

Eigen Space of Mesh Distortion Energy Hessian

YUFENG ZHU

Mesh distortion optimization is a popular research topic and has wide range of applications in computer graphics, including geometry modeling, variational shape interpolation, UV parameterization, elastoplastic simulation, etc. In recent years, many solvers have been proposed to solve this nonlinear optimization efficiently, among which projected Newton has been shown to have best convergence rate and work well in both 2D and 3D applications. Traditional Newton approach suffers from ill conditioning and indefiniteness of local energy approximation. A crucial step in projected Newton is to fix this issue by projecting energy Hessian onto symmetric positive definite (SPD) cone so as to guarantee the search direction always pointing to decrease the energy locally. Such step relies on time consuming Eigen decomposition of element Hessian, which has been addressed by several work before on how to obtain a conjugacy that is as diagonal as possible. In this report, we demonstrate an analytic form of Hessian eigen system for distortion energy defined using principal stretches, which is the most general representation. Compared with existing projected Newton diagonalization approaches, our formulation is more general as it doesn't require the energy to be representable by tensor invariants. In this report, we will only show the derivation for 3D and the extension to 2D case is straightforward.

ACM Reference Format:

Yufeng Zhu. 2021. Eigen Space of Mesh Distortion Energy Hessian. 1, 1 (March 2021), 9 pages. <https://doi.org/10.1145/nnnnnnn.nnnnnnn>

1 INTRODUCTION

There have already been many interesting mesh distortion optimization solvers so far [Bouaziz et al. 2014; Claici et al. 2017; Kovalsky et al. 2016; Liu et al. 2018, 2017; Peng et al. 2018; Rabinovich et al. 2017; Shtengel et al. 2017; Sin et al. 2011; Smith et al. 2018, 2019; Stomakhin et al. 2012; Teran et al. 2005; Xu et al. 2015; Zhu et al. 2018]. As this is just a technical report on hessian diagonalization, we will only include the references here instead of going to detailed discussion one by one. The most related previous work are [Smith et al. 2018, 2019; Stomakhin et al. 2012; Teran et al. 2005]. However, they either assume that the distortion energy can be represented using tensor invariants or just achieve a block diagonal structure. In this work, we will show the analytic eigen system of element hessian whose energy is defined using principal stretches, which is more general representation of mesh distortion. For any 3D mesh distortion energy defined using principal stretches, we prove that its diagonal form is composed of six scalars and one 3×3 block. Depending on the energy definition, the 3×3 block might also have an analytic diagonalization.

Author's address: Yufeng Zhu, mike323zyf@gmail.com.

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for components of this work owned by others than ACM must be honored. Abstracting with credit is permitted. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee. Request permissions from permissions@acm.org.

© 2021 Association for Computing Machinery.

XXXX-XXXX/2021/3-ART \$15.00

<https://doi.org/10.1145/nnnnnnn.nnnnnnn>

2 RELATED WORK

2.1 Distortion Energies

Mesh distortion optimization is largely characterized by distortion energy. Many fundamental physical and geometric modeling problems reduce to minimizing measures of distortion over meshes. A wide range of energies have been proposed to fulfill such tasks for various application purposes. Linear approaches have the benefit of good performance as the distortion energies adopted are usually modeled as quadratic and require only a single linear system factorization [Lipman et al. 2005; Weber et al. 2009, 2007; Zayer et al. 2005], but they also suffer from severe artifacts under large deformations. On the other hand, nonlinear measurement requires more sophisticated solvers but behaves well even for extreme deformations.

Diverse range of nonlinear energies have been proposed to minimize various mapping distortions in the field of geometry processing, generally focused on minimizing either measures of isometric [Aigerman et al. 2015; Chao et al. 2010; Liu et al. 2008; Smith and Schaefer 2015] or conformal [Ben-Chen et al. 2008; Desbrun et al. 2002; Hormann and Greiner 2002; Lévy et al. 2002; Mullen et al. 2008; Weber et al. 2012] distortion. As rigid as possible (ARAP), as similar as possible (ASAP) and as killing as possible (AKAP) are three alternative ways to minimize isometric/conformal distortions [Alexa et al. 2000; Igarashi et al. 2005; Solomon et al. 2011; Sorkine and Alexa 2007]. Other type energies like Dirichlet energy [Schüller et al. 2013], maximal stretch energy [Sorkine et al. 2002] and Green-Lagrange energy [Bonet and Wood 2008] are also popular choices for applications.

Distortion optimization is also applicable to physical based animation which typically minimizes hyperelastic potentials formed by integrating strain energy densities over the material domain to simulate elastic solids with large deformations. These material models date back to Mooney [Mooney 1940] and Rivlin [Rivlin 1948]. Their Mooney-Rivlin and Neo-Hookean materials, and many subsequent hyperelastic materials, e.g. St. Venant-Kirchoff, Ogden, Fung [Bonet and Burton 1998], are constructed from empirical observation and analysis of deforming real-world materials. Material properties in these models are specified according to experiment for scientific computing applications [Ogden 1972], or alternately are directly set by users in other cases [Xu et al. 2015], e.g., to meet artistic needs. Modified energy model [Stomakhin et al. 2012] has also been proposed in computer graphics to stabilize physical simulation.

There are various ways to parameterize the aforementioned distortion energies, including strain tensor invariants [Teran et al. 2005], stretch tensor invariants [Smith et al. 2019], principal stretches (singular values of mapping Jacobian) [Stomakhin et al. 2012; Xu et al. 2015], etc. Among these options, principal stretch is the most general parameterization approach as it not only covers both 2D and 3D cases but also plays a more fundamental role than the others in defining distortion measurement. Moreover, it has been shown that principal stretch based distortion energy is more user-friendly

and can be manipulated or modified for application purposes more easily and intuitively [Stomakhin et al. 2012; Xu et al. 2015].

2.2 First Order Methods

To obtain deformed mesh with optimal distortion measured by energies we just introduced, many solvers have been proposed and studied so far. The local-global method has been recognized as one of the most popular approaches and applied to many applications, including surface modeling [Sorkine and Alexa 2007], parameterization [Liu et al. 2008] and volumetric deformation [Bouaziz et al. 2014], etc. Not until recent years did researchers find out that such method is closely related to Sobolev gradient [Neuberger 2006, 2010]. Kovalsky et al. [2016] extended this method by introducing acceleration to speed up its convergence, while Liu et al. [2017] instead combined it with L-BFGS for the same goal. Other notable solver improvement includes iterative reweighting Laplacian method [Rabinovich et al. 2017] and killing vector field proximal algorithm [Claici et al. 2017]. Both methods improve solver convergence by using more effective quadratic approximation during each iteration but also require solving a new linear system at every step. To overcome such issue, alternative approaches, which focus on efficient way of constructing effective local approximation, have been proposed by Zhu et al. [2018] and Peng et al. [2018], both of which are variants of Broyden class method. All the listed techniques are regarded as first order methods as they rely on linear approximation of distortion energy. Other first order methods, like Gauss Newton [Eigensatz and Pauly 2009], block coordinate descent [Fu et al. 2015] and L-BFGS [Smith and Schaefer 2015], are also used in graphics applications, but they all share the poor convergence rate problem, which leads to the extensive study of efficient second order solvers.

2.3 Second Order Methods

Second order methods generally can achieve the most rapid convergence for convex energies but requires modification for nonconvex ones [Nocedal and Wright 2006] to ensure that the proxy is at least positive semi-definite (PSD). At each iterate, the distortion energy hessian is evaluated to form a proxy matrix and special care must be taken in order to handle its indefiniteness. Trust region could be a quick fix to this problem and actually has been adopted by several previous work [Chao et al. 2010; Schüller et al. 2013]. However, finding the proper window size could require much more extra computational cost without any simplification, like the Dogleg method. Another popular strategy is fixed Newton which attracts lots of attention from research community in recent years. Composite majorization, a tight convex majorizer, was recently proposed as an analytic PSD approximation of the hessian [Shtengel et al. 2017]. Its proxy is efficient to assemble, but is limited to 2D problems. More general is the projected Newton method that projects per-element Hessians to the PSD cone prior to assembly [Fu and Liu 2016; Stomakhin et al. 2012; Teran et al. 2005; Xu et al. 2015]. However, these approaches still depend on numerical constructions of projected hessian and do not yield closed-form expressions for the underlying hessian's eigen pairs. Chen and Weber [2017] developed analytic hessian projection in a reduced basis setting but only applies to 2D planar cases. Energy specific approaches [McAdams et al. 2011;

Smith et al. 2018] are able to reveal the analytic eigen system of certain distortion measurement, which are recently generalized by Smith et al. [2019] to cover more isotropic energies representable in stretch tensor invariants. Our work, as a further improvement over existing approaches, provides closed-form expressions for eigen values and eigen vectors of distortion energies parameterized by principal stretches, which is the most general and convenient way to define distortion measurement in both geometry and simulation areas.

