbf{B}$, where

$$\mathbf{B} \triangleq \Big[\;\; f_{0,k}\hat{\mathbf{s}}(\mathbf{p}_1,\ldots,\mathbf{p}_m)^T \quad \mathbf{0} \quad f_{1,k}\hat{\mathbf{s}}(\mathbf{q}_1,\ldots,\mathbf{q}_m)^T \quad \mathbf{0} \quad f_{2,k}\hat{\mathbf{s}}(\mathbf{r}_1,\ldots,\mathbf{r}_m)^T \quad \mathbf{0} \;\;\Big]^T \tag{38}$$

Each $\mathbf{0}$ is a $N-m$ row vector of zeros. The block $\hat{\mathbf{s}}(\mathbf{d}_1,\ldots,\mathbf{d}_m)\in\mathbb{R}^m$ is defined as

$$\hat{\mathbf{s}}(\mathbf{d}_1,\ldots,\mathbf{d}_m)\triangleq\begin{bmatrix}\hat{s}(\mathbf{x}_1;\mathbf{d}_1)\\ \vdots\\ \hat{s}(\mathbf{x}_m;\mathbf{d}_m)\end{bmatrix}$$

The inclusion of a direction $\mathbf{d}_i$ indicates that the recession value $\hat{s}(\mathbf{x}_i)$ is in the direction $\mathbf{d}_i$. We denote it $\hat{s}$ instead of the usual $\dot{s}$ because this value may be nonphysical and encompasses all contributions to motion. For example, the recession of a quadrature point is $\dot{s}\Delta t$; $\hat{s}$ would include this factor of Δt.

The multiplication $\mathbf{E}_{\text{mod}}^{(3\text{Dc})}\triangleq\mathcal{M}\mathbf{B}$ has the same block structure as (28) and (38); the result is

$$\mathbf{E}_{\text{mod}}^{(3\text{Dc})}=\begin{bmatrix}\mathbf{E}_1^T & \mathbf{0} & \mathbf{E}_2^T & \mathbf{0} & \mathbf{E}_3^T & \mathbf{0}\end{bmatrix}^T \tag{39}$$

where the vectors $\mathbf{E}_1,\mathbf{E}_2,\mathbf{E}_3$ have components

$$\left.\begin{aligned}E_1^{(j)}&\triangleq f_{0,k}\hat{s}(\mathbf{x}_i;\mathbf{p}_j)p_j^{(0)}+f_{1,k}\hat{s}(\mathbf{x}_i;\mathbf{q}_j)q_j^{(0)}+f_{2,k}\hat{s}(\mathbf{x}_i;\mathbf{r}_j)r_j^{(0)}\\ E_2^{(j)}&\triangleq f_{0,k}\hat{s}(\mathbf{x}_i;\mathbf{p}_j)p_j^{(1)}+f_{1,k}\hat{s}(\mathbf{x}_i;\mathbf{q}_j)q_j^{(1)}+f_{2,k}\hat{s}(\mathbf{x}_i;\mathbf{r}_j)r_j^{(1)}\\ E_3^{(j)}&\triangleq f_{0,k}\hat{s}(\mathbf{x}_i;\mathbf{p}_j)p_j^{(2)}+f_{1,k}\hat{s}(\mathbf{x}_i;\mathbf{q}_j)q_j^{(2)}+f_{2,k}\hat{s}(\mathbf{x}_i;\mathbf{r}_j)r_j^{(2)}\end{aligned}\right\} \tag{40}$$

3. Summary

For node k within an element, suppose we have determined the directions of motion $\mathbf{v}_{k_1}$, $\mathbf{v}_{k_2}$, and $\mathbf{v}_{k_3}$ with known recession values $\hat{s}_1$, $\hat{s}_2$, and $\hat{s}_3$, respectively. These values and directions may be nonphysical and may not all be enforced. The procedure to compute the needed corrections is as follows:

1. Arrange the directions into $\mathbf{p}_k$, $\mathbf{q}_k$, and $\mathbf{r}_k$ via the Rearrangement Lemma.
2. Mark which of these directions are to be enforced via $f_{0,k}$, $f_{1,k}$, and $f_{2,k}$.
3. For each node on the boundary $\mathbf{x}_k$, $1\le k\le m$, compute the nonzero entries to the correction matrix $\mathbf{R}_k$ via (36). Add these values into the local element stiffness matrix $\mathbf{A}\big|_{\mathcal{T}}$.
4. Compute the correction (39) to the right-side vector. The entries of this addition are given in (40).

C. Quantifying the Motion

Section B assumes that all the directions $\mathbf{p}_k$, $\mathbf{q}_k$, and $\mathbf{r}_k$ and the corresponding $\hat{s}$ values are known *a priori*. `CHAR` specifies the values of $\dot{s}$ only at the face quadrature points. Constructing the direction and recession pairs $\big(\mathbf{p}_k,\hat{s}(\mathbf{x}_k;\mathbf{p}_k)\big)$, $\big(\mathbf{q}_k,\hat{s}(\mathbf{x}_k;\mathbf{q}_k)\big)$, and $\big(\mathbf{r}_k,\hat{s}(\mathbf{x}_k;\mathbf{r}_k)\big)$ at each node $\mathbf{x}_k$ from $\dot{s}$ and $\boldsymbol{\nu}$ at quadrature points is non-trivial and is the focus of this section. To illustrate the challenges we face, consider Figure 7 below.

