Eurographics Symposium on Geometry Processing 2026
M. Skouras and M. Livesu
(Guest Editors)

Volume 45 (2026), Number 5

Spatial Eigenanalysis of 2D Deformation Energies

Haomiao Wu[1] and Kui Wu[2] and Theodore Kim[1]

[1]Yale University, U.S.A.
[2]Lightspeed, U.S.A.

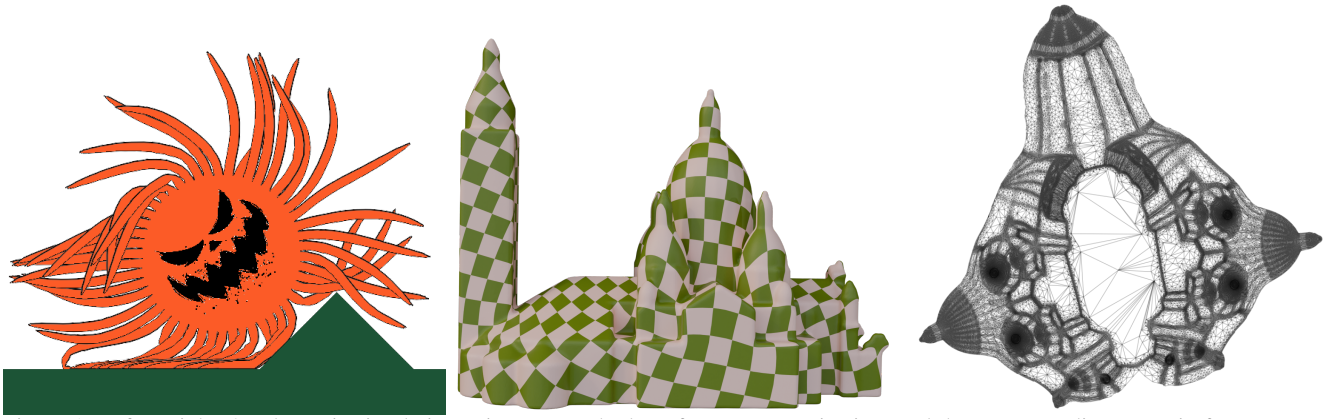


Figure 1: Left to right: 2D dynamic simulation using our method, surface parameterization, and the corresponding UV wireframe. We use our analytic spatial eigenanalysis to develop fast and robust algorithms that can be applied to diverse elasticity problems.

Abstract

We present the first closed-form, per-element eigensystems for 2D deformation energies in the primary domain of spatial coordinates. Previous analyses have only been able to find such eigensystems in proxy coordinates such as the deformation gradient, but as the spatial positions directly correspond to global degrees of freedom, our analysis opens up avenues for novel algorithms that were not possible before. First, we propose a spectral preconditioner that approximates global near-nullspace modes using local analytical eigenvectors and clusters global eigenvalues using a low-rank correction. Second, we derive a unified projected Newton formulation that combines the element eigensystem with the mass and damping terms, yielding a superior eigenvalue filter. We test our preconditioner on surface parameterization problems and our projected Newton framework on dynamic simulation examples. The results show significant efficiency gains over previous methods. In particular, our physics-aware filtering strategy reduces eigenvalue clamping by up to $30\times$*, which in turn decreases the Newton iteration count by roughly* $2\times$*.*

CCS Concepts

• ***Computing methodologies*** → *Physical simulation; Mesh geometry models;*

1. Introduction

The eigensystem that arises from an elastic energy's Hessian can reveal the exact geometric nature of the modes that an energy produces under deformation. Understanding this eigenstructure can in turn have a variety of applications. Per-element analytic eigensystems have been derived for solid [SGK19], surface tension [HGMRT20], shell [Kim20], collision penalty [SK23], and collision barrier [HCLK24] energies. In these cases, the goal is typically to project an indefinite Hessian onto positive semi-definiteness (PSD) within a Newton solver.

In a related line of inquiry, global eigensystems have been used to construct deformation [BJ05], sound synthesis [VDDKP01], and shape analysis [CBC*25] subspaces. However, the coordinate systems of these two approaches are usually mismatched. The global eigensystems are usually formulated, and numerically solved for, in terms of *spatial position*, which correspond to eigenvectors whose degrees of freedom match the vertex positions of the whole object. In contrast, the per-element eigensystems have been shown to admit clean, closed-form solutions in a proximal *deformation gradi-*

ent space, where intermediate Hessians for individual elements are filtered before expanding back to the primary spatial coordinates.

Up until now, it has not been clear if there is a similarly clean per-element eigensystem in the *primary spatial coordinates*. Determining if this is possible is the first step towards understanding the relationship between the element-wise and global eigensystems. We take that first step here, and show that there is such an eigensystem for 2D deformation energies.

These results have two immediate applications. First, we use them to construct a lightweight spectral preconditioner for conjugate gradients. Unlike methods such as multigrid [WLF*20, BHM00], additive Schwarz [WWW22] and incomplete Cholesky [CSHD21] that rely on complex setups and problem-specific assumptions, our preconditioner is easy to assemble and generally applicable to ill-posed scenarios, while still achieving roughly 2-4× speedups compared with those state-of-the-art methods and a baseline Jacobi preconditioner.

Second, we show that our position-based formulation can better preserve the structure of the original physical system under positive semi-definiteness (PSD) projection. Previous approaches were limited to filtering the Hessian of various elastic potential energies (stiffness matrix), but our spatial analysis allows us to see the combined eigenstructure of the stiffness, mass, and damping matrices, and filter accordingly.