3 BACKGROUND

Mesh distortion optimization is a computational approach to find mapping relationship between two domains, reference and deformed, in 2D or 3D. It has wide range of useful and essential applications in computer graphics field, for instance, planar animation, UV parameterization, geometry modeling, elastoplastic simulation, variational shape interpolation, etc. The goal is to preserve local metrics as much as possible with respect to constraints if any. Such problem starts from meshes, \mathbf{X} , that discretize the reference domain and looks for optimal deformed meshes, \mathbf{x} , satisfying given constraints. (See Figure 1 as an illustration example.) Optimality judgement depends on the local metrics to be preserved. One practical and

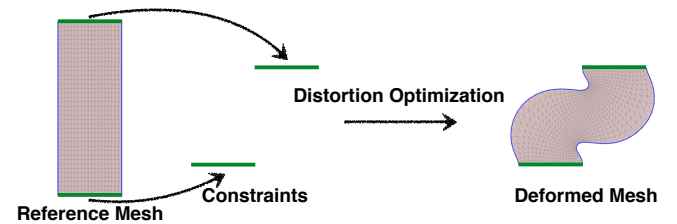


Fig. 1

popular direction to solve this problem is through formulating it as a constrained nonlinear local optimization problem,

$$\begin{aligned} \min_{\mathbf{x}} \quad & \mathcal{E}_{\mathbf{X}}(\mathbf{x}) \\ \text{s.t.} \quad & \mathbf{x} \in C, \end{aligned} \quad (1)$$

where C denotes the feasible set or region. Applying traditional constrained optimization techniques directly, like proximal gradient, barrier method, primal dual interior point method, etc, either suffers from slow convergence rate or stability issue caused by problem indefiniteness. In recent years, it attracts lots of attention and interests from researchers who proposed various efficient and stable solvers for this problem, including preconditioned gradient descent [Bouaziz et al. 2014; Claici et al. 2017; Kovalsky et al. 2016; Liu et al. 2018; Rabinovich et al. 2017], variants of quasi-Newton [Liu et al. 2017; Peng et al. 2018; Zhu et al. 2018] and fixed Newton methods [Shtengel et al. 2017; Smith et al. 2018, 2019; Stomakhin et al. 2012; Teran et al. 2005]. Among these approaches, fixed Newton methods demonstrate the most promising second order convergence rate property, especially when close to the local optimum. Like traditional Newton method, such approaches iteratively look for the solution where the mesh distortion energy gradient, $\nabla \mathcal{E}_{\mathbf{X}}(\mathbf{x})$, vanishes. During each iteration, a crucial step is to solve a linear system

for the solution update direction,

$$\mathcal{H}\mathbf{p} = -\nabla\mathcal{E}_{\mathbf{X}}, \quad (2)$$

where \mathcal{H} represents energy hessian and \mathbf{p} is the update direction. For monotonic energy descent approaches, \mathcal{H} is required to be symmetric positive definite (SPD) in order to guarantee converged iterations. However, for general nonlinear energy, like $\mathcal{E}_{\mathbf{X}}$, its hessian usually doesn't hold such property everywhere within the feasible region \mathcal{C} . It is well known that the indefiness or negative definess of \mathcal{H} will cause the Newton iteration diverge. In order to stabilize the solver, fixed Newton methods first project \mathcal{H} onto SPD cone and then use its SPD projection, \mathcal{H}^+ , to solve for update direction,

$$\mathcal{H}^+\mathbf{p} = -\nabla\mathcal{E}_{\mathbf{X}}. \quad (3)$$

Composite majorization[Shtengel et al. 2017] obtains the projection through convex-concave decomposition of $\mathcal{E}_{\mathbf{X}}$ and uses the convex part's hessian as \mathcal{H}^+ , but so far efficient decomposition method is limited to 2D setting only. Projected Newton methods adopt the more general eigen decomposition approach,

$$\mathcal{H} = \mathbf{K}^T \mathbf{\Lambda} \mathbf{K} \Rightarrow \mathcal{H}^+ = \mathbf{K}^T \mathbf{\Lambda}^+ \mathbf{K}, \quad (4)$$

by clamping all negative and nearly zero eigenvalues of $\mathbf{\Lambda}$ to a small positive threshold, where $\mathbf{\Lambda}^+$ is the clamped diagonal eigenvalue matrix. However, cost of eigen decomposition grows very quickly as the system size increases. To overcome this problem, researchers [Stomakhin et al. 2012; Teran et al. 2005] explored and found that they can first efficiently convert \mathcal{H} into a block diagonal form through congruent transformation and then only apply eigen decomposition to small sub-blocks, which is much faster than applying to \mathcal{H} directly. Recently, Smith et al. [2018; 2019] further improve this result by assuming the distortion energy, $\mathcal{E}_{\mathbf{X}}$, is representable using invariants of stretch tensor. In such cases, they show that at least $\frac{2}{3}$ portion in 3D and $\frac{1}{2}$ portion in 2D of eigenvalues have analytical expressions. Even though such assumption already covers many popular energy choices used in geometry and physical simulation problems, it still restricts its application to limited range of distortion energy considering energy that can be defined by more atomic representation, principal stretches. We are showing in this work that for more general energy defined using principal stretches, the same portion of eigenvalues can always be evaluated analytically. Whether analytic expressions exist for the left portion of eigenvalues depends on the energy definition in terms of principal stretches and can be told by studying a small matrix, 3×3 in 3D and 2×2 in 2D.

3.1 Problem Definition

For the ease of explanation, we assume the reference domain is discretized using P1 element for 2D and 3D problems and our derivation should be generalizable to other types of elements, like P2, Q1, S1, etc. Moreover, in this paper, we only discuss 3D case and it's straightforward to extend our analysis to 2D case. Our input would be a 3D reference domain discretized using conforming tetrahedra mesh with piecewise linear shape functions. The function space spanned by these piecewise linear basis is supposed to contain an approximate domain deformation solution, whose approximation error should be orthogonal to this function space. We use \mathbf{X}_i to

represent position of reference mesh's vertex i and \mathbf{x}_i to represent its corresponding position in deformed mesh. Each position vector will have three entries in 3D, for example, $\mathbf{x}_i = [\mathbf{x}_{i0}, \mathbf{x}_{i1}, \mathbf{x}_{i2}]^T$, representing its x -, y - and z - coordinates. In order to find optimal deformed mesh with minimal distortion with respect to reference \mathbf{X} , we need to define distortion energy, $\mathcal{E}_{\mathbf{X}}$, that measures the deformation quality. A common choice adopted in computer graphics community to define $\mathcal{E}_{\mathbf{X}}$ is by aggregating distortion of each mesh element

$$\mathcal{E}_{\mathbf{X}} = \sum_t A_t \Psi_{\mathbf{X}}(\mathbf{x}), \quad (5)$$

where A_t is the volume of t -th tetrahedra in reference mesh \mathbf{X} and $\Psi_{\mathbf{X}}(\cdot)$ is the distortion kernel that measures a single element's deformation. It's clear that due to linearity of differential operator, energy hessian \mathcal{H} would be linear combination of mesh elements' kernel hessian,

$$\mathcal{H} = \sum_t A_t \tilde{\mathcal{H}}_t, \quad (6)$$

which also implies that we can enforce kernel hessian to be SPD instead of enforcing energy hessian directly. The benefit is that we can largely reduce the problem size of the SPD projection step. As distortion kernel, $\Psi_{\mathbf{X}}$, is defined per mesh element, which only involves four vertices of a tetrahedra, $\tilde{\mathcal{H}}$ has a low rank decomposition,

$$\tilde{\mathcal{H}} = \mathbf{P}^T \mathbf{H} \mathbf{P}, \quad (7)$$

where $\mathbf{P} \in \mathbb{R}^{12 \times 3n}$ is a permutation matrix for mesh with n vertices and $\mathbf{H} \in \mathbb{R}^{12 \times 12}$ is the second order differential of $\Psi_{\mathbf{X}}$ with respect to single tetrahedra element's 12 vertex coordinates. Projected Newton methods then explore various approaches to efficiently diagonalize \mathbf{H} in order to apply SPD projection as in Equation 4. As this projection need to be applied for every mesh element during each iteration, its computational cost will severely affect the solver's performance. Conventional eigen decomposition method relies on two-step algorithm, Householder reflection and shifted QR iteration, whose efficiency is not acceptable for per element computation. Existing projected Newton methods either rely on numerical eigenvalue computation or only support limited distortion energy. In this work, we present an analytic eigen system for the most general representation of distortion energy parameterized using principal stretches. For distortion energy that is representable using invariants of stretch tensor, it's not surprising to see that our results match exactly what have been shown by Smith et al. [2019]. However, our approach is much more flexible and applicable to a much broader class of distortion energy.

As we are interested in how to diagonalize per element \mathbf{H} efficiently, it's enough to focus on single tetrahedra case. To simplify math notation, we will use \mathbf{X}_i and \mathbf{x}_i , $i \in \{0, 1, 2, 3\}$ to represent the element's four vertices in reference and deformed mesh. Moreover, if we assume the reference mesh never changes, which is usually the assumption adopted by geometry optimization and hyperelastic simulation, we can drop the subscript and use Ψ to represent distortion kernel. For situations where reference mesh does change, like plastic deformation, our derivation can be extended very easily.

3.2 Energy Parameterization

An essential property of Ψ is rigid motion invariant, which means it remains constant when rotating or translating the deformed mesh. For isotropic energy, such invariant property also holds when reference mesh is under rigid motion. There are quite a few ways to define Ψ to satisfy this requirement, among which the most atomic, general and popular approach in both geometry and simulation fields is using principal stretch σ . σ is defined as the singular value of deformation gradient $\mathbf{F} = \frac{\partial \mathbf{x}}{\partial \mathbf{X}}$. For piecewise linear shape function, it can be evaluated as

$$\begin{aligned} \mathbf{F} &= \mathbf{D}_w \mathbf{D}_m^{-1}, \\ \mathbf{D}_w &= [\mathbf{x}_1 - \mathbf{x}_0, \quad \mathbf{x}_2 - \mathbf{x}_0, \quad \mathbf{x}_3 - \mathbf{x}_0], \\ \mathbf{D}_m &= [\mathbf{X}_1 - \mathbf{X}_0, \quad \mathbf{X}_2 - \mathbf{X}_0, \quad \mathbf{X}_3 - \mathbf{X}_0]. \end{aligned} \quad (8)$$

And the principal stretch is evaluated by its singular value decomposition (SVD),

$$\begin{aligned} \mathbf{F} &= \mathbf{U} \mathbf{\Sigma} \mathbf{V}^T, \\ \mathbf{\Sigma} &= \begin{bmatrix} \sigma_0 & & \\ & \sigma_1 & \\ & & \sigma_2 \end{bmatrix}, \end{aligned} \quad (9)$$

For energy that accepts invertible configuration, we adopt signed SVD where \mathbf{U} and \mathbf{V} are always rotation matrix. Thus we can parameterize distortion kernel through σ as

$$\Psi(\mathbf{x}) := \Psi(\sigma_0(\mathbf{x}), \sigma_1(\mathbf{x}), \sigma_2(\mathbf{x})). \quad (10)$$