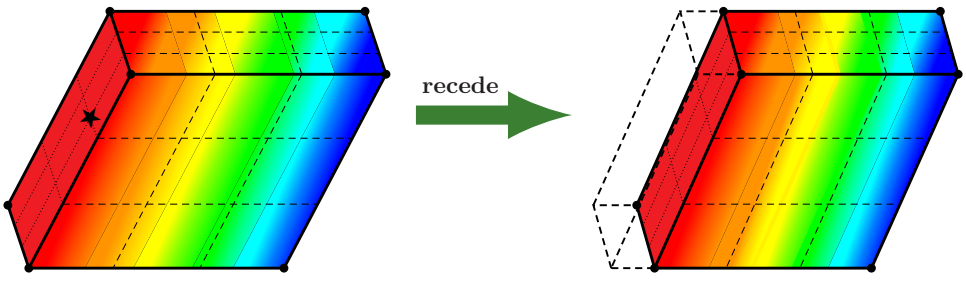


Fig. 7: Physically Correct Recession. If the temperature over the parallelogram's surface (red) is uniform, the surface should recede in a direction parallel to its current plane of existence (right).

Figure 7 depicts a parallelepiped with a single receding face (colored red); the faces adjacent to it are sliding faces and the remaining one (parallelogram in the back) is fixed. On physical grounds, we expect the parallelogram face to melt away and the element to kept its general shape through recession.

Figure 8 above highlights the difficulty in choosing directions of recession that lead to the physically correct result. At the edge corners (in blue), we require that the node remain on the sliding face; this keeps the exoskeleton formed by the four corner nodes and preserves the parallelepiped's general shape.

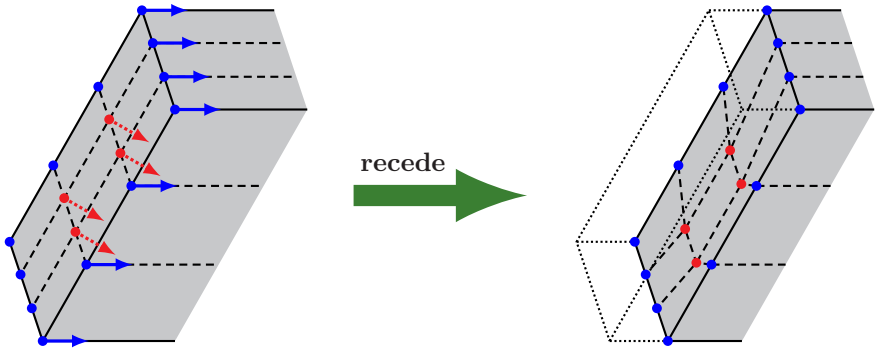


Fig. 8: When the "Right" Direction Goes Wrong. Edge corners (blue nodes) should move in the face in the sliding side set so that the sliding condition trumps the receding one. At the red interior nodes, the "natural" direction of motion is the normal vector. The result of this motion is on the left.

At the interior nodes (in red), we have no information of the existence of a sliding face because all analysis is done on a per-element basis. Furthermore, even if we did know of the sliding side sets, there is one in each cardinal direction–which is the "right one"? As such, it seems that the natural direction for interior nodes is the normal vector. However, the right side of the figure reveals that such motion causes mesh quality to deteriorate. Continued motion along the normal sets up a future unavoidable collision with an edge.

Complicating matters is that each interior node is contained in multiple elements. Figure 8 depicts a simple example in which each face in the receding side set has the same normal. If the receding set is a surface and not a mere plane, defining the normal at nodes is difficult—and defining $\hat{s}$ is harder yet.

1. Pure Corner Nodes

Perhaps contrary to expectations, the simplest case is the pure corner. A typical situation is depicted below:

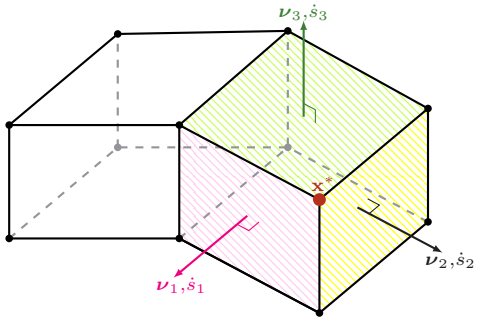


Fig. 9: Typical Pure Corner Case. The three different colors indicate the three side sets that contain the pure corner $\mathbf{x}^*$. The normal vectors for each side are indicated.