We found that our strategy can filters up to **30**× fewer eigenvalues, avoiding the overly conservative clampings of deformation gradient-based strategies. These reduced clampings better preserve the structure of the underlying physics, resulting in a $\sim$ **2**× reduction in Newton iterations compared with previous methods [CLL*24] for dynamic simulation.

Our contributions are as follows:

- Closed-form eigensystems for a variety of 2D hyperelastic energies that can be efficiently computed at runtime.
- A spectral PCG preconditioner based on our eigenvectors.
- Improved projected Newton convergence through a unified treatment of the stiffness, mass, and damping terms.
- Improved speed and robustness over recent preconditioning [CSHD21, Dem19] and filtering [CLJ*24, CLL*24] strategies.

2. Related Works

This section provides a brief overview of elastic energy formulations, optimization methods, and their applications in simulation and surface parameterization.

Elastic Energy. The corotational elasticity model [CPSS10, MZS*11, MDM*02, NKJF09] was introduced to improve robustness under large rotational deformations. In contrast, hyperelastic material models such as Neo-Hookean [SGK18] and Saint Venant–Kirchhoff [PDA00] formulations have been widely adopted to capture large deformations with higher physical fidelity. Force-filtering approaches [ITF04, TSIF05] further improve visual quality comparable to Neo-Hookean materials with additional user-defined parameters. Analytic eigensystems have played a central role in enabling efficient and stable optimization of these energies. McAdams et al. [MZS*11] derived analytic eigenvalues for corotational elasticity, while Smith et al. [SGK18] addressed a customized Neo-Hookean energy. Subsequent work derived per-element analytic eigensystems for a broad range of energies, including solid elasticity [SGK19], surface tension [HGMRT20], shell models [Kim20], collision penalty energies [SK23], collision barrier formulations [HCLK24], and isotropic ARAP models [LCK22], offering alternative treatments of as-rigid-as-possible energies [SA07]. Related Hessian-modification ideas have also appeared in Chen et al. [CW17], and one adaption [GSC18] examines the spectrum of local spatial Hessians, albeit without a closed-form analytical formulation.

Optimization Methods. The minimization of elastic energies typically leads to nonlinear optimization problems. Projected Newton methods address this challenge by restoring the positive definiteness of the Hessian to guarantee convergence to a local minimum. Common projection strategies include clamping negative eigenvalues [TSIF05], diagonal regularization [FL16], or reflecting negative eigenvalues to their absolute values [CLL*24]. More recently, Chen et al. [CLJ*24] proposed an adaptive eigenvalue filtering strategy that stabilizes the optimization of Neo-Hookean energy and its variants within the Projected Newton framework. Alternative optimization approaches include local–global methods [BML*14], the Alternating Direction Method of Multipliers (ADMM) [OBLN17], Chebyshev extrapolation [Wan15], L-BFGS [LBK17], and multigrid methods [LYL*25]. Excellent overviews [SB12, KE22] are available.

Surface Parameterization. Surface parameterization seeks to flatten a 3D surface onto the plane while minimizing elastic distortion [FH05]. Numerous distortion-optimization techniques have been proposed, including Most-Isometric Parameterizations (MIPS) [HG99, FLG15], the local–global approach [LZX*08], Accelerated Quadratic Proxy (AQP) [KGL16], locally injective mapping [RPPSH17, SKPSH13], convex second-order approximations [SPSH*17], Progressive Parameterizations [LYNF18], and Blended Cured Quasi-Newton (BCQN) [ZBK18]. These parameterizations are often guided by cross- or frame-field directions [VCD*17, KNP07] for applications such as mesh quadrangulation [JTPSH15]. While line search, barrier functions, and related strategies allow efficient processing of large meshes and tolerate imperfect initial solutions, the final quality remains sensitive to initial injectivity, mesh structure, and boundary conditions. Comprehensive surveys [FH05] are again available.

3. Position-Based Eigenanalysis

In this section, we derive closed-form analytical eigensystems of energy Hessians with respect to vertex positions. The majority of common isotropic deformation energies can be written in scalar invariants that are functions of singular values of the deformation gradient $\mathbf{F}$. We will be using the invariants of Smith et al. [SGK19]:

$$I_1 = \mathrm{tr}\mathbf{S}, \qquad I_2 = \|\mathbf{S}\|^2, \qquad I_3 = \det\mathbf{S} \tag{1}$$

Where $\mathbf{S}$ comes from the polar decomposition $\mathbf{F} = \mathbf{RS}$, I_1 and I_2 measure shape stretching and distortion, and I_3 tracks volume. Only two out of the three invariants are independent in 2D, and the relationship between the three is given by:

$$I_1^2 = I_2 + 2I_3 \tag{2}$$

However, in later derivations we will show that the three-invariant style allows us to more cleanly locate a closed-form solution.

It is known that the deformation gradient-based Hessians of the three invariants have eigenvectors that align with each other and are sparse under the SVD frame of $\mathbf{F}$. However, our analysis shows that, unlike in $\mathbf{F}$ space, in spatial coordinates the spectra of different invariant Hessians are coupled differently.

Therefore, we will separate the eigenanalysis into two cases: shape distortion-based energies $\Psi(I_1, I_2)$ and volume preservation-based energies $\Psi(I_2, I_3)$. The three-invariant style allows us to separate into these two cases. In 2D, $\Psi(I_2, I_3)$ directly penalizes local area change, while $\Psi(I_1, I_2)$ can exhibit less lateral contraction under stretching. We are then able to study the coupling between the quadratic stretching invariant I_2 and the other two invariants separately, resulting in cleaner matrix structures. We can further utilize the fact that most elastic energies are linear to I_2 to reveal further sparsity in the corresponding Hessians. We will walk through the intuition behind the derivation that reduces the eigenproblems to clean, low-rank forms.