The first order differential of Ψ can then be evaluated as

$$\nabla \Psi = \begin{bmatrix} \frac{\partial \Psi}{\partial \mathbf{x}_{00}} \\ \frac{\partial \Psi}{\partial \mathbf{x}_{01}} \\ \frac{\partial \Psi}{\partial \mathbf{x}_{02}} \\ \vdots \\ \frac{\partial \Psi}{\partial \mathbf{x}_{30}} \\ \frac{\partial \Psi}{\partial \mathbf{x}_{31}} \\ \frac{\partial \Psi}{\partial \mathbf{x}_{32}} \end{bmatrix}, \quad \frac{\partial \Psi}{\partial \mathbf{x}_{ij}} = \sum_{k=0}^2 \frac{\partial \Psi}{\partial \sigma_k} \frac{\partial \sigma_k}{\partial \mathbf{x}_{ij}}, \quad (11)$$

and its second order differential \mathbf{H} can be written as

$$\mathbf{H} = \begin{bmatrix} \frac{\partial^2 \Psi}{\partial \mathbf{x}_{00} \partial \mathbf{x}_{00}} & \cdots & \frac{\partial^2 \Psi}{\partial \mathbf{x}_{00} \partial \mathbf{x}_{32}} \\ \vdots & \ddots & \vdots \\ \frac{\partial^2 \Psi}{\partial \mathbf{x}_{32} \partial \mathbf{x}_{00}} & \cdots & \frac{\partial^2 \Psi}{\partial \mathbf{x}_{32} \partial \mathbf{x}_{32}} \end{bmatrix}, \quad (12)$$

where each entry is defined as

$$\frac{\partial^2 \Psi}{\partial \mathbf{x}_{ip} \partial \mathbf{x}_{jq}} = \sum_{k=0}^2 \sum_{l=0}^2 \frac{\partial^2 \Psi}{\partial \sigma_k \partial \sigma_l} \frac{\partial \sigma_l}{\partial \mathbf{x}_{ip}} \frac{\partial \sigma_k}{\partial \mathbf{x}_{jq}} + \sum_{k=0}^2 \frac{\partial \Psi}{\partial \sigma_k} \frac{\partial^2 \sigma_k}{\partial \mathbf{x}_{ip} \partial \mathbf{x}_{jq}}. \quad (13)$$

Other choices of parameterization are also possible, for example, invariants of stretch tensor [Smith et al. 2019], $I_1 = \sum_i \sigma_i$, $I_2 = \sum_i \sigma_i^2$, $I_3 = \prod_i \sigma_i$, or invariants of Cauchy-Green strain tensor [Teran et al. 2005], $I = \sum_i \sigma_i^2$, $II = \sum_{i \neq j} \sigma_i^2 \sigma_j^2$, $III = \prod_i \sigma_i$, which are all easily representable in principal stretches but not vice versa. Thus our work provides the most general analysis framework for eigen system of \mathbf{H} .

3.3 SVD Differential

Before diving into the detailed discussion, we introduce one more concept, SVD differential, which serves as an essential building block to our eigen analysis. Even though SVD computation (Equation 9) relies on iterative numerical algorithms, SVD differential provides analytical derivatives of SVD components as long as parameterization of \mathbf{F} is given. Here we just list several important results that will be used in our later discussion and leave all detailed derivations in Appendix A. Suppose \mathbf{F} is parameterized by some scalar x , then we have

$$\frac{\partial \sigma_i}{\partial x} = \mathbf{U}_i^T \frac{\partial \mathbf{F}}{\partial x} \mathbf{V}_i, \quad \mathbf{U}^T \frac{\partial \mathbf{U}}{\partial x} = \omega_x^u, \quad \frac{\partial \mathbf{V}^T}{\partial x} \mathbf{V} = -\omega_x^v, \quad (14)$$

where both ω_x^u and ω_x^v are skew-symmetric matrix,

$$\omega = \begin{bmatrix} 0 & \omega^0 & \omega^1 \\ -\omega^0 & 0 & \omega^2 \\ -\omega^1 & -\omega^2 & 0 \end{bmatrix}. \quad (15)$$

Their entries can be computed by solving three 2×2 linear systems, such as

$$\begin{bmatrix} \sigma_1 & -\sigma_0 \\ -\sigma_0 & \sigma_1 \end{bmatrix} \begin{bmatrix} \omega_x^{u0} \\ \omega_x^{v0} \end{bmatrix} = \begin{bmatrix} \mathbf{U}_1^T \frac{\partial \mathbf{F}}{\partial x} \mathbf{V}_1 \\ \mathbf{U}_0^T \frac{\partial \mathbf{F}}{\partial x} \mathbf{V}_0 \end{bmatrix}. \quad (16)$$

Such systems will become singular when two principal stretches are identical or sum to zero, which is a numerical stability issue that requires special treatment as shown in previous work [Sin et al. 2011; Stomakhin et al. 2012; Xu et al. 2015]. Same as Smith et al. [Smith et al. 2019], we avoid such annoying numerical artifact by providing the analytical eigen system directly. Finally, we still need the second order derivatives of principal stretches. Suppose now \mathbf{F} can be parameterized by x and y , then we have

$$\frac{\partial^2 \Sigma}{\partial x \partial y} = \text{diag}(\omega_x^u \Sigma \omega_y^v + \omega_y^u \Sigma \omega_x^v - \Sigma \omega_x^v \omega_y^v - \omega_y^u \omega_x^u \Sigma), \quad (17)$$

where $\text{diag}(\cdot)$ extracts the diagonal part of the input matrix.

4 EIGEN ANALYSIS

In this section, we will show the derivation of finding the eigen system for distortion kernel's second order differential \mathbf{H} defined in Equation 12. We demonstrate that for 3D problems, \mathbf{H} has a null space of dimension three and six of its eigen pairs can be analytically obtained while whether the other three have analytical expressions depends on the kernel definition in terms of principal stretches. As an overview of our approach, we first explore the null space of \mathbf{H} and reduce our problem from $\mathbf{H} \in \mathbb{R}^{12 \times 12}$ to $\tilde{\mathbf{H}} \in \mathbb{R}^{9 \times 9}$. Next, we show that the $\tilde{\mathbf{H}}$ can be decomposed into one 3×3 block and one 6×6 diagonal block that contains the eigen values with analytical expressions. Finally, we apply our results to several energy kernel examples to demonstrate its flexibility and usefulness. Again, we will only provide results of essential steps and encourage interested readers to appendix for detailed derivations.

4.1 From $\mathbb{R}^{12 \times 12}$ To $\mathbb{R}^{9 \times 9}$

We first show that $\mathbf{H} \in \mathbb{R}^{12 \times 12}$ can be factorized as $\mathbf{H} = \mathbf{K}^T \tilde{\mathbf{H}} \mathbf{K}$, $\mathbf{K} \in \mathbb{R}^{9 \times 12}$, $\tilde{\mathbf{H}} \in \mathbb{R}^{9 \times 9}$. This is due to the fact that \mathbf{H} has a null space of dimension three. For 3D P1 element, we have the following

equality always hold,

$$\frac{\partial \Psi}{\partial \mathbf{x}_0} + \frac{\partial \Psi}{\partial \mathbf{x}_1} + \frac{\partial \Psi}{\partial \mathbf{x}_2} + \frac{\partial \Psi}{\partial \mathbf{x}_3} \equiv \mathbf{0}. \quad (18)$$

Intuitively, this means kernel derivative acts passively. Or if we take a physical point of view, internal forces should cancel out when there is no external forces. Detailed proof is provided in Appendix B. An immediate result is that we can represent $\frac{\partial \Psi}{\partial \mathbf{x}_i}$ using the other three. For example, Equation 11 can be rewritten as

$$\nabla \Psi = \begin{bmatrix} \frac{\partial \Psi}{\partial \mathbf{x}_{00}} \\ \frac{\partial \Psi}{\partial \mathbf{x}_{01}} \\ \frac{\partial \Psi}{\partial \mathbf{x}_{02}} \\ \vdots \\ -\left(\frac{\partial \Psi}{\partial \mathbf{x}_{00}} + \frac{\partial \Psi}{\partial \mathbf{x}_{10}} + \frac{\partial \Psi}{\partial \mathbf{x}_{20}}\right) \\ -\left(\frac{\partial \Psi}{\partial \mathbf{x}_{01}} + \frac{\partial \Psi}{\partial \mathbf{x}_{11}} + \frac{\partial \Psi}{\partial \mathbf{x}_{21}}\right) \\ -\left(\frac{\partial \Psi}{\partial \mathbf{x}_{02}} + \frac{\partial \Psi}{\partial \mathbf{x}_{12}} + \frac{\partial \Psi}{\partial \mathbf{x}_{22}}\right) \end{bmatrix}, \quad (19)$$

where we replace the last three entries with linear combination of the first nine ones. Similar idea can also be applied to \mathbf{H} which is derivative of $\nabla \Psi$. If we divide \mathbf{H} into a block 2×2 form,

$$\mathbf{H} = \begin{bmatrix} \tilde{\mathbf{H}} & \hat{\mathbf{H}} \\ \hat{\mathbf{H}}^T & \hat{\mathbf{H}} \end{bmatrix}, \quad \tilde{\mathbf{H}} = \begin{bmatrix} \frac{\partial^2 \Psi}{\partial \mathbf{x}_{00} \partial \mathbf{x}_{00}} & \cdots & \frac{\partial^2 \Psi}{\partial \mathbf{x}_{00} \partial \mathbf{x}_{22}} \\ \vdots & \ddots & \vdots \\ \frac{\partial^2 \Psi}{\partial \mathbf{x}_{22} \partial \mathbf{x}_{00}} & \cdots & \frac{\partial^2 \Psi}{\partial \mathbf{x}_{22} \partial \mathbf{x}_{22}} \end{bmatrix}, \quad (20)$$

then entries of $\tilde{\mathbf{H}} \in \mathbb{R}^{9 \times 3}$ and $\hat{\mathbf{H}} \in \mathbb{R}^{3 \times 3}$ can be represented as linear combination of entries in $\tilde{\mathbf{H}} \in \mathbb{R}^{9 \times 9}$. Notice $\tilde{\mathbf{H}}$ is the second order differential of distortion kernel with respect to just the first three element vertex coordinates, which excludes the fourth one's coordinates. As shown in Appendix C, we can thus have

$$\mathbf{K} = \begin{bmatrix} \mathbf{I} & & & -\mathbf{I} \\ & \mathbf{I} & & -\mathbf{I} \\ & & \mathbf{I} & -\mathbf{I} \end{bmatrix} \quad (21)$$

satisfying $\mathbf{H} = \mathbf{K}^T \tilde{\mathbf{H}} \mathbf{K}$. Here \mathbf{I} is 3×3 identity matrix.