If any of the three side sets is of motion type "stationary," the node should remain fixed in place. We set $\mathbf{p} = \mathbf{e}_1$, $\mathbf{q} = \mathbf{e}_2$, $\mathbf{r} = \mathbf{e}_3$ and $\hat{s}(\mathbf{x}^*; \cdot) \equiv 0$ and move on. We therefore focus on combinations of sliding and

receding side sets. Unlike the other node types we shall explore later, there are no subcases for each possible combination of sliding and receding side sets.

It is tempting to choose the face normals as directions of motion (as in Figure 9). However, $\dot{s}$ is known only at quadrature points; choosing the value of $\dot{s}$ at the closest quadrature point is physically incorrect and likely to lead to poor mesh quality and skewed geometry. Instead, the solution is, in essence, to move entire side sets: we collect the face quadrature points from each face containing $\mathbf{x}^*$ in the side set, move each according to its known values of $\boldsymbol{\nu}$ and $\dot{s}$, and then compute the best-fit plane through the resultant points. We then find the intersection of these three planes, guaranteed to be unique except in degenerate cases.

The process is prescribed precisely in Algorithm 2 below:

Algorithm 2 (Pure Corner Motion).

1. Choose a side set S_j which contains $\mathbf{x}^*$. Let $\mathcal{F}_1, \ldots, \mathcal{F}_{m_j} \in S_j$ be the faces containing $\mathbf{x}^*$.
2. Let $\mathbf{x}_{k,1}, \ldots, \mathbf{x}_{k,q_k} \in \mathcal{F}_k$ be the quadrature points of the k^{th} face. Let $\boldsymbol{\nu}_{k,i}$ and $\dot{s}_{k,i}$ be the normal and recession value, respectively, specified at the quadrature point $\mathbf{x}_{k,i}$.
3. Compute the centroid of the collection $\bigcup_{k=1}^{m_j} \left\{\mathbf{x}_{k,i} + (\dot{s}_{k,i}\Delta t)\boldsymbol{\nu}_{k,i}\right\}_{i=1}^{q_k}$:
$$\mathbf{c}_j \triangleq \frac{1}{m_j}\sum_{k=1}^{m_j}\frac{1}{q_k}\sum_{i=1}^{q_k}\left[\mathbf{x}_{k,i} + (\dot{s}_{k,i}\Delta t)\boldsymbol{\nu}_{k,i}\right]$$
4. For each $k = 1, \ldots, m_j$, form the $3 \times q_k$ matrix
$$\mathbf{A}_k \triangleq \left[\begin{array}{c|c|c} \mathbf{x}_{k,1} + (\dot{s}_{k,1}\Delta t)\boldsymbol{\nu}_{k,1} - \mathbf{c}_j & \cdots & \mathbf{x}_{k,q_k} + (\dot{s}_{k,q_k}\Delta t)\boldsymbol{\nu}_{k,q_k} - \mathbf{c}_j \end{array}\right]$$
and glue all of these together to form the $3 \times Q_j$ matrix, where $Q_j \triangleq \prod_{k=1}^{m_j} q_k$
$$\mathbf{A} = \left[\begin{array}{ccc} \mathbf{A}_1 & \cdots & \mathbf{A}_{m_j} \end{array}\right]$$
5. Compute the singular value decomposition of $\mathbf{A}$:
$$\mathbf{A} = \mathbf{U}\boldsymbol{\Sigma}\mathbf{V}^T$$
where $\mathbf{U} \in \mathbb{R}^{3\times 3}$, $\boldsymbol{\Sigma} \in \mathbb{R}^{3\times Q_j}$, and $\mathbf{V} \in \mathbb{R}^{Q_j \times Q_j}$. Let $\mathbf{d}_1 \triangleq \mathbf{u}_3$, the 3^{rd} column of $\mathbf{U}$.
6. Repeat Steps 1-5 for the remaining two side sets to obtain $\mathbf{c}_2$, $\mathbf{d}_2$, $\mathbf{c}_3$, and $\mathbf{d}_3$.
7. Solve the linear system
$$\left[\begin{array}{c} \mathbf{d}_1^T \\ \mathbf{d}_2^T \\ \mathbf{d}_3^T \end{array}\right]\mathbf{x}_{\text{new}} = \begin{bmatrix}\langle \mathbf{d}_1, \mathbf{c}_1\rangle \\ \langle \mathbf{d}_2, \mathbf{c}_2\rangle \\ \langle \mathbf{d}_3, \mathbf{c}_3\rangle\end{bmatrix} \tag{41}$$
8. Set $\hat{\mathbf{a}} \triangleq \mathbf{x}^*_{\text{new}} - \mathbf{x}^*$ and $\hat{s} \triangleq |\hat{\mathbf{a}}|$. Compute the singular value decomposition
$$\hat{\mathbf{a}} = \hat{\mathbf{U}}\hat{\boldsymbol{\Sigma}}\hat{\mathbf{V}}^T$$
9. Set $\mathbf{w}_1 \triangleq (\text{sgn}\langle \hat{\mathbf{a}}, \hat{\mathbf{u}}_1\rangle)\,\hat{\mathbf{u}}_1$, $\mathbf{w}_2 \triangleq \hat{\mathbf{u}}_2$, $\mathbf{w}_3 \triangleq \hat{\mathbf{u}}_3$, where $\hat{\mathbf{u}}_i$ is the i^{th} column of $\hat{\mathbf{U}}$.
10. Follow Section B with the direction and recession pairs $(\mathbf{w}_1, \hat{s})$, $(\mathbf{w}_2, 0)$, $(\mathbf{w}_3, 0)$, all enforced.