3.1. Background and Formulation

We will focus our analysis on 2D. We represent vectors by bold lowercase letters and matrices by bold capital letters. Matrices and their flattened vector forms will be denoted by the same letter. Let the matrix of vertex positions for one triangle be $\mathbf{X} = [\mathbf{x}_0|\mathbf{x}_1|\mathbf{x}_2] \in \mathbb{R}^{2\times 3}$ and its rest position matrix be $\bar{\mathbf{X}}$. The deformation gradient is defined as $\mathbf{F} = \partial \mathbf{x}/\partial \bar{\mathbf{x}}$. The relationship between the deformation gradient and the vertex positions can be written as:

$$\mathbf{C} = \begin{bmatrix} -1 & 1 & 0 \\ -1 & 0 & 1 \end{bmatrix} \qquad \mathbf{D}_s = \mathbf{X}\mathbf{C}^\top \qquad \mathbf{D}_m = \bar{\mathbf{X}}\mathbf{C}^\top \tag{3}$$

$$\mathbf{D} = \mathbf{D}_m^{-\top}\mathbf{C} \in \mathbb{R}^{2\times 3} \qquad \mathbf{F} = \mathbf{D}_s\mathbf{D}_m^{-1} = \mathbf{X}\mathbf{D}^\top \tag{4}$$

For clarity, we will perform our analysis using matrix and vector forms rather than tensor forms. By the chain rule, the energy Hessian with respect to positions $\mathbf{x}$ is:

$$\mathbf{H} = \frac{\partial^2 \Psi}{\partial \mathbf{x}^2} = \left(\frac{\partial \mathbf{f}}{\partial \mathbf{x}}\right)^\top \frac{\partial^2 \Psi}{\partial \mathbf{f}^2}\left(\frac{\partial \mathbf{f}}{\partial \mathbf{x}}\right) = \mathbf{G}^\top \mathbf{H}_\mathbf{f} \mathbf{G} \quad \mathbf{G} = \mathbf{D} \otimes \mathbf{I}_2 \tag{5}$$

where $\mathbf{f} = \mathrm{vec}(\mathbf{F})$, $\mathbf{I}_2$ is the 2 by 2 identity matrix, $\mathbf{H}_\mathbf{f} \in \mathbb{R}^{4\times 4}$ is the $\mathbf{F}$-based Hessian and $\mathbf{G} \in \mathbb{R}^{4\times 6}$. The $\otimes$ is the Kronecker product, which for $\mathbf{A} \in \mathbb{R}^{2\times 2}$ and $\mathbf{B} \in \mathbb{R}^{2\times 2}$ is defined as

$$\mathbf{A} = \begin{bmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{bmatrix}, \qquad \mathbf{A} \otimes \mathbf{B} = \begin{bmatrix} a_{11}\mathbf{B} & a_{12}\mathbf{B} \\ a_{21}\mathbf{B} & a_{22}\mathbf{B} \end{bmatrix}.$$

Previous work focused on the eigenanalysis of the matrix $\mathbf{H}_\mathbf{f}$, which can be computed using the invariants:

$$\mathbf{H}_\mathbf{f} = \sum_{i=1}^{3}\left(\frac{\partial^2 \Psi}{\partial I_i^2}\left(\frac{\partial I_i}{\partial \mathbf{f}}\right)\left(\frac{\partial I_i}{\partial \mathbf{f}}\right)^\top + \frac{\partial \Psi}{\partial I_i}\frac{\partial^2 I_i}{\partial \mathbf{f}^2}\right) + \sum_{i<j}\left(\frac{\partial^2 \Psi}{\partial I_i \partial I_j}\left(\frac{\partial I_i}{\partial \mathbf{f}}\right)\left(\frac{\partial I_j}{\partial \mathbf{f}}\right)\right). \tag{6}$$

However, because $\mathbf{G}$ is not orthonormal, there is no simple relationship between the eigensystems of $\mathbf{H}$ and $\mathbf{H}_\mathbf{f}$, so the spatial eigenanalysis must start from scratch.

The second term from Eq. 6 is a cross-derivative between invariants, which we omit because they evaluate to zero in the energies we will examine. Plugging $\mathbf{H}_\mathbf{f}$ into Eq. 5 then yields:

$$\mathbf{H} = \mathbf{G}^\top\left(\sum_{i=1}^{3}\frac{\partial^2 \Psi}{\partial I_i^2}\left(\frac{\partial I_i}{\partial \mathbf{f}}\right)\left(\frac{\partial I_i}{\partial \mathbf{f}}\right)^\top + \frac{\partial \Psi}{\partial I_i}\frac{\partial^2 I_i}{\partial \mathbf{f}^2}\right)\mathbf{G}. \tag{7}$$

The first term is a rank-1 outer product, where each matrix entry is the product of one component from each vector, and the second term is the invariant Hessian. For compactness, we denote the scalar derivatives as $\alpha_i = \partial^2\Psi/\partial I_i^2$, $\beta_i = \partial\Psi/\partial I_i$. The singular value decompositions of $\mathbf{F}$ and $\mathbf{D}$ are denoted as $\mathbf{F} = \mathbf{U}\Sigma\mathbf{V}$ and $\mathbf{D} = \mathbf{U}_\mathbf{D}\Sigma_\mathbf{D}\mathbf{V}_\mathbf{D}$. The singular values of $\mathbf{F}$ are then σ_0, σ_1 and those of $\mathbf{D}$ are τ_0, τ_1. Since $\mathbf{D}$ only depends on the rest positions, its SVD can be precomputed at the rest state. For convenience, we will be applying a thin SVD on $\mathbf{D}$ such that $\Sigma_\mathbf{D} \in \mathbb{R}^{2\times 2}$ and $\mathbf{V}_\mathbf{D} \in \mathbb{R}^{3\times 2}$.