4.2 Analytic Decomposition of $\tilde{\mathbf{H}}$

Given the above factorization result, we only need to show how SPD projection can be efficiently applied to $\tilde{\mathbf{H}}$ in the following. Thus we are going to demonstrate how $\tilde{\mathbf{H}}$ can be analytically decomposed into one 3×3 block and one 6×6 diagonal block. According to Equation 13, we can also separate $\tilde{\mathbf{H}}$ into two terms,

$$\tilde{\mathbf{H}} = \tilde{\mathbf{H}}^\# + \tilde{\mathbf{H}}^*, \quad (22)$$

where entries of $\tilde{\mathbf{H}}^\#$ and $\tilde{\mathbf{H}}^*$ are $\sum_{k=0}^2 \sum_{l=0}^2 \frac{\partial^2 \Psi}{\partial \sigma_k \partial \sigma_l} \frac{\partial \sigma_l}{\partial \mathbf{x}_{ip}} \frac{\partial \sigma_k}{\partial \mathbf{x}_{jq}}$ and $\sum_{k=0}^2 \frac{\partial \Psi}{\partial \sigma_k} \frac{\partial^2 \sigma_k}{\partial \mathbf{x}_{ip} \partial \mathbf{x}_{jq}}$, $i, j, p, q \in \{0, 1, 2\}$ correspondingly. It's easy to see that if both $\tilde{\mathbf{H}}^\#$ and $\tilde{\mathbf{H}}^*$ are SPD, $\tilde{\mathbf{H}}$ is also SPD. Again we are going to decompose them into some simpler forms where SPD projection is efficient to be applied. We will show that $\tilde{\mathbf{H}}^\#$ can be decomposed into a 3×3 block that only depends on distortion kernel definition in terms of principal stretches while $\tilde{\mathbf{H}}^*$ can be decomposed into diagonal form. Based on entry formulation of $\tilde{\mathbf{H}}^\#$, it's easy

to see that it has the following decomposition, $\tilde{\mathbf{H}}^\# = (\tilde{\mathbf{K}}^\#)^T \tilde{\mathbf{D}}^\# \tilde{\mathbf{K}}^\#$, where

$$\tilde{\mathbf{D}}^\# = \begin{bmatrix} \frac{\partial^2 \Psi}{\partial \sigma_0 \partial \sigma_0} & \frac{\partial^2 \Psi}{\partial \sigma_0 \partial \sigma_1} & \frac{\partial^2 \Psi}{\partial \sigma_0 \partial \sigma_2} \\ \frac{\partial^2 \Psi}{\partial \sigma_1 \partial \sigma_0} & \frac{\partial^2 \Psi}{\partial \sigma_1 \partial \sigma_1} & \frac{\partial^2 \Psi}{\partial \sigma_1 \partial \sigma_2} \\ \frac{\partial^2 \Psi}{\partial \sigma_2 \partial \sigma_0} & \frac{\partial^2 \Psi}{\partial \sigma_2 \partial \sigma_1} & \frac{\partial^2 \Psi}{\partial \sigma_2 \partial \sigma_2} \end{bmatrix}, \quad \tilde{\mathbf{K}}^\# = \begin{bmatrix} \frac{\partial \sigma_0}{\partial \mathbf{x}_{00}} & \cdots & \frac{\partial \sigma_0}{\partial \mathbf{x}_{22}} \\ \frac{\partial \sigma_1}{\partial \mathbf{x}_{00}} & \cdots & \frac{\partial \sigma_1}{\partial \mathbf{x}_{22}} \\ \frac{\partial \sigma_2}{\partial \mathbf{x}_{00}} & \cdots & \frac{\partial \sigma_2}{\partial \mathbf{x}_{22}} \end{bmatrix}. \quad (23)$$

As $\tilde{\mathbf{D}}^\#$ is the second order differential of distortion kernel with respect to principal stretches, it only requires kernel definition, which is usually provided before hand, to determine its diagonalizability. In ??, we will utilize this decomposition to provide analytical eigen pairs for several distortion kernel examples. In cases where no analytical eigen expressions exist for this 3×3 system, we adopt numerical solutions instead.

For $\tilde{\mathbf{H}}^*$, it's not easy to see similar decomposition immediately. We leave all detailed derivations in Appendix D and give the essential steps here. Similar to Equation 22, we first separate $\tilde{\mathbf{H}}^*$ into three terms,

$$\tilde{\mathbf{H}}^* = \tilde{\mathbf{H}}^{*0,1} + \tilde{\mathbf{H}}^{*1,2} + \tilde{\mathbf{H}}^{*2,0}, \quad (24)$$

where each term will have a decomposition. Next, we take $\tilde{\mathbf{H}}^{*0,1}$ as an illustration example and the other two can be decomposed in similar manner. As a first step, we obtain the following decomposition,

$$\tilde{\mathbf{H}}^{*0,1} = (\tilde{\mathbf{K}}^{*0,1})^T \tilde{\mathbf{D}}^{*0,1} \tilde{\mathbf{K}}^{*0,1}, \quad \tilde{\mathbf{K}}^{*0,1} = \begin{bmatrix} \omega_{\mathbf{x}_{00}}^{u0} & \cdots & \omega_{\mathbf{x}_{22}}^{u0} \\ \omega_{\mathbf{x}_{00}}^{v0} & \cdots & \omega_{\mathbf{x}_{22}}^{v0} \end{bmatrix}, \quad (25)$$

$$\tilde{\mathbf{D}}^{*0,1} = \begin{bmatrix} \sigma_0 \frac{\partial \Psi}{\partial \sigma_0} + \sigma_1 \frac{\partial \Psi}{\partial \sigma_1} & -(\sigma_0 \frac{\partial \Psi}{\partial \sigma_1} + \sigma_1 \frac{\partial \Psi}{\partial \sigma_0}) \\ -(\sigma_1 \frac{\partial \Psi}{\partial \sigma_0} + \sigma_0 \frac{\partial \Psi}{\partial \sigma_1}) & \sigma_1 \frac{\partial \Psi}{\partial \sigma_1} + \sigma_0 \frac{\partial \Psi}{\partial \sigma_0} \end{bmatrix},$$

where each column vector of $\tilde{\mathbf{K}}^{*0,1} \in \mathbb{R}^{2 \times 9}$ can be obtained by solving a small 2×2 linear system as shown in Equation 16. Thus we have

$$\begin{bmatrix} \omega_{\mathbf{x}_{ip}}^{u0} \\ \omega_{\mathbf{x}_{ip}}^{v0} \end{bmatrix} = \frac{1}{\sigma_1^2 - \sigma_0^2} \begin{bmatrix} \sigma_1 & \sigma_0 \\ \sigma_0 & \sigma_1 \end{bmatrix} \begin{bmatrix} \mathbf{U}_0^T \frac{\partial \mathbf{F}}{\partial \mathbf{x}_{ip}} \mathbf{V}_1 \\ \mathbf{U}_1^T \frac{\partial \mathbf{F}}{\partial \mathbf{x}_{ip}} \mathbf{V}_0 \end{bmatrix}, \quad i, p \in \{0, 1, 2\}. \quad (26)$$

By regrouping matrix products, we obtain a new decomposition for $\tilde{\mathbf{H}}^{*0,1}$ as $(\tilde{\mathbf{K}}^{*0,1})^T \tilde{\mathbf{D}}^{*0,1} \tilde{\mathbf{K}}^{*0,1}$, where

$$\tilde{\mathbf{D}}^{*0,1} = \frac{1}{(\sigma_1^2 - \sigma_0^2)^2} \begin{bmatrix} \sigma_1 & \sigma_0 \\ \sigma_0 & \sigma_1 \end{bmatrix}^T \tilde{\mathbf{D}}^{*0,1} \begin{bmatrix} \sigma_1 & \sigma_0 \\ \sigma_0 & \sigma_1 \end{bmatrix}, \quad (27)$$

$$\tilde{\mathbf{K}}^{*0,1} = \begin{bmatrix} \mathbf{U}_0^T \frac{\partial \mathbf{F}}{\partial \mathbf{x}_{00}} \mathbf{V}_1 & \cdots & \mathbf{U}_0^T \frac{\partial \mathbf{F}}{\partial \mathbf{x}_{22}} \mathbf{V}_1 \\ \mathbf{U}_1^T \frac{\partial \mathbf{F}}{\partial \mathbf{x}_{00}} \mathbf{V}_0 & \cdots & \mathbf{U}_1^T \frac{\partial \mathbf{F}}{\partial \mathbf{x}_{22}} \mathbf{V}_0 \end{bmatrix}.$$

In this case, we can diagonalize $\tilde{\mathbf{D}}^{*0,1}$ as

$$\tilde{\mathbf{D}}^{*0,1} = \begin{bmatrix} -1 & 1 \\ 1 & 1 \end{bmatrix}^T \begin{bmatrix} \frac{\partial \Psi}{\partial \sigma_0} - \frac{\partial \Psi}{\partial \sigma_1} & 0 \\ \frac{\partial \Psi}{\partial \sigma_0 - \sigma_1} & \frac{\partial \Psi}{\partial \sigma_0} + \frac{\partial \Psi}{\partial \sigma_1} \\ 0 & \frac{\partial \Psi}{\partial \sigma_0 + \sigma_1} \end{bmatrix} \begin{bmatrix} -1 & 1 \\ 1 & 1 \end{bmatrix}. \quad (28)$$