Steps 1 and 2 are self-explanatory: collect all faces containing $\mathbf{x}^*$ along with the quadrature points and recession values and normals at those points. In steps 3 and 4, we work with the "new quadrature" points $\tilde{\mathbf{x}}_{k,i} \triangleq \mathbf{x}_{k,i} + (\dot{s}_{k,i}\Delta t)\boldsymbol{\nu}_{k,i}$. Because $\dot{s}$ is specified at each quadature point, we can compute with absolute certainty the new locations of only quadrature points after recession.

From the collection of $\tilde{\mathbf{x}}_{k,i}$ from the entire side set (containing $\mathbf{x}^*$), we must extract a single plane. For all but sliding side sets or single face receding sets and those with uniform temperature (so that $\dot{s} \equiv$ constant across the side set), the set of $\tilde{\mathbf{x}}_{k,i}$ are unlikely to be coplanar. To determine the post-recession face, we must obtain a best-fit plane through these points. As in 2D, Lemma 1 and Lemma 2 aid us in our question. As we did before, we call upon the singular value decomposition to obtain this best-fit plane.

Each plane generated in Step 5 is described by the equation $\langle \mathbf{x} - \mathbf{c}_i, \mathbf{d}_i \rangle = 0$. Writing out these three dot products and casting into matrix form leads to the 41, the solvability of which is aided by the following:

Lemma 5.

The normal vectors $\boldsymbol{\nu}_1$, $\boldsymbol{\nu}_2$, and $\boldsymbol{\nu}_3$ of adjacent faces of a tetrahedron span $\mathbb{R}^3$.

Proof.

Let $\mathbf{a}$, $\mathbf{b}$, and $\mathbf{c}$ be the edge vectors at the vertex formed by the intersection of the faces having normals $\boldsymbol{\nu}_1$, $\boldsymbol{\nu}_2$, and $\boldsymbol{\nu}_3$, arranged such that $\boldsymbol{\nu}_1 = \mathbf{a} \times \mathbf{b}$, $\boldsymbol{\nu}_2 = \mathbf{b} \times \mathbf{c}$, and $\boldsymbol{\nu}_3 = \mathbf{a} \times \mathbf{c}$. Because all vectors lie in $\mathbb{R}^3$, we have that

$$\det \left[\begin{array}{c|c|c} \boldsymbol{\nu}_1 & \boldsymbol{\nu}_2 & \boldsymbol{\nu}_3 \end{array} \right] = \boldsymbol{\nu}_1 \cdot (\boldsymbol{\nu}_2 \times \boldsymbol{\nu}_3) \tag{42}$$

Expanding $\boldsymbol{\nu}_2$ and $\boldsymbol{\nu}_3$ in terms of $\mathbf{a}$, $\mathbf{b}$, and $\mathbf{c}$, and applying the identity for the the vector triple product $\mathbf{a}' \times (\mathbf{b}' \times \mathbf{c}') = \mathbf{b}'(\mathbf{a}' \cdot \mathbf{c}') - \mathbf{c}'(\mathbf{a}' \cdot \mathbf{b}')$, we have

$$\boldsymbol{\nu}_2 \times \boldsymbol{\nu}_3 = \big(\mathbf{a} \cdot (\mathbf{b} \times \mathbf{c})\big)\mathbf{c}$$

Hence,

$$\boldsymbol{\nu}_1 \cdot (\boldsymbol{\nu}_2 \times \boldsymbol{\nu}_3) = \big[(\mathbf{a} \times \mathbf{b}) \cdot \mathbf{c}\big]\big[(\mathbf{b} \times \mathbf{c}) \cdot \mathbf{a}\big] = \big[(\mathbf{a} \times \mathbf{b}) \cdot \mathbf{c}\big]^2$$

by the nature of the scalar triple product. The volume of the tetrahedron is $\frac{1}{6}\,|(\mathbf{a} \times \mathbf{b}) \cdot \mathbf{c}|$, so if the lefthand side is zero, then the tetrahedron is vacuous—a contradiction. Accordingly, it follows that $\boldsymbol{\nu}_1$, $\boldsymbol{\nu}_2$, and $\boldsymbol{\nu}_3$ are linearly independent and hence span $\mathbb{R}^3$. ■

For hexahedra and pyramids, we can construct a tetrahedron contained in the original polygon that overlaps with the original adjacent faces in question, thus extending the above result to these polygons. The lemma itself, of course, is not directly applicable to (41). The vectors that determine (41) are normal vectors from planes not known *a priori* to form a polygon. However, since the new plane is a best-fit of perturbed points from the original and the determinant is a continuous map that measures volume, we can reasonably expect (42) to remain nonzero. The contrary requires motion of one face so dramatic that it becomes parallel to another, highly unlikely unless this were nearly true originally (e.g., a shield-like corner).