Finally, $\mathbf{G}$ is a rectangular matrix with a rank-2 null space that represents the translation modes, from which we can extract the first two trivial eigenpairs:

$$\mathbf{q}_0 = \begin{bmatrix} 1 & 0 & 1 & 0 & 1 & 0 \end{bmatrix}^\top \qquad \lambda_0 = 0$$
$$\mathbf{q}_1 = \begin{bmatrix} 0 & 1 & 0 & 1 & 0 & 1 \end{bmatrix}^\top \qquad \lambda_1 = 0. \tag{8}$$

3.2. Eigensystem of Shape Distortion-Based Energies

3.2.1. Derivation

Energies that focus on shape distortion, such as ARAP [SA07] and co-rotational [MZS*11], can be expressed in terms of $\Psi(I_1, I_2)$. We further assume that the energy depends linearly on I_2, which implies that it is quadratic in stretching and $\alpha_2 = 0$. This assumption also prohibits us from using only two invariants. For example, in ARAP, using Eq. 2 to eliminate I_1, we obtain

$$\Psi_{\mathrm{ARAP}} = I_2 - 2I_1 + 2 = I_2 - 2\sqrt{I_2 + 2I_3} + 2 \tag{9}$$

However, the dependency on I_2 now becomes nonlinear and unnecessarily complicates the structure of the derivatives.

Plugging in these constraints and applying the identity $(\mathbf{A} \otimes \mathbf{B})\mathrm{vec}(\mathbf{X}) = \mathrm{vec}(\mathbf{B}\mathbf{X}\mathbf{A}^\top)$, Eq. 7 expands to:

$$\mathbf{H} = \alpha_1 \mathrm{vec}(\mathbf{U}\mathbf{V}^\top\mathbf{D})\mathrm{vec}(\mathbf{U}\mathbf{V}^\top\mathbf{D})^\top + \gamma_1 \mathrm{vec}(\mathbf{U}\mathbf{T}\mathbf{V}^\top\mathbf{D})\mathrm{vec}(\mathbf{U}\mathbf{T}\mathbf{V}^\top\mathbf{D})^\top + 2\beta_2(\mathbf{D}^\top\mathbf{D} \otimes \mathbf{I}_2) \tag{10}$$

$$\mathbf{T} = \begin{bmatrix} 0 & 1 \\ -1 & 0 \end{bmatrix} \qquad \sigma = \sigma_0 + \sigma_1 \qquad \gamma_1 = \frac{\beta_1}{\sigma} \tag{11}$$

$\mathbf{H}$ is a dense 6×6 matrix, but we can sparsify it by rotating it into the I_1 frame from the first term:

$$\begin{aligned}\hat{\mathbf{H}} &= (\mathbf{V}_\mathbf{D} \otimes \mathbf{U}\mathbf{V}^\top\mathbf{U}_\mathbf{D})^\top \mathbf{H} (\mathbf{V}_\mathbf{D} \otimes \mathbf{U}\mathbf{V}^\top\mathbf{U}_\mathbf{D}) \\ &= \alpha_1 \mathrm{vec}(\Sigma_\mathbf{D})\mathrm{vec}(\Sigma_\mathbf{D})^\top + \gamma_1 \mathrm{vec}(\mathbf{T}\Sigma_\mathbf{D})\mathrm{vec}(\mathbf{T}\Sigma_\mathbf{D})^\top \\ &\quad + 2\beta_2(\Sigma_\mathbf{D}^\top\Sigma_\mathbf{D} \otimes \mathbf{I}_2)\end{aligned} \tag{12}$$

Applying the $\mathbf{V}_\mathbf{D}$ frame removes the translation modes, yielding a

sparse 4×4 matrix:

$$\hat{\mathbf{H}} = \begin{bmatrix} (\alpha_1+2\beta_2)\tau_0^2 & 0 & 0 & \alpha_1\tau_0\tau_1 \\ 0 & (\gamma_1+2\beta_2)\tau_0^2 & -\gamma_1\tau_0\tau_1 & 0 \\ 0 & -\gamma_1\tau_0\tau_1 & (\gamma_1+2\beta_2)\tau_1^2 & 0 \\ \alpha_1\tau_0\tau_1 & 0 & 0 & (\alpha_1+2\beta_2)\tau_1^2 \end{bmatrix}$$

This matrix is block-diagonal up to permutation, and each 2×2 block can be solved analytically. The four eigenpairs of $\hat{\mathbf{H}}$ are then:

$$\lambda_{2,3} = \frac{1}{2}\left((\gamma_1+2\beta_2)\tau^2 \pm \sqrt{(\gamma_1+2\beta_2)^2\tau^4 - 16\beta_2(\gamma_1+\beta_2)\tau_0^2\tau_1^2}\right)$$

$$\mathbf{P}_{2,3} = \begin{bmatrix} 0 & (\gamma_1+2\beta_2)\tau_0^2-\lambda_{2,3} \\ \gamma_1\tau_0\tau_1 & 0 \end{bmatrix} \tag{13}$$

$$\lambda_{4,5} = \frac{1}{2}\left((\alpha_1+2\beta_2)\tau^2 \pm \sqrt{(\alpha_1+2\beta_2)^2\tau^4 - 16\beta_2(\alpha_1+\beta_2)\tau_0^2\tau_1^2}\right)$$

$$\mathbf{P}_{4,5} = \begin{bmatrix} -\alpha_1\tau_0\tau_1 & 0 \\ 0 & (\alpha_1+2\beta_2)\tau_0^2-\lambda_{4,5} \end{bmatrix} \tag{14}$$

We denote $\tau^2 = \tau_0^2+\tau_1^2$, and the eigenvectors are shown in non-normalized form.