To see this more clearly, we first separate $\tilde{\mathbf{D}}^{*0,1}$ into sum of the following two terms,

$$\tilde{\mathbf{D}}^{*0,1} = \frac{\partial \Psi}{\partial \sigma_0} - \frac{\partial \Psi}{\partial \sigma_1} \begin{bmatrix} 1 & -1 \\ -1 & 1 \end{bmatrix} + \frac{\partial \Psi}{\partial \sigma_0} + \frac{\partial \Psi}{\partial \sigma_1} \begin{bmatrix} 1 & 1 \\ 1 & 1 \end{bmatrix}, \quad (29)$$

where the two constant 2×2 matrices have simple diagonal forms as

$$\begin{aligned} \begin{bmatrix} 1 & -1 \\ -1 & 1 \end{bmatrix} &= \begin{bmatrix} -1 & 1 \\ 1 & 1 \end{bmatrix}^T \begin{bmatrix} 2 & 0 \\ 0 & 0 \end{bmatrix} \begin{bmatrix} -1 & 1 \\ 1 & 1 \end{bmatrix}, \\ \begin{bmatrix} 1 & 1 \\ 1 & 1 \end{bmatrix} &= \begin{bmatrix} -1 & 1 \\ 1 & 1 \end{bmatrix}^T \begin{bmatrix} 0 & 0 \\ 0 & 2 \end{bmatrix} \begin{bmatrix} -1 & 1 \\ 1 & 1 \end{bmatrix}. \end{aligned} \quad (30)$$

Notice both matrices share the same eigen space, so we can combine these two factorizations together and obtain Equation 28. Finally, given the above results, we can diagonalize $\tilde{\mathbf{H}}^{*0,1}$ through decomposition, $(\tilde{\mathbf{K}}^{*0,1})^T \tilde{\mathbf{D}}^{*0,1} \tilde{\mathbf{K}}^{*0,1}$, where

$$\tilde{\mathbf{D}}^{*0,1} = \begin{bmatrix} \frac{\partial \Psi}{\partial \sigma_0} - \frac{\partial \Psi}{\partial \sigma_1} & 0 \\ 0 & \frac{\partial \Psi}{\partial \sigma_0} + \frac{\partial \Psi}{\partial \sigma_1} \end{bmatrix}, \quad \tilde{\mathbf{K}}^{*0,1} = \begin{bmatrix} -1 & 1 \\ 1 & 1 \end{bmatrix} \tilde{\mathbf{K}}^{\bullet 0,1}. \quad (31)$$

For the other two matrices, $\tilde{\mathbf{H}}^{*1,2}$ and $\tilde{\mathbf{H}}^{*2,0}$, we can obtain similar results, which justifies our claim that six eigen pairs of \mathbf{H} have analytical forms. If we combine all the derivations from this section together, we obtain the analytical decomposition of \mathbf{H} as

$$\mathbf{H} = \mathbf{K}^T \tilde{\mathbf{H}} \mathbf{K} = \mathbf{K}^T \tilde{\mathbf{K}}^T \tilde{\mathbf{D}} \tilde{\mathbf{K}} \mathbf{K},$$

$$\tilde{\mathbf{D}} = \begin{bmatrix} \tilde{\mathbf{D}}^{\#} & & & \\ & \tilde{\mathbf{D}}^{*0,1} & & \\ & & \tilde{\mathbf{D}}^{*1,2} & \\ & & & \tilde{\mathbf{D}}^{*2,0} \end{bmatrix}, \quad \tilde{\mathbf{K}} = \begin{bmatrix} \tilde{\mathbf{K}}^{\#} \\ \tilde{\mathbf{K}}^{*0,1} \\ \tilde{\mathbf{K}}^{*1,2} \\ \tilde{\mathbf{K}}^{*2,0} \end{bmatrix}. \quad (32)$$

Moreover, the six eigen values with analytical expressions may cause numerical instability issues as sum and difference of principal stretches appear as denominators (see Equation 28). At the first glance, such annoying problem seems to be unavoidable but depends on the distortion kernel definition.

REFERENCES

Noam Aigerman, Roi Poranne, and Yaron Lipman. 2015. Seamless Surface Mappings. *ACM Transactions on Graphics* 34, 4 (2015), 72:1–72:13.

Marc Alexa, Daniel Cohen-Or, and David Levin. 2000. As-rigid-as-possible Shape Interpolation. In *Proceedings of the 27th Annual Conference on Computer Graphics and Interactive Techniques (SIGGRAPH '00)*. 157–164.

Mirela Ben-Chen, Craig Gotsman, and Guy Bunin. 2008. Conformal Flattening by Curvature Prescription and Metric Scaling. *Computer Graphics Forum* (2008).

J Bonet and AJ Burton. 1998. A simple orthotropic, transversely isotropic hyperelastic constitutive equation for large strain computations. *Computer methods in applied mechanics and engineering* 162, 1 (1998), 151–164.

Javier Bonet and Richard D. Wood. 2008. *Nonlinear Continuum Mechanics for Finite Element Analysis*. Cambridge University Press.

Sofien Bouaziz, Sebastian Martin, Tiantian Liu, Ladislav Kavan, and Mark Pauly. 2014. Projective Dynamics: Fusing Constraint Projections for Fast Simulation. *ACM Transactions on Graphics* 33, 4 (2014), 154:1–154:11.

Isaac Chao, Ulrich Pinkall, Patrick Sanan, and Peter Schröder. 2010. A Simple Geometric Model for Elastic Deformations. *ACM Transactions on Graphics* 29, 4 (2010), 38:1–38:6.

Renjie Chen and Ofir Weber. 2017. GPU-accelerated Locally Injective Shape Deformation. *ACM Transactions on Graphics* 36, 6 (2017), 214:1–214:13.

S. Claiici, M. Bessmeltsev, S. Schaefer, and J. Solomon. 2017. Isometry-Aware Preconditioning for Mesh Parameterization. *Computer Graphics Forum* 36, 5 (2017), 37–47.

Mathieu Desbrun, Mark Meyer, and Pierre Alliez. 2002. Intrinsic Parameterizations of Surface Meshes. *Computer Graphics Forum* 21 (2002), 209–218.

Michael Eigensatz and Mark Pauly. 2009. Positional, Metric, and Curvature Control for Constraint-Based Surface Deformation. *Computer Graphics Forum* 28 (04 2009).

Xiao-Ming Fu and Yang Liu. 2016. Computing Inversion-free Mappings by Simplex Assembly. *ACM Transactions on Graphics* 35, 6 (2016), 216:1–216:12.

Xiao-Ming Fu, Yang Liu, and Baining Guo. 2015. Computing Locally Injective Mappings by Advanced MIPS. *ACM Transactions on Graphics* 34, 4 (2015), 71:1–71:12.

Kai Hormann and Günther Greiner. 2002. MIPS: An Efficient Global Parameterization Method. In *Technical Report. DTIC Document*.

Takeo Igarashi, Tomer Moscovich, and John F. Hughes. 2005. As-rigid-as-possible Shape Manipulation. *ACM Transactions on Graphics* 24, 3 (2005), 1134–1141.

Shahar Z. Kovalsky, Meirav Galun, and Yaron Lipman. 2016. Accelerated Quadratic Proxy for Geometric Optimization. *ACM Transactions on Graphics* 35, 4 (2016), 134:1–134:11.

Bruno Lévy, Sylvain Petitjean, Nicolas Ray, and Jérôme Maillot. 2002. Least Squares Conformal Maps for Automatic Texture Atlas Generation. *ACM Transactions on Graphics* 21, 3 (2002), 362–371.

Yaron Lipman, Olga Sorkine, David Levin, and Daniel Cohen-Or. 2005. Linear Rotation-invariant Coordinates for Meshes. *ACM Transactions on Graphics* 24, 3 (2005), 479–487.

Ligang Liu, Chunyang Ye, Ruiqi Ni, and Xiao-Ming Fu. 2018. Progressive Parameterizations. *ACM Transactions on Graphics* 37, 4 (2018), 41:1–41:12.

Ligang Liu, Lei Zhang, Yin Xu, Craig Gotsman, and Steven J. Gortler. 2008. A Local/Global Approach to Mesh Parameterization. In *Proceedings of the Symposium on Geometry Processing (SGP '08)*. 1495–1504.

Tiantian Liu, Sofien Bouaziz, and Ladislav Kavan. 2017. Quasi-Newton Methods for Real-Time Simulation of Hyperelastic Materials. *ACM Transactions on Graphics* 36, 4 (2017).

Aleka McAdams, Yongning Zhu, Andrew Selle, Mark Empey, Rasmus Tamstorf, Joseph Teran, and Efthychios Sifakis. 2011. Efficient Elasticity for Character Skinning with Contact and Collisions. *ACM Transactions on Graphics* 30, 4 (2011), 37:1–37:12.

Melvin Mooney. 1940. A Theory of Large Elastic Deformation. *Journal of Applied Physics* 11, 9 (1940), 582–592.

Patrick Mullen, Yiyang Tong, Pierre Alliez, and Mathieu Desbrun. 2008. Spectral Conformal Parameterization. In *Proceedings of the Symposium on Geometry Processing (SGP '08)*. 1487–1494.

John Neuberger. 2006. *Steepest descent for general systems of linear differential equations in Hilbert space*. Vol. 1032. 390–406.

John Neuberger. 2010. *Sobolev Gradients and Differential Equations*. Vol. 1670.

J. Nocedal and S. Wright. 2006. *Numerical Optimization*. Springer New York.

Raymond Ogden. 1972. Large Deformation Isotropic Elasticity – On the Correlation of Theory and Experiment for Incompressible Rubberlike Solids. In *Proceedings of the Royal Society of London. Series A, Mathematical and Physical Sciences*. 565–584.

Théodore Papadopoulos and Manolis I. A. Lourakis. 2000. Estimating the Jacobian of the Singular Value Decomposition: Theory and Applications. In *Proceedings of the 6th European Conference on Computer Vision-Part I (ECCV '00)*. 554–570.