The intersection point determined in Step 7 is the location of the new node. Step 8 computes an artificial direction of motion $\hat{\mathbf{a}}$, directed toward the new location from the old. The recession value $\hat{s}$ is the length of this vector; SVD provides both a normalized version of $\hat{\mathbf{a}}$ and a basis for $\{\hat{\mathbf{a}}\}^{\perp}$. We must adjust for sign, since $\hat{\mathbf{u}}_1 = \pm\,|\hat{\mathbf{a}}|^{-1}\,\hat{\mathbf{a}}$. It is crucial that the direction be pointed toward $\mathbf{x}_{\text{new}}$ to be consistent with $\dot{s}$.

2. Edge Corner Nodes

Recall that an edge corner node is one that is contained in two different side sets. Geometry can differ wildly among instances of edge corners (two simple examples are depicted in Figure 6). Since the set of potential

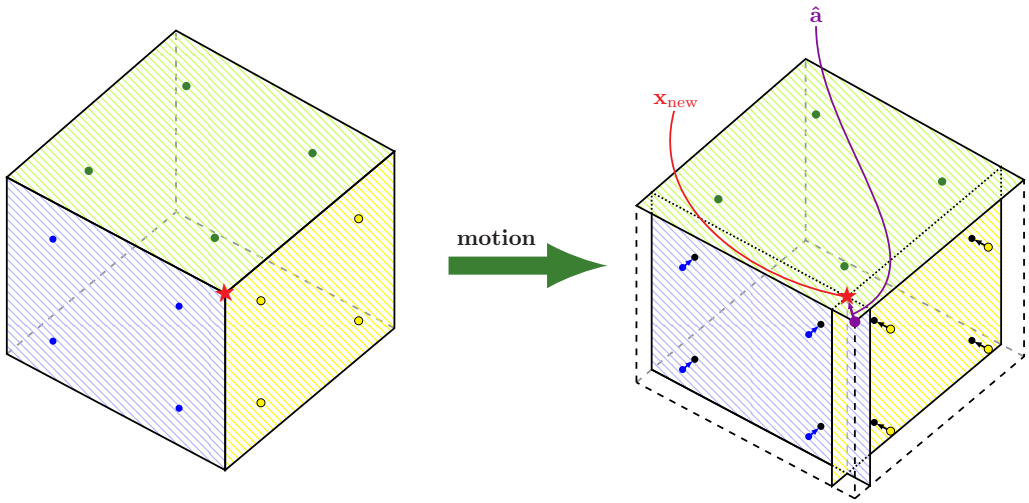


Fig. 10: Visual Depiction of Algorithm 2. This figure depicts a simple example of Algorithm 2 in action. The geometry before motion is shown on the left. The blue (left) and yellow (right) faces are receding, with uniform recession at each face's quadrature points but different between the two faces. The green (top) face is sliding. The left diagram shows the motion of the node: it moves from the purple circle to the red star $\mathbf{x}_{\text{new}}$. The direction $\hat{\mathbf{a}}$ is indicated. The length of this vector gives $\hat{s}$.

configurations is seemingly unbounded, any attempt to account for special geometries amounts to a feral *anatidae* pursuit (or more colloquially, a "wild goose chase"). Instead, we design for "typical" cases such as those in the figure and place some burden on the user to have designed a reasonable mesh.

We identify three possible configurations: when both side sets are receding, when one is sliding and the other receding, and when both are sliding. When one set is stationary, we set all motion to zero as we did before.

3. Edge Corner: Receder/Receder

We begin with the most complex of the cases: double receders. This configuration causes the most issues for simple mesh motion schemes; success in handling this case is therefore a hallmark of the tangle-free mesh motion scheme. The method is ostensibly similar to pure corner case but necessarily differs in a few aspects due to the increased complexity of the edge corner.

Summary of method: Very similar in spirit to the pure corner case. We have two sides in motion. We compute the new faces by best-fit planes as we did in the pure corner case. To replace the missing third side, we use internal (non-boundary) faces that contain the node. The intersection with the new faces is computed for each internal face. We then take the new location of the node to be the centroid of all these intersection points.

4. Edge Corner Nodes: Receder/Slider

We begin by presenting the mathematical algorithm. Explanation of the various steps follows after.

Algorithm 3 (Edge Corner: Double Receder).

1. Let $\mathcal{F}_1, \ldots, \mathcal{F}_m$ be the boundary faces containing $\mathbf{x}^*$ in the receding side set; $\mathcal{F}'_1, \ldots, \mathcal{F}'_{m'}$, in the sliding side set.
2. *(Receding side set.)* Let $\mathbf{x}_{k,1}, \ldots, \mathbf{x}_{k,q_k} \in \mathcal{F}_k$ be the quadrature points of the k^{th} receding face. Let $\boldsymbol{\nu}_{k,i}$ and $\dot{s}_{k,i}$ be the normal and recession value, respectively, specified at $\mathbf{x}_{k,i}$.