For the case of ARAP, plugging in $\alpha_1 = 0, \beta_1 = -2, \beta_2 = 1$ yields particularly compact expressions:

$$\mathbf{P}_{2,3} = \begin{bmatrix} 0 & 2\left(1-\frac{1}{\sigma}\right)\tau_0^2-\lambda_{2,3} \\ -\frac{2\tau_0\tau_1}{\sigma} & 0 \end{bmatrix}$$

$$\lambda_{2,3} = \left(1-\frac{1}{\sigma}\right)\tau^2 \pm \sqrt{\left(1-\frac{1}{\sigma}\right)^2\tau^4 - 4(1-\frac{2}{\sigma})\tau_0^2\tau_1^2)}$$

$$\mathbf{P}_4 = \begin{bmatrix} 1 & 0 \\ 0 & 0 \end{bmatrix} \quad \lambda_4 = 2\tau_0^2 \quad \mathbf{P}_5 = \begin{bmatrix} 0 & 0 \\ 0 & 1 \end{bmatrix} \quad \lambda_5 = 2\tau_1^2 \tag{15}$$

The final eigenvectors of $\mathbf{H}$ can be obtained by rotating $\mathbf{P}_*$ back into the original frame based on Eq. 12:

$$\mathbf{Q}_i = \mathbf{U}\mathbf{V}^\top\mathbf{U}_\mathbf{D}\mathbf{P}_i\mathbf{V}_\mathbf{D}^\top \qquad i = 2,3,4,5 \tag{16}$$

The indices start from 2 in order to skip the the first two translation eigenmodes from Eq. 8.

3.2.2. Discussion and Definiteness

From these expressions, we can see that $\mathbf{P}_4$ and $\mathbf{P}_5$ are the scaling modes that align with the direction of the local force, while $\mathbf{P}_2$ and $\mathbf{P}_3$ align orthogonally. This is similar to the $\mathbf{F}$-based eigensystem in Smith et al. [SGK19], where the force frame coincides with the SVD frame of $\mathbf{F}$. The general expressions from Eq. 13 and 14 also show that the eigenmodes are scaled by the rest state singular values τ_0 and τ_1 in different directions, instead of matching the principal directions as in the $\mathbf{F}$-based analysis.

Many elastic energies are known to go indefinite when an element is compressed, which is usually related to buckling [SGK19, WK23]. Examining the expressions, we see that λ_2 is always positive and λ_3 becomes negative when $\beta_2(\gamma_1+\beta_2) < 0$. Since $\beta_2 = \partial\Psi/\partial I_2$ roughly represents the stretching stiffness, it usually holds that $\beta_2 > 0$, so $\lambda_3 < 0$ when $\beta_2 < -\beta_1/\sigma_0+\sigma_1$. For ARAP, that translates to $\sigma_0+\sigma_1 < 2$, which is when the element is compressed. In $\lambda_{4,5}$, the sign of $\alpha_1+\beta_2$ depends only on the formulation of the energy. For ARAP and related energies, $\lambda_{4,5}$ of the scaling modes are always positive. This is consistent with the $\mathbf{F}$-based case as well as Sylvester's law of inertia, given that $\hat{\mathbf{H}}$ and $\mathbf{H_f}$ are congruent for non-degenerate rest triangles.

3.3. Eigensystem of Volume Preservation-Based Energies

3.3.1. Derivation

Elastic energies like Stable Neo-Hookean (SNH) [SGK18] can be expressed by a function $\Psi(I_2, I_3)$ in 2D, while energies like Symmetric Dirichlet [SS15] and MIPS [HG99] contain cross-invariant terms that fall outside the scope of our analysis. For example, the MIPS energy is $\Psi_{\text{MIPS}} = I_2/I_3$, producing a $\partial^2\Psi/\partial I_i\partial I_j \neq 0$ term back into Eqn. 6 that must be accounted for.

Similar to §3.1, we assume that $\alpha_2 = 0$ for energies with low-order stretching. The Hessian from Eq. 7 is:

$$\mathbf{H} = \alpha_3\mathbf{g}_3\mathbf{g}_3^\top + (\mathbf{D}^\top\otimes\mathbf{I}_2)(\beta_2\mathbf{H}_2+\beta_3\mathbf{H}_3)(\mathbf{D}\otimes\mathbf{I}_2) \tag{17}$$

where $\mathbf{g}_3$ is the gradient of I_3 w.r.t. $\mathbf{x}$, while $\mathbf{H}_2$ and $\mathbf{H}_3$ are Hessians of I_2 and I_3 w.r.t. $\mathbf{f}$. They are given by $\mathbf{H}_2 = 2\mathbf{I}_4$ and