Yue Peng, Bailin Deng, Juyong Zhang, Fanyu Geng, Wenjie Qin, and Ligang Liu. 2018. Anderson Acceleration for Geometry Optimization and Physics Simulation. *ACM Transactions on Graphics* 37, 4 (2018), 42:1–42:14.

Michael Rabinovich, Roi Poranne, Daniele Panozzo, and Olga Sorkine-Hornung. 2017. Scalable Locally Injective Mappings. *ACM Transactions on Graphics* 36, 4 (2017).

Ronald S. Rivlin. 1948. Some Applications of Elasticity Theory to Rubber Engineering. In *Proc. Rubber Technology Conference*. 1–8.

Christian Schüller, Ladislav Kavan, Daniele Panozzo, and Olga Sorkine-Hornung. 2013. Locally Injective Mappings. *Computer Graphics Forum* 32, 5 (2013), 125–135.

Anna Shtengel, Roi Poranne, Olga Sorkine-Hornung, Shahar Z. Kovalsky, and Yaron Lipman. 2017. Geometric Optimization via Composite Majorization. *ACM Transactions on Graphics* 36, 4 (2017), 38:1–38:11.

Funshing Sin, Yufeng Zhu, Yongqiang Li, and Daniel Schroeder. 2011. Invertible Isotropic Hyperelasticity using SVD Gradients. In *In ACM SIGGRAPH / Eurographics Symposium on Computer Animation (Posters)*.

Breannan Smith, Fernando De Goes, and Theodore Kim. 2018. Stable Neo-Hookean Flesh Simulation. *ACM Transactions on Graphics* 37, 2 (2018), 12:1–12:15.

Breannan Smith, Fernando De Goes, and Theodore Kim. 2019. Analytic Eigensystems for Isotropic Distortion Energies. *ACM Transactions on Graphics* (2019).

Jason Smith and Scott Schaefer. 2015. Bijective Parameterization with Free Boundaries. *ACM Transactions on Graphics* 34, 4 (2015), 70:1–70:9.

Justin Solomon, Mirela Ben-Chen, Adrian Butscher, and Leonidas J. Guibas. 2011. As-Killing-As-Possible Vector Fields for Planar Deformation. *Computer Graphics Forum* 30 (2011), 1543–1552.

Olga Sorkine and Marc Alexa. 2007. As-rigid-as-possible Surface Modeling. In *Proceedings of the Fifth Eurographics Symposium on Geometry Processing (SGP '07)*. 109–116.

Olga Sorkine, Daniel Cohen-Or, Rony Goldenthal, and Dani Lischinski. 2002. Bounded-distortion piecewise mesh parameterization. In *Proceedings of IEEE Visualization*. 355–362.

Alexey Stomakhin, Russell Howes, Craig Schroeder, and Joseph M. Teran. 2012. Energetically Consistent Invertible Elasticity. In *Proceedings of the 11th ACM SIGGRAPH / Eurographics Conference on Computer Animation (EUROSCA'12)*. 25–32.

Joseph Teran, Efthychios Sifakis, Geoffrey Irving, and Ronald Fedkiw. 2005. Robust Quasistatic Finite Elements and Flesh Simulation. In *Proceedings of the 2005 ACM SIGGRAPH/Eurographics Symposium on Computer Animation (SCA '05)*. 181–190.

- Ofir Weber, Mirela Ben-Chen, and Craig Gotsman. 2009. Complex Barycentric Coordinates with Applications to Planar Shape Deformation. *Computer Graphics Forum* 28, 2 (2009).
- Ofir Weber, Ashish Myles, and Denis Zorin. 2012. Computing Extremal Quasiconformal Maps. *Computer Graphics Forum* 31, 5 (2012), 1679–1689.
- Ofir Weber, Olga Sorkine, Yaron Lipman, and Craig Gotsman. 2007. Context-Aware Skeletal Shape Deformation. *Computer Graphics Forum* 26, 3 (2007).
- Hongyi Xu, Funshing Sin, Yufeng Zhu, and Jernej Barbič. 2015. Nonlinear Material Design Using Principal Stretches. *ACM Transactions on Graphics* 34, 4 (2015), 75:1–75:11.
- Rhaleb Zayer, Christian Roessl, Zachi Karni, and Hans-Peter Seidel. 2005. Harmonic Guidance for Surface Deformation. *Computer Graphics Forum* 24, 3 (2005).
- Yufeng Zhu. 2020. Eigen Analysis of Mesh Distortion Energy Hessian. *Unpublished manuscript* (2020), 9.
- Yufeng Zhu, Robert Bridson, and Danny M. Kaufman. 2018. Blended Cured Quasi-Newton for Distortion Optimization. *ACM Transactions on Graphics* 37, 4 (2018), 40:1–40:14.

A SVD DIFFERENTIAL

We provide detailed derivations to compute SVD differential as shown in Equation 14 and Equation 17. If \mathbf{F} is parameterized by x and according to SVD factorization as well as product rule, we can have the following result by taking derivative with respect to x on both sides of Equation 9,

$$\begin{aligned} \frac{\partial \mathbf{F}}{\partial x} &= \frac{\partial \mathbf{U}}{\partial x} \Sigma \mathbf{V}^T + \mathbf{U} \frac{\partial \Sigma}{\partial x} \mathbf{V}^T + \mathbf{U} \Sigma \frac{\partial \mathbf{V}^T}{\partial x} \\ \Rightarrow \mathbf{U}^T \frac{\partial \mathbf{F}}{\partial x} \mathbf{V} &= \mathbf{U}^T \frac{\partial \mathbf{U}}{\partial x} \Sigma + \frac{\partial \Sigma}{\partial x} + \Sigma \frac{\partial \mathbf{V}^T}{\partial x} \mathbf{V}. \end{aligned} \quad (33)$$

If we already know \mathbf{F} in terms of x , then we can compute rotation differential $\frac{\partial \mathbf{U}}{\partial x}$, $\frac{\partial \mathbf{V}^T}{\partial x}$ and singular value differential $\frac{\partial \Sigma}{\partial x}$ given the fact that $\mathbf{U}^T \frac{\partial \mathbf{U}}{\partial x}$, $\frac{\partial \mathbf{V}^T}{\partial x} \mathbf{V}$ are skew-symmetric and $\frac{\partial \Sigma}{\partial x}$ is diagonal. This is also known as SVD gradient, which has been investigated before [Papadopoulos and Lourakis 2000]. To compute first order derivative of distortion kernel, $\nabla \Psi$, this is already enough as we only need to know $\frac{\partial \Sigma}{\partial x}$. In order to compute second order derivative, \mathbf{H} , we also need to evaluate $\mathbf{U}^T \frac{\partial \mathbf{U}}{\partial x}$, $\frac{\partial \mathbf{V}^T}{\partial x} \mathbf{V}$ and the second order derivative of singular values. If now \mathbf{F} is parameterized by x and y , we can apply similar trick as Equation 33,

$$\begin{aligned} \frac{\partial^2 \mathbf{F}}{\partial x \partial y} &= \frac{\partial^2 \mathbf{U}}{\partial x \partial y} \Sigma \mathbf{V}^T + \frac{\partial \mathbf{U}}{\partial x} \frac{\partial \Sigma}{\partial y} \mathbf{V}^T + \frac{\partial \mathbf{U}}{\partial x} \Sigma \frac{\partial \mathbf{V}^T}{\partial y} \\ &+ \frac{\partial \mathbf{U}}{\partial y} \frac{\partial \Sigma}{\partial x} \mathbf{V}^T + \mathbf{U} \frac{\partial^2 \Sigma}{\partial x \partial y} \mathbf{V}^T + \mathbf{U} \frac{\partial \Sigma}{\partial x} \frac{\partial \mathbf{V}^T}{\partial y} \\ &+ \frac{\partial \mathbf{U}}{\partial y} \Sigma \frac{\partial \mathbf{V}^T}{\partial x} + \mathbf{U} \frac{\partial \Sigma}{\partial y} \frac{\partial \mathbf{V}^T}{\partial x} + \mathbf{U} \Sigma \frac{\partial^2 \mathbf{V}^T}{\partial x \partial y}, \\ \Rightarrow \mathbf{U}^T \frac{\partial^2 \mathbf{F}}{\partial x \partial y} \mathbf{V} &= \mathbf{U}^T \frac{\partial^2 \mathbf{U}}{\partial x \partial y} \Sigma + \mathbf{U}^T \frac{\partial \mathbf{U}}{\partial x} \frac{\partial \Sigma}{\partial y} + \mathbf{U}^T \frac{\partial \mathbf{U}}{\partial x} \Sigma \frac{\partial \mathbf{V}^T}{\partial y} \mathbf{V} \\ &+ \mathbf{U}^T \frac{\partial \mathbf{U}}{\partial y} \frac{\partial \Sigma}{\partial x} + \frac{\partial^2 \Sigma}{\partial x \partial y} + \frac{\partial \Sigma}{\partial x} \frac{\partial \mathbf{V}^T}{\partial y} \mathbf{V} \\ &+ \mathbf{U}^T \frac{\partial \mathbf{U}}{\partial y} \Sigma \frac{\partial \mathbf{V}^T}{\partial x} \mathbf{V} + \frac{\partial \Sigma}{\partial y} \frac{\partial \mathbf{V}^T}{\partial x} \mathbf{V} + \Sigma \frac{\partial^2 \mathbf{V}^T}{\partial x \partial y} \mathbf{V}. \end{aligned} \quad (34)$$