(continued).

3. Compute the centroid of the collection $\bigcup_{k=1}^{m} \left\{ \mathbf{x}_{k,i} + (\dot{s}_{k,i}\Delta t)\boldsymbol{\nu}_{k,i} \right\}_{i=1}^{q_k}$:

$$\mathbf{c} \triangleq \frac{1}{m} \sum_{k=1}^{m} \frac{1}{q_k} \sum_{i=1}^{q_k} \left[\mathbf{x}_{k,i} + (\dot{s}_{k,i}\Delta t)\boldsymbol{\nu}_{k,i} \right]$$

4. For each $k = 1, \ldots, m$, form the $3 \times q_k$ matrix

$$\mathbf{A}_k \triangleq \left[\begin{array}{c|c|c|c} \mathbf{x}_{k,1} + (\dot{s}_{k,1}\Delta t)\boldsymbol{\nu}_{k,1} - \mathbf{c} & \cdots & \mathbf{x}_{k,q_k} + (\dot{s}_{k,q_k}\Delta t)\boldsymbol{\nu}_{k,q_k} - \mathbf{c} \end{array} \right]$$

and glue all of these together to form the $3 \times Q$ matrix, where $Q \triangleq \prod_{k=1}^{m} q_k$,

$$\mathbf{A} = \left[\begin{array}{ccc} \mathbf{A}_1 & \cdots & \mathbf{A}_m \end{array} \right]$$

5. Compute the singular value decomposition of $\mathbf{A}$:

$$\mathbf{A} = \mathbf{U}\boldsymbol{\Sigma}\mathbf{V}^T$$

where $\mathbf{U} \in \mathbb{R}^{3\times 3}$, $\boldsymbol{\Sigma} \in \mathbb{R}^{3\times Q}$, and $\mathbf{V} \in \mathbb{R}^{Q\times Q}$. Let $\mathcal{P}$ be the plane with normal vector $\boldsymbol{\nu} \triangleq \mathbf{u}_3$, the 3^{rd} column of $\mathbf{U}$, passing through the point $\mathbf{c}$.

6. *(Sliding side set.)* Let $\mathbf{x}'_{k,1}, \ldots, \mathbf{x}'_{k,q'_k} \in \mathcal{F}'_k$ be the quadrature points of the k^{th} sliding face.

7. Compute the centroid of the collection $\bigcup_{k=1}^{m'} \left\{ \mathbf{x}'_{k,i} \right\}_{i=1}^{q'_k}$:

$$\mathbf{c}' \triangleq \frac{1}{m'} \sum_{k=1}^{m'} \frac{1}{q'_k} \sum_{i=1}^{q'_k} \mathbf{x}'_{k,i}$$

8. Form the $3 \times Q'$ matrix, where $Q' \triangleq \prod_{k=1}^{m'} q'_k$,

$$\mathbf{A}' \triangleq \left[\begin{array}{c|c|c|c|c|c|c} \mathbf{x}'_{1,1} - \mathbf{c}' & \cdots & \mathbf{x}'_{1,q_1} - \mathbf{c}' & \cdots & \mathbf{x}'_{m',q'_{m'}} - \mathbf{c}' & \cdots & \mathbf{x}'_{m',q'_{m'}} - \mathbf{c}' \end{array} \right]$$

9. Compute the singular value decomposition of $\mathbf{A}'$:

$$\mathbf{A} = \mathbf{U}'\boldsymbol{\Sigma}'{\mathbf{V}'}^T$$

where $\mathbf{U}' \in \mathbb{R}^{3\times 3}$, $\boldsymbol{\Sigma}' \in \mathbb{R}^{3\times Q'}$, and $\mathbf{V}' \in \mathbb{R}^{Q'\times Q'}$. Let $\mathcal{P}'$ be the plane with normal vector $\boldsymbol{\nu}' \triangleq \mathbf{u}'_3$, the 3^{rd} column of $\mathbf{U}'$, passing through the point $\mathbf{c}'$.

10. Let $\mathcal{F}''_1, \ldots, \mathcal{F}''_b$ be the non-boundary (internal) faces containing $\mathbf{x}^*$. Let $\boldsymbol{\nu}''_k$ and $\mathbf{c}''_k$ be the normal and centroid, respectively, of $\mathcal{F}''_k$.

11. For each $k = 1, \ldots, b$, compute the intersection of $\mathcal{P}$, $\mathcal{P}'$, and $\mathcal{F}''_k$ by solving

$$\left[\begin{array}{c|c|c} \boldsymbol{\nu} & \boldsymbol{\nu}' & \boldsymbol{\nu}''_k \end{array} \right]^T \mathbf{x}^{(k)}_{\text{new}} = \begin{bmatrix} \langle \boldsymbol{\nu}, \mathbf{c} \rangle \\ \langle \boldsymbol{\nu}', \mathbf{c}' \rangle \\ \langle \boldsymbol{\nu}''_k, \mathbf{c}''_k \rangle \end{bmatrix} \tag{43}$$

12. Compute the centroid of $\left\{ \mathbf{x}^{(k)}_{\text{new}} \right\}_{k=1}^{b}$. Call this point $\mathbf{x}^*_{\text{new}}$.