$$\mathbf{g}_3 = \text{vec}\left(\text{adj}(\mathbf{F})^\top\mathbf{D}\right) \qquad \mathbf{H}_3 = \begin{bmatrix} 0 & 0 & 0 & 1 \\ 0 & 0 & -1 & 0 \\ 0 & -1 & 0 & 0 \\ 1 & 0 & 0 & 0 \end{bmatrix} \tag{18}$$

where $\text{adj}(\mathbf{F})$ is the adjoint matrix of $\mathbf{F}$. We then apply the following transformation to sparsify $\mathbf{H}$ while removing the translation modes:

$$\begin{aligned}\hat{\mathbf{H}} &= (\mathbf{V}_\mathbf{D}^\top\otimes\mathbf{I}_2)\mathbf{H}(\mathbf{V}_\mathbf{D}\otimes\mathbf{I}_2) \\ &= \alpha_3\mathbf{h}\mathbf{h}^\top + (\Sigma_\mathbf{D}^\top\otimes\mathbf{I}_2)(\beta_2\mathbf{H}_2+\beta_3\mathbf{H}_3)(\Sigma_\mathbf{D}\otimes\mathbf{I}_2)\end{aligned} \tag{19}$$

where $\mathbf{h} = \text{vec}\left(\text{adj}(\mathbf{F})^\top\mathbf{U}_\mathbf{D}\Sigma_\mathbf{D}\right)$. The second term of Eq. 19 is

$$\begin{bmatrix} 2\beta_2\tau_0^2 & 0 & 0 & \beta_3\tau_0\tau_1 \\ 0 & 2\beta_2\tau_0^2 & -\beta_3\tau_0\tau_1 & 0 \\ 0 & -\beta_3\tau_0\tau_1 & 2\beta_2\tau_1^2 & 0 \\ \beta_3\tau_0\tau_1 & 0 & 0 & 2\beta_2\tau_1^2 \end{bmatrix}, \tag{20}$$

which can be permuted into a sparse block-diagonal matrix where the two diagonal blocks share the same two eigenvalues. The eigenpairs of the matrix are then:

$$\lambda_\pm = \beta_2\tau^2 \pm \sqrt{\beta_2^2\tau^4 - (4\beta_2^2-\beta_3^2)\tau_0^2\tau_1^2} \tag{21}$$

$$\mathbf{P}_{2,3} = \frac{1}{n_\pm}\begin{bmatrix} 0 & \lambda_\pm-2\beta_2\tau_0^2 \\ -\beta_3\tau_0\tau_1 & 0 \end{bmatrix} \tag{22}$$

$$\mathbf{P}_{4,5} = \frac{1}{n_\pm}\begin{bmatrix} \beta_3\tau_0\tau_1 & 0 \\ 0 & \lambda_\pm-2\beta_2\tau_0^2 \end{bmatrix} \tag{23}$$

$$n_\pm = \sqrt{\left(\lambda_\pm-2\beta_2\tau_0^2\right)^2+\beta_3^2\tau_0^2\tau_1^2} \tag{24}$$

For convenience, we normalize $\mathbf{P}$ using $n_\pm$. Due to eigenvalue multiplicity, the eigenvectors can be chosen freely from the rank-2 subspaces spanned by $\mathbf{P}_2, \mathbf{P}_4$ or $\mathbf{P}_3, \mathbf{P}_5$ respectively corresponding to λ_+ and λ_- . Therefore, we are able to choose one eigenvector from each subspace that is orthogonal to $\mathbf{h}$. Defining the projection of $\mathbf{h}$ on the above four eigenvectors as $\kappa_i = \mathbf{h}^\top\text{vec}(\mathbf{P}_i), i = 2,3,4,5$, we

can construct two new eigenvectors orthogonal to $\mathbf{h}$:

$$\hat{\mathbf{P}}_2 = \frac{\kappa_4\mathbf{P}_2 - \kappa_2\mathbf{P}_4}{\kappa_{24}} \qquad \hat{\mathbf{P}}_3 = \frac{\kappa_5\mathbf{P}_3 - \kappa_3\mathbf{P}_5}{\kappa_{35}} \tag{25}$$

$$\kappa_{24} = \sqrt{\kappa_2^2 + \kappa_4^2}, \qquad \kappa_{35} = \sqrt{\kappa_3^2 + \kappa_5^2} \tag{26}$$

$\hat{\mathbf{P}}_2$ and $\hat{\mathbf{P}}_3$ are not affected by the rank-1 update of the first term in Eq. 19, so they are already eigenvectors of $\hat{\mathbf{H}}$. The remaining two eigenvectors live in the orthogonal complement of $\hat{\mathbf{P}}_2$ and $\hat{\mathbf{P}}_3$, and we have found that a convenient basis is:

$$\mathbf{Y} = \left[\mathrm{vec}\left(\tfrac{\kappa_2\mathbf{P}_2+\kappa_4\mathbf{P}_4}{\kappa_{24}}\right) \mid \mathrm{vec}\left(\tfrac{\kappa_3\mathbf{P}_3+\kappa_5\mathbf{P}_5}{\kappa_{35}}\right)\right]. \tag{27}$$