Here we are interested in $\frac{\partial^2 \Sigma}{\partial x \partial y}$, but the above equations also include other unknowns, $\frac{\partial^2 \mathbf{U}}{\partial x \partial y}$ and $\frac{\partial^2 \mathbf{V}^T}{\partial x \partial y}$. To get rid of them, we first spend some effort to see what they look like. Suppose we have a rotation matrix \mathbf{Q} with parameterization x and y . Then we can derive the

following result,

$$\begin{aligned} \mathbf{Q}^T \mathbf{Q} = \mathbf{I} &\Rightarrow \frac{\partial \mathbf{Q}^T}{\partial x} \mathbf{Q} + \mathbf{Q}^T \frac{\partial \mathbf{Q}}{\partial x} = \mathbf{O}, \\ \Rightarrow \frac{\partial^2 \mathbf{Q}^T}{\partial x \partial y} \mathbf{Q} + \frac{\partial \mathbf{Q}^T}{\partial x} \frac{\partial \mathbf{Q}}{\partial y} + \frac{\partial \mathbf{Q}^T}{\partial y} \frac{\partial \mathbf{Q}}{\partial x} + \mathbf{Q}^T \frac{\partial^2 \mathbf{Q}}{\partial x \partial y} &= \mathbf{O}, \\ \Rightarrow \frac{\partial^2 \mathbf{Q}^T}{\partial x \partial y} \mathbf{Q} + \frac{\partial \mathbf{Q}^T}{\partial x} \mathbf{Q} \mathbf{Q}^T \frac{\partial \mathbf{Q}}{\partial y} + \frac{\partial \mathbf{Q}^T}{\partial y} \mathbf{Q} \mathbf{Q}^T \frac{\partial \mathbf{Q}}{\partial x} + \mathbf{Q}^T \frac{\partial^2 \mathbf{Q}}{\partial x \partial y} &= \mathbf{O}, \quad (35) \\ \Rightarrow \frac{\partial^2 \mathbf{Q}^T}{\partial x \partial y} \mathbf{Q} + \frac{\partial \mathbf{Q}^T}{\partial x} \mathbf{Q} \mathbf{Q}^T \frac{\partial \mathbf{Q}}{\partial y} &= -(\mathbf{Q}^T \frac{\partial^2 \mathbf{Q}}{\partial x \partial y} + \frac{\partial \mathbf{Q}^T}{\partial y} \mathbf{Q} \mathbf{Q}^T \frac{\partial \mathbf{Q}}{\partial x}) \\ &= -(\frac{\partial^2 \mathbf{Q}^T}{\partial x \partial y} \mathbf{Q} + \frac{\partial \mathbf{Q}^T}{\partial x} \mathbf{Q} \mathbf{Q}^T \frac{\partial \mathbf{Q}}{\partial y})^T. \end{aligned}$$

Thus $\mathbf{U}^T \frac{\partial^2 \mathbf{U}}{\partial x \partial y} + \frac{\partial \mathbf{U}^T}{\partial y} \mathbf{U} \mathbf{U}^T \frac{\partial \mathbf{U}}{\partial x}$ and $\frac{\partial^2 \mathbf{V}^T}{\partial x \partial y} \mathbf{V} + \frac{\partial \mathbf{V}^T}{\partial x} \mathbf{V} \mathbf{V}^T \frac{\partial \mathbf{V}}{\partial y}$ are also skew-symmetric. If we adopt the same math notation convention used in Equation 14, we have

$$\begin{aligned} \omega_x^u &:= \mathbf{U}^T \frac{\partial \mathbf{U}}{\partial x}, \omega_y^u := \mathbf{U}^T \frac{\partial \mathbf{U}}{\partial y}, \omega_x^v := -\frac{\partial \mathbf{V}^T}{\partial x} \mathbf{V}, \omega_y^v := -\frac{\partial \mathbf{V}^T}{\partial y} \mathbf{V}, \\ \omega_{xy}^u &:= \mathbf{U}^T \frac{\partial^2 \mathbf{U}}{\partial x \partial y} + \frac{\partial \mathbf{U}^T}{\partial y} \mathbf{U} \mathbf{U}^T \frac{\partial \mathbf{U}}{\partial x} = \mathbf{U}^T \frac{\partial^2 \mathbf{U}}{\partial x \partial y} - \omega_y^u \omega_x^u, \quad (36) \\ \omega_{xy}^v &:= -(\frac{\partial^2 \mathbf{V}^T}{\partial x \partial y} \mathbf{V} + \frac{\partial \mathbf{V}^T}{\partial x} \mathbf{V} \mathbf{V}^T \frac{\partial \mathbf{V}}{\partial y}) = -\frac{\partial^2 \mathbf{V}^T}{\partial x \partial y} \mathbf{V} + \omega_x^v \omega_y^v. \end{aligned}$$

Then we can rewrite Equation 34 as

$$\begin{aligned} \mathbf{U}^T \frac{\partial^2 \mathbf{F}}{\partial x \partial y} \mathbf{V} &= (\omega_{xy}^u + \omega_y^u \omega_x^u) \Sigma + \omega_x^u \frac{\partial \Sigma}{\partial y} - \omega_x^u \Sigma \omega_y^v \\ &+ \omega_y^u \frac{\partial \Sigma}{\partial x} + \frac{\partial^2 \Sigma}{\partial x \partial y} - \frac{\partial \Sigma}{\partial x} \omega_y^v \\ &- \omega_y^u \Sigma \omega_x^v - \frac{\partial \Sigma}{\partial y} \omega_x^v - \Sigma (\omega_{xy}^v - \omega_x^v \omega_y^v). \end{aligned} \quad (37)$$

Notice that product of skew-symmetric matrix and diagonal matrix has zero diagonals and the second order differential $\frac{\partial^2 \Sigma}{\partial x \partial y}$ is diagonal matrix, we have

$$\frac{\partial^2 \Sigma}{\partial x \partial y} = \text{diag}(\omega_x^u \Sigma \omega_y^v + \omega_y^u \Sigma \omega_x^v - \Sigma \omega_x^v \omega_y^v - \omega_y^u \omega_x^u \Sigma + \mathbf{U}^T \frac{\partial^2 \mathbf{F}}{\partial x \partial y} \mathbf{V}), \quad (38)$$

where $\text{diag}(\cdot)$ extracts the diagonal part of the input matrix. In our P1 element setting, \mathbf{F} is a linear function (see Equation 8), so its second order derivative is always zero. Thus we can further simplify our result and get Equation 17.

B ZERO NET ELEMENT DERIVATIVE

We then provide proof for the statement of Equation 18.

PROOF. According to the definition of \mathbf{F} in Equation 8, it's easy to verify that the following equality always holds for any p -th coordinate,

$$\sum_{i=0}^3 \frac{\partial \mathbf{F}}{\partial \mathbf{x}_{ip}} \equiv \mathbf{O}, \quad \forall p \in \{0, 1, 2\}, \quad (39)$$

where \mathbf{O} is a 3×3 zero matrix. Then according to Equation 14, we have

$$\begin{aligned} \sum_{i=0}^3 \frac{\partial \sigma_k}{\partial \mathbf{x}_{ip}} &= \mathbf{U}_k^T \left(\sum_{i=0}^3 \frac{\partial \mathbf{F}}{\partial \mathbf{x}_{ip}} \right) \mathbf{V}_k \equiv 0, \quad \forall k \in \{0, 1, 2\}, \\ \Rightarrow \sum_{k=0}^2 \frac{\partial \Psi}{\partial \sigma_k} \sum_{i=0}^3 \frac{\partial \sigma_k}{\partial \mathbf{x}_{ip}} &\equiv 0 \Rightarrow \sum_{i=0}^3 \frac{\partial \Psi}{\partial \mathbf{x}_{ip}} \equiv 0, \quad \forall p \in \{0, 1, 2\}. \end{aligned} \quad (40)$$

As the above equality holds for any p -th coordinate, the statement of Equation 18 is always true. \square

C FROM $\mathbb{R}^{12 \times 12}$ TO $\mathbb{R}^{9 \times 9}$

Next we utilize the result just proved to show $\mathbf{H} = \mathbf{K}^T \tilde{\mathbf{H}} \mathbf{K}$ as in Equation 20 and Equation 21. The key idea is that entries of $\tilde{\mathbf{H}}$ and $\hat{\mathbf{H}}$ can be represented as linear combinations of entries from $\tilde{\mathbf{H}}$, where

$$\begin{aligned} \tilde{\mathbf{H}} &= \begin{bmatrix} \frac{\partial^2 \Psi}{\partial \mathbf{x}_{00} \partial \mathbf{x}_{30}} & \cdots & \frac{\partial^2 \Psi}{\partial \mathbf{x}_{00} \partial \mathbf{x}_{32}} \\ \vdots & \ddots & \vdots \\ \frac{\partial^2 \Psi}{\partial \mathbf{x}_{22} \partial \mathbf{x}_{30}} & \cdots & \frac{\partial^2 \Psi}{\partial \mathbf{x}_{22} \partial \mathbf{x}_{32}} \end{bmatrix}, \\ \hat{\mathbf{H}} &= \begin{bmatrix} \frac{\partial^2 \Psi}{\partial \mathbf{x}_{30} \partial \mathbf{x}_{30}} & \cdots & \frac{\partial^2 \Psi}{\partial \mathbf{x}_{30} \partial \mathbf{x}_{32}} \\ \vdots & \ddots & \vdots \\ \frac{\partial^2 \Psi}{\partial \mathbf{x}_{32} \partial \mathbf{x}_{30}} & \cdots & \frac{\partial^2 \Psi}{\partial \mathbf{x}_{32} \partial \mathbf{x}_{32}} \end{bmatrix}. \end{aligned} \quad (41)$$

First, recall that Equation 18 holds true for every p -th coordinate and we have,

$$\frac{\partial \Psi}{\partial \mathbf{x}_{0p}} + \frac{\partial \Psi}{\partial \mathbf{x}_{1p}} + \frac{\partial \Psi}{\partial \mathbf{x}_{2p}} + \frac{\partial \Psi}{\partial \mathbf{x}_{3p}} = 0, \quad \forall p \in \{0, 1, 2\}. \quad (42)$$

To compute entries of $\tilde{\mathbf{H}}$, namely $\frac{\partial^2 \Psi}{\partial \mathbf{x}_{jq} \partial \mathbf{x}_{3p}}$, $j, p, q \in \{0, 1, 2\}$, we simply take derivative of Equation 42 with respect to \mathbf{x}_{jq} , $j, q \in \{0, 1, 2\}$ and have

$$-\left(\frac{\partial^2 \Psi}{\partial \mathbf{x}_{jq} \partial \mathbf{x}_{0p}} + \frac{\partial^2 \Psi}{\partial \mathbf{x}_{jq} \partial \mathbf{x}_{1p}} + \frac{\partial^2 \Psi}{\partial \mathbf{x}_{jq} \partial \mathbf{x}_{2p}} \right) = \frac{\partial^2 \Psi}{\partial \mathbf{x}_{jq} \partial \mathbf{x}_{3p}}. \quad (43)$$

Then to compute entries of $\hat{\mathbf{H}}$, namely $\frac{\partial^2 \Psi}{\partial \mathbf{x}_{3p} \partial \mathbf{x}_{3q}}$, $p, q \in \{0, 1, 2\}$, we again take derivative of Equation 42 with respect to \mathbf{x}_{3q} , $q \in \{0, 1, 2\}$ and have

$$-\left(\frac{\partial^2 \Psi}{\partial \mathbf{x}_{3q} \partial \mathbf{x}_{0p}} + \frac{\partial^2 \Psi}{\partial \mathbf{x}_{3q} \partial \mathbf{x}_{1p}} + \frac{\partial^2 \Psi}{\partial \mathbf{x}_{3q} \partial \mathbf{x}_{2p}} \right) = \frac{\partial^2 \Psi}{\partial \mathbf{x}_{3q} \partial \mathbf{x}_{3p}}. \quad (44)$$

Thus we can represent every entry in $\tilde{\mathbf{H}}$ and $\hat{\mathbf{H}}$ through linear combination of entries in $\tilde{\mathbf{H}}$. With little bit more effort, it's not hard to show the result of Equation 21.