(continued).

13. Set $\hat{\mathbf{a}} \triangleq \mathbf{x}^*_{\text{new}} - \mathbf{x}^*$ and $\hat{s} \triangleq |\hat{\mathbf{a}}|$. Compute the singular value decomposition
$$\hat{\mathbf{a}} = \hat{\mathbf{U}}\hat{\mathbf{\Sigma}}\hat{\mathbf{V}}^T$$
14. Set $\mathbf{w}_1 \triangleq (\text{sgn}\langle\hat{\mathbf{a}}, \hat{\mathbf{u}}_1\rangle)\,\hat{\mathbf{u}}_1$, $\mathbf{w}_2 \triangleq \hat{\mathbf{u}}_2$, $\mathbf{w}_3 \triangleq \hat{\mathbf{u}}_3$, where $\hat{\mathbf{u}}_i$ is the i^{th} column of $\hat{\mathbf{U}}$.
15. Follow Section B with the direction and recession pairs $(\mathbf{w}_1, \hat{s})$, $(\mathbf{w}_2, 0)$, $(\mathbf{w}_3, 0)$, all enforced.

Steps 1-5 mirror those for the pure corner case exactly; steps 6-9 repeat the procedure for the sliding side set. As before, we compute a single best fit plane for all containing faces in each side set. In the pure corner case, Here, though, the maneuver seems to be more precarious.
[Some results and theorems that prove what we did isn't so bad]

5. Edge Corner Nodes: Slider/Slider

This section will look similar to the one on Pure Corners once it is written.

Summary of method: The two edges, $\mathcal{E}_1$ and $\mathcal{E}_2$, containing the node that live between the sidesets are identified (there are always only two). We move along $\mathcal{E}_1$ to the next node, identify the other edge between the sidesets containing that node, and move along it. We repeat this process until we come to another sideset or loop back to where we started. If the encountered sideset is a receder, then we select $\mathcal{E}_2$ as the direction of motion; otherwise, we use $\mathcal{E}_1$. We then compute the orthogonal complement to the selected direction with the SVD and enforce zero-motion conditions in these two orthogonal directions.

6. Interior Nodes

Recall that an interior node is defined by a single common side set shared among all faces containing $\mathbf{x}^*$. Our analysis requires us to distinguish three cases by the nature of motion of this side set.

The easiest case is when the motion of the side set is "stationary." While this case can be covered by one of the two coming cases, the amount of detection and computation done in those cases ultimately goes to waste for a stationary side set. It is therefore more efficient to detect this case and handle it immediately: set $\mathbf{p} = \mathbf{e}_1$, $\mathbf{q} = \mathbf{e}_2$, $\mathbf{r} = \mathbf{e}_3$ and $\hat{s}(\mathbf{x}^*; \cdot) \equiv 0$ and enforce these boundary conditions.

7. Interior Nodes: Receders

It is most clear to present the algorithm first and then explain its components afterwards.

Algorithm 4 (Interior Receder).

1. Let $\mathcal{F}_1, \ldots, \mathcal{F}_m$ be the boundary faces containing $\mathbf{x}^*$.
2. Let $\mathbf{x}_{k,1}, \ldots, \mathbf{x}_{k,q_k} \in \mathcal{F}_k$ be the quadrature points of the k^{th} face. Let $\boldsymbol{\nu}_{k,i}$ and $\dot{s}_{k,i}$ be the normal and recession value, respectively, specified at the quadrature point $\mathbf{x}_{k,i}$.
3. Compute the centroid of the points $\left\{\mathbf{x}_{k,i} + (\dot{s}_{k,i}\Delta t)\boldsymbol{\nu}_{k,i}\right\}_{i=1}^{q_k}$:
$$\mathbf{c}_k \triangleq \frac{1}{q_k}\sum_{i=1}^{q_k}\left[\mathbf{x}_{k,i} + (\dot{s}_{k,i}\Delta t)\boldsymbol{\nu}_{k,i}\right]$$

(continued).

4. Form the $3 \times q_k$ matrix

$$\mathbf{A}_k \triangleq \left[\begin{array}{c|c|c} \mathbf{x}_{k,1} + (\dot{s}_{k,1}\Delta t)\boldsymbol{\nu}_{k,1} - \mathbf{c}_k & \cdots & \mathbf{x}_{k,q_k} + (\dot{s}_{k,q_k}\Delta t)\boldsymbol{\nu}_{k,q_k} - \mathbf{c}_k \end{array} \right]$$

5. Compute the singular value decomposition of $\mathbf{A}_k$:

$$\mathbf{A}_k = \mathbf{U}_k \boldsymbol{\Sigma}_k \mathbf{V}_k^T$$

where $\mathbf{U}_k \in \mathbb{R}^{3\times 3}$, $\boldsymbol{\Sigma}_k \in \mathbb{R}^{3\times q_k}$, and $\mathbf{V}_k \in \mathbb{R}^{q_k \times q_k}$. Let $\hat{\boldsymbol{\nu}}_k \triangleq \mathbf{u}_3$, the 3rd column of $\mathbf{U}_k$.