By restricting $\hat{\mathbf{H}}$ to this subspace,

$$\mathbf{Y}^\top\hat{\mathbf{H}}\mathbf{Y} = \begin{bmatrix} \lambda_+ + \alpha_3\kappa_{24}^2 & \alpha_3\kappa_{24}\kappa_{35} \\ \alpha_3\kappa_{24}\kappa_{35} & \lambda_- + \alpha_3\kappa_{35}^2 \end{bmatrix} \tag{28}$$

we can solve the 2D problem to yield the updated eigenvalues:

$$\begin{aligned}\eta_\pm &= \left(\beta_2\tau^2 + \frac{\alpha_3}{2}\|\mathbf{h}\|^2\right) \\ &\pm \left(\sqrt{(\beta_2\tau^2 + \frac{\alpha_3}{2}\|\mathbf{h}\|^2)^2 - \left((4\beta_2^2 - \beta_3^2)\tau_1^2\tau_2^2 + \omega\right)}\right) \\ \omega &= \alpha_3(\lambda_-\kappa_{24}^2 + \lambda_+\kappa_{35}^2).\end{aligned} \tag{29}$$

The red terms are added by the rank-1 perturbation along $\mathbf{h}$, compared with $\lambda_\pm$ from Eq. 21. The corresponding eigenvectors are:

$$\hat{\mathbf{P}}_{4,5} = \mathbf{Y}\begin{bmatrix} -\alpha_3\kappa_{24}\kappa_{35} \\ \lambda_+ + \alpha_3\kappa_{24}^2 - \eta_\pm \end{bmatrix} \tag{30}$$

Applying the rotation back, we have the final eigenpairs of $\mathbf{H}$:

$$\mathbf{Q}_i = \hat{\mathbf{P}}_i\mathbf{V}_\mathbf{D}^\top \qquad i = 2,3,4,5 \tag{31}$$

$$\lambda_2 = \lambda_+ \qquad \lambda_3 = \lambda_- \qquad \lambda_4 = \eta_+ \qquad \lambda_5 = \eta_- \tag{32}$$

3.3.2. Discussion and Definiteness

Similar to the eigensystem of $\Psi(I_1, I_2)$, we again have two eigenvectors orthogonal to the gradient. But for $\Psi(I_2, I_3)$ the eigenvectors do not conveniently become sparse in the SVD frame of $\mathbf{h}$.

By Eq. 21, λ_- turns negative when $4\beta_2^2 < \beta_3^2$. For SNH, we have:

$$\alpha_3 = \lambda \qquad \beta_2 = \frac{\mu}{2} \qquad \beta_3 = \lambda\left(I_3 - 1 - \frac{\mu}{\lambda}\right) \tag{33}$$

Where μ and λ are the Lamé parameters, E_Y is Young's modulus Poisson's ratio is ν. In this case, the condition for $\lambda_- < 0$ is $I_3 < 1$ or $I_3 > 1 + {}^{(1-2\nu)}/_{\nu}$, which is when the volume is compressed or stretched beyond a threshold. For the eigenpairs along the gradient direction $\mathbf{h}$ from Eq. 29, they are less likely to go negative because of the added ω, which matches the intuition that stretching modes are safer.

4. Application to Numerical Solvers

4.1. Preliminaries

Elasticity can be described by a constrained energy minimization:

$$\mathbf{v} = \arg\min_{\mathbf{v}} E(\mathbf{v}) \qquad \text{s.t.} \quad c(\mathbf{v}) = 0 \tag{34}$$

where $\mathbf{v}$ is the vertex positions, $E(\mathbf{v})$ is the objective function that incorporates potentials like elasticity, inertia, and externalities, and $c(\mathbf{v})$ is a function that includes kinematic or contact constraints. Since $E(\mathbf{v})$ is nonlinear for most elastic energies, Newton's method is a popular solution approach. An unconstrained and linearized formulation with dynamics at each Newton step can be given by:

$$\left(\frac{1}{\Delta t^2}\mathbf{M} + \frac{1}{\Delta t}\mathbf{C} + \mathbf{K}(\mathbf{v})\right)\Delta\mathbf{v} = -\mathbf{M}\ddot{\mathbf{v}} + \phi_{\text{int}} + \phi_{\text{ext}} - \mathbf{C}\dot{\mathbf{v}} \tag{35}$$

where $\mathbf{M}$ is the mass matrix, $\mathbf{C}$ is the non-linear Rayleigh damping matrix, and ϕ_{ext} is the external force. Given the elastic energy $\Psi(\mathbf{v})$, $\mathbf{K} = \nabla^2\Psi(\mathbf{v})$ is the global stiffness matrix and $\phi_{\text{int}} = -\nabla\Psi(\mathbf{v})$ is the internal force. Fixed vertices and rigid-body contacts can be added using constraint filtering [BW98]. We denote left-hand side system matrix as $\mathbf{A}$, and the right-hand side target as $\mathbf{b}$.

For each Newton step, PCG is one of the most commonly used linear system solvers, but its convergence is only guaranteed when the matrix is semi-positive definite (SPD). Using the analytical expressions from the previous section, we can employ the new spatial eigensystems in lieu of $\mathbf{F}$-based projection.

Since the spatial per-element Hessians are directly added into the global matrix, the local eigenvectors directly correspond to global degrees of freedom (DoF). We will show that this property can be exploited to integrate local stiffness-matrix eigensystems with other terms in the system matrix or in the global spectrum, and to construct lightweight algorithms that accelerate the solver.

4.2. A Spectral Preconditioner for PCG

In a problem with n vertices and m elements, suppose the dimension is d, and the element size is s. Given a linear system $\mathbf{Av} = \mathbf{b}$ and a preconditioner matrix $\mathbf{B}$ with dn degrees of freedom, PCG solves for $\mathbf{B}^{-1}\mathbf{Av} = \mathbf{B}^{-1}\mathbf{b}$. The convergence rate of PCG is governed by the condition number of $\mathbf{A}$, and a well-tailored preconditioner can significantly reduce the number of iterations. We propose a novel spectral preconditioner that can be readily integrated with our analytic eigensystem.