D DIAGONALIZATION OF $\tilde{\mathbf{H}}^*$

In order to decompose $\tilde{\mathbf{H}}^*$ into diagonal form, we first expand its entries and regroup them as

$$\begin{aligned} \frac{\partial^2 \sigma_0}{\partial x \partial y} &= \underbrace{(\omega_x^{u0} \omega_y^{u0} + \omega_x^{v0} \omega_y^{v0})}_{S_{x,y}^{0,1}} + \underbrace{(\omega_x^{u1} \omega_y^{u1} + \omega_x^{v1} \omega_y^{v1})}_{S_{x,y}^{2,0}} \sigma_0 \\ &\quad - \underbrace{(\omega_y^{u0} \omega_x^{v0} + \omega_x^{u0} \omega_y^{v0})}_{T_{x,y}^{0,1}} \sigma_1 - \underbrace{(\omega_y^{u1} \omega_x^{v1} + \omega_x^{u1} \omega_y^{v1})}_{T_{x,y}^{2,0}} \sigma_2, \\ \frac{\partial^2 \sigma_1}{\partial x \partial y} &= \underbrace{(\omega_x^{u0} \omega_y^{u0} + \omega_x^{v0} \omega_y^{v0})}_{S_{x,y}^{0,1}} + \underbrace{(\omega_x^{u2} \omega_y^{u2} + \omega_x^{v2} \omega_y^{v2})}_{S_{x,y}^{1,2}} \sigma_1 \\ &\quad - \underbrace{(\omega_y^{u0} \omega_x^{v0} + \omega_x^{u0} \omega_y^{v0})}_{T_{x,y}^{0,1}} \sigma_0 - \underbrace{(\omega_y^{u2} \omega_x^{v2} + \omega_x^{u2} \omega_y^{v2})}_{T_{x,y}^{1,2}} \sigma_2, \\ \frac{\partial^2 \sigma_2}{\partial x \partial y} &= \underbrace{(\omega_x^{u2} \omega_y^{u2} + \omega_x^{v2} \omega_y^{v2})}_{S_{x,y}^{1,2}} + \underbrace{(\omega_x^{u1} \omega_y^{u1} + \omega_x^{v1} \omega_y^{v1})}_{S_{x,y}^{2,0}} \sigma_2 \\ &\quad - \underbrace{(\omega_y^{u2} \omega_x^{v2} + \omega_x^{u2} \omega_y^{v2})}_{T_{x,y}^{1,2}} \sigma_1 - \underbrace{(\omega_y^{u1} \omega_x^{v1} + \omega_x^{u1} \omega_y^{v1})}_{T_{x,y}^{2,0}} \sigma_0, \end{aligned} \quad (45)$$

where we adopt the same notation convention in Equation 15. Notice here we use x and y to represent vertex coordinates. Then for each entry of $\tilde{\mathbf{H}}^*$, we have

$$\begin{aligned} \tilde{\mathbf{H}}_{x,y}^* &= [(S_{x,y}^{0,1} + S_{x,y}^{2,0}) \sigma_0 - T_{x,y}^{0,1} \sigma_1 - T_{x,y}^{2,0} \sigma_2] \frac{\partial \Psi}{\partial \sigma_0} \\ &\quad + [(S_{x,y}^{0,1} + S_{x,y}^{1,2}) \sigma_1 - T_{x,y}^{0,1} \sigma_0 - T_{x,y}^{1,2} \sigma_2] \frac{\partial \Psi}{\partial \sigma_1} \\ &\quad + [(S_{x,y}^{1,2} + S_{x,y}^{2,0}) \sigma_2 - T_{x,y}^{1,2} \sigma_1 - T_{x,y}^{2,0} \sigma_0] \frac{\partial \Psi}{\partial \sigma_2} \\ &= [S_{x,y}^{0,1} \sigma_0 - T_{x,y}^{0,1} \sigma_1] \frac{\partial \Psi}{\partial \sigma_0} + [S_{x,y}^{0,1} \sigma_1 - T_{x,y}^{0,1} \sigma_0] \frac{\partial \Psi}{\partial \sigma_1} \\ &\quad + [S_{x,y}^{1,2} \sigma_1 - T_{x,y}^{1,2} \sigma_2] \frac{\partial \Psi}{\partial \sigma_1} + [S_{x,y}^{1,2} \sigma_2 - T_{x,y}^{1,2} \sigma_1] \frac{\partial \Psi}{\partial \sigma_2} \\ &\quad + [S_{x,y}^{2,0} \sigma_0 - T_{x,y}^{2,0} \sigma_2] \frac{\partial \Psi}{\partial \sigma_0} + [S_{x,y}^{2,0} \sigma_2 - T_{x,y}^{2,0} \sigma_0] \frac{\partial \Psi}{\partial \sigma_2}. \end{aligned} \quad (46)$$

Thus we can separate $\tilde{\mathbf{H}}^*$ into the sum of three parts as shown in Equation 24, where each part is defined as

$$\begin{aligned} \tilde{\mathbf{H}}_{x,y}^{*0,1} &= [S_{x,y}^{0,1} \sigma_0 - T_{x,y}^{0,1} \sigma_1] \frac{\partial \Psi}{\partial \sigma_0} + [S_{x,y}^{0,1} \sigma_1 - T_{x,y}^{0,1} \sigma_0] \frac{\partial \Psi}{\partial \sigma_1}, \\ \tilde{\mathbf{H}}_{x,y}^{*1,2} &= [S_{x,y}^{1,2} \sigma_1 - T_{x,y}^{1,2} \sigma_2] \frac{\partial \Psi}{\partial \sigma_1} + [S_{x,y}^{1,2} \sigma_2 - T_{x,y}^{1,2} \sigma_1] \frac{\partial \Psi}{\partial \sigma_2}, \\ \tilde{\mathbf{H}}_{x,y}^{*2,0} &= [S_{x,y}^{2,0} \sigma_0 - T_{x,y}^{2,0} \sigma_2] \frac{\partial \Psi}{\partial \sigma_0} + [S_{x,y}^{2,0} \sigma_2 - T_{x,y}^{2,0} \sigma_0] \frac{\partial \Psi}{\partial \sigma_2}. \end{aligned} \quad (47)$$

If we take a look at one of them, like $\tilde{\mathbf{H}}_{x,y}^{*0,1}$, we have

$$\begin{aligned}
\tilde{\mathbf{H}}_{x,y}^{*0,1} &= [S_{x,y}^{0,1}\sigma_0 - T_{x,y}^{0,1}\sigma_1] \frac{\partial\Psi}{\partial\sigma_0} + [S_{x,y}^{0,1}\sigma_1 - T_{x,y}^{0,1}\sigma_0] \frac{\partial\Psi}{\partial\sigma_1} \\
&= (\omega_x^{u0}\omega_y^{u0} + \omega_x^{v0}\omega_y^{v0})\sigma_0 \frac{\partial\Psi}{\partial\sigma_0} - (\omega_y^{u0}\omega_x^{v0} + \omega_x^{u0}\omega_y^{v0})\sigma_1 \frac{\partial\Psi}{\partial\sigma_0} \\
&+ (\omega_x^{u0}\omega_y^{u0} + \omega_x^{v0}\omega_y^{v0})\sigma_1 \frac{\partial\Psi}{\partial\sigma_1} - (\omega_y^{u0}\omega_x^{v0} + \omega_x^{u0}\omega_y^{v0})\sigma_0 \frac{\partial\Psi}{\partial\sigma_1} \quad (48) \\
&= \omega_x^{u0} \left(\sigma_0 \frac{\partial\Psi}{\partial\sigma_0} + \sigma_1 \frac{\partial\Psi}{\partial\sigma_1} \right) \omega_y^{u0} - \omega_x^{u0} \left(\sigma_0 \frac{\partial\Psi}{\partial\sigma_1} + \sigma_1 \frac{\partial\Psi}{\partial\sigma_0} \right) \omega_y^{v0} \\
&- \omega_y^{u0} \left(\sigma_1 \frac{\partial\Psi}{\partial\sigma_0} + \sigma_0 \frac{\partial\Psi}{\partial\sigma_1} \right) \omega_x^{v0} + \omega_x^{v0} \left(\sigma_1 \frac{\partial\Psi}{\partial\sigma_1} + \sigma_0 \frac{\partial\Psi}{\partial\sigma_0} \right) \omega_y^{v0}.
\end{aligned}$$

Thus we know $\tilde{\mathbf{H}}^{*0,1}$ has the decomposition as shown in Equation 25. Similar derivations also apply to $\tilde{\mathbf{H}}^{*1,2}$ and $\tilde{\mathbf{H}}^{*2,0}$.