6. Repeat Steps 2-5 for each $k \in \{1, \ldots, m\}$ to obtain $\mathbf{c}_k$ and $\hat{\boldsymbol{\nu}}_k$ for each face $\mathcal{F}_k$.

7. Locate each edge containing $\mathbf{x}^*$ that is not contained on $\mathcal{F}_k$. This will yield the collection $\{\bar{\mathbf{e}}_{k,i}\}_{i=1}^{n_k}$, each of which should be first normalized.

8. For each $i = 1, \ldots, n_k$, compute the intersection of the line $\ell_i(t) = \mathbf{x}^* + t\,\bar{\mathbf{e}}_{k,i}$ with the plane with normal $\hat{\boldsymbol{\nu}}_k$ passing through the point $\mathbf{c}_k$ (k^{th} best fit plane). First, compute

$$\alpha_i \triangleq \frac{\langle \mathbf{c}_k - \mathbf{x}^*, \hat{\boldsymbol{\nu}}_k \rangle}{\langle \bar{\mathbf{e}}_{k,i}, \hat{\boldsymbol{\nu}}_k \rangle}$$

The intersection point is then given by

$$\mathbf{y}_{k,i} \triangleq \mathbf{x}^* + \alpha_i \bar{\mathbf{e}}_{k,i}$$

9. Repeat Steps 7 and 8 for each $k \in \{1, \ldots, m\}$.

10. Compute the centroid of the collection $\bigcup_{k=1}^{m} \{\mathbf{y}_{k,i}\}_{i=1}^{n_k}$:

$$\mathbf{x}^*_{\text{new}} \triangleq \left(\sum_{k=1}^{m} n_k \right)^{-1} \sum_{k=1}^{m} \sum_{i=1}^{n_k} \mathbf{y}_{k,i}$$

11. Set $\hat{\mathbf{a}} \triangleq \mathbf{x}^*_{\text{new}} - \mathbf{x}^*$ and $\hat{s} \triangleq |\hat{\mathbf{a}}|$. Compute the singular value decomposition

$$\hat{\mathbf{a}} = \hat{\mathbf{U}} \hat{\boldsymbol{\Sigma}} \hat{\mathbf{V}}^T$$

12. Set $\mathbf{w}_1 \triangleq (\text{sgn}\langle \hat{\mathbf{a}}, \hat{\mathbf{u}}_1 \rangle)\, \hat{\mathbf{u}}_1$, $\mathbf{w}_2 \triangleq \hat{\mathbf{u}}_2$, $\mathbf{w}_3 \triangleq \hat{\mathbf{u}}_3$, where $\hat{\mathbf{u}}_i$ is the i^{th} column of $\hat{\mathbf{U}}$.

13. Follow Section B with the direction and recession pairs $(\mathbf{w}_1, \hat{s})$, $(\mathbf{w}_2, 0)$, $(\mathbf{w}_3, 0)$.

8. Interior Nodes: Sliders

This section will look similar to the preceding one once it's written.

Summary of the method: use radial basis functions to obtain an estimate of the surface normal vector. By perturbing along the known normal vectors at the quadrature points, we can construct a function whose gradient gives the normal at any point.

D. Example Problems

This section will contain several examples of the three-dimensional scheme in action.

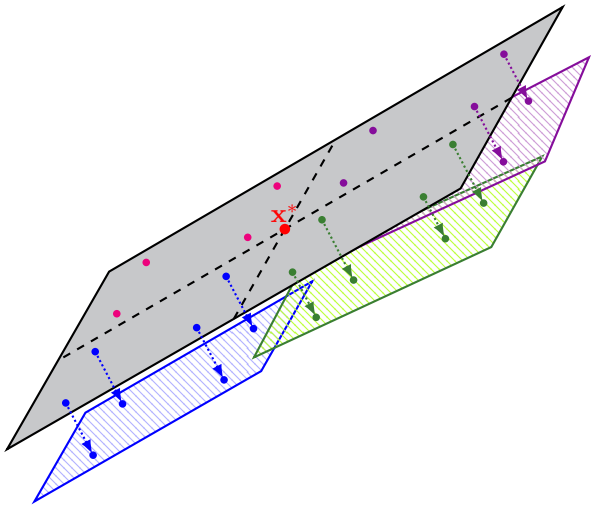


Fig. 11: Algorithm 4 in Action, Part I. This figure shows Steps 1-6 of Algorithm 4 in action. The quadrature points on each face move their specified $\dot{s}$ units along the normal for that location. The best-fit plane for the new points is then computed (colored planes) by the SVD.

IV. Conclusion

Summary and wrap-up.

Appendix

Some proofs of results I have used; they are not new results but the insight in the proof may be useful to those who have not seen these concepts before but are interested.