4.2.1. Spectral Preconditioning via Low-Rank Update

We base our method on the classical spectral preconditioners via low-rank update [CDG03], as they generally and tolerate inexact eigenvectors. The inverse of the preconditioner can be written as:

$$\mathbf{B}^{-1} = \mathbf{B}_0^{-1} + \mathbf{Z}(\mathbf{N}^{-1} - \mathbf{Z}^\top\mathbf{B}_0^{-1}\mathbf{Z})\mathbf{Z}^\top \qquad \mathbf{N} = \mathbf{Z}^\top\mathbf{AZ} \tag{36}$$

where $\mathbf{B}_0$ is a base conditioner. $\mathbf{Z} \in \mathbb{R}^{dn\times k}$ is the approximate basis of k eigenvectors with the lowest eigenvalues. This preconditioner enforces a prescribed inverse on the subspace spanned by the basis $\mathbf{Z}$. The low-eigenvalue modes that ruin the conditioning are isolated and handled explicitly by precomputing the low-dimensional inverse of $\mathbf{N}$. In other directions, it will behave like the base preconditioner $\mathbf{B}_0$.

4.2.2. Global Eigenvector Estimation

Spectral preconditioners reach high efficiency when they well-approximate the global spectrum, but have been less popular in

graphics because of the high cost of computing a global numerical eigendecomposition. Our analysis gives abundant information on local structures, which we now aim to map them to the global space.

Let the eigenvector matrix of the per-element Hessian for the i-th element be $\Gamma_i \in \mathbb{R}^{ds\times ds}$ and its j-th column be $\mathbf{q}_j^i$. Let $\mathbf{J}_i \in \mathbb{R}^{dn\times ds}$ be the scatter matrix with binary entries that maps local vectors of element i to the corresponding global DoFs. Without loss of generality, we assume that there exists a mapping from the collection of local eigenvectors to the smallest eigenvalue basis $\mathbf{Z}$:

$$\mathbf{Z}^{=}\Phi(\{\Gamma_i\}) \qquad i \qquad = 1,2,3,\cdots,m \tag{37}$$

Φ is highly non-linear, with no known closed-form expression. Since we are estimating $\mathbf{Z}$, we can approximate Φ using a linear operator. Grouping by element, we formulate the approximation:

$$\mathbf{Z} \approx \tilde{\mathbf{Z}} = \mathcal{S}\sum_{i=1}^{m} \mathbf{J}_i\mathbf{L}_i(\Gamma_i\Omega_i) \tag{38}$$

where $\Omega_i \in \mathbb{R}^{ds\times k}$ is the main linear mapping matrix. The k columns of $(\Gamma_i\Omega_i)$ are the local contribution of the i-th element to the k lowest eigenvalue vectors in $\mathbf{Z}$. Each column is a linear combination of the local eigenvectors $\{\mathbf{q}_j^i\}_{i=0}^{ds-1}$ with coefficients given by the column of Ω_i. Before adding to global entries by applying $\mathbf{J}_i$, $\mathbf{L}_i$ is a diagonal matrix that weights the vertex-wise contribution per local DoF. $\mathcal{S}$ is the standard matrix [AB03] of constraint filters.

Our task now is to estimate the coefficients from $\mathbf{L}_i$ and Ω_i to approximate the global eigenvectors $\mathbf{Z}$ from the local $\mathbf{q}_i^j$. In practice, we perform one numerical eigensolve using an iterative eigensolver to obtain $\mathbf{Z}_0$ at the beginning of the first Newton step or at the initial state. Next, we compute the coefficients of Ω_i by first extracting the local contribution directly using the constant scatter matrix $\mathbf{J}_i$ and subsequently projecting it to the local eigenspace using Γ_i:

$$\Omega_i = \Gamma_i^\top \mathbf{J}_i^\top \mathbf{Z}_0 \tag{39}$$

Because of overlaps between elements, we average the contributions by vertex degree before summing. This ensures that $\mathbf{Z}_0$ can be recovered exactly, and global translations will only be composed of local translation modes. Specifically, let the vector of vertex degrees be $\xi \in \mathbb{R}^{dn}$, where the value of one vertex is repeated across the d dimensions. We define the local weight matrix as $\mathbf{L}_i = (\mathbf{J}_i)^\top \mathrm{diag}(\xi)^{-1}\mathbf{J}_i$. Plugging into Eq. 38 gives:

$$\tilde{\mathbf{Z}} = \mathrm{diag}(\xi)^{-1}\mathcal{S}\sum_{i=1}^{m} \mathbf{J}_i(\Gamma_i\Omega_i) \tag{40}$$

This is evaluated at each Newton step with the updated local eigenvectors Γ_i. Following that, $\tilde{\mathbf{Z}}$ can be optionally orthonormalized. Finally, we plug $\tilde{\mathbf{Z}}$ into Eq. 36 to set up the preconditioner $\mathbf{B}$. This completes our preconditioner design.

4.3. A Unified Projected Newton

Next we will design an improved filtering method for dynamics simulations. Expanding the non-linear damping and stiffness term from Eq. 35, the system matrix of a Newton step is:

$$\mathbf{A} = \left(\frac{1}{\Delta t^2} + \frac{\alpha_{\text{damp}}}{\Delta t}\right)\mathbf{M} + \frac{\beta_{\text{damp}}}{\Delta t}\mathbf{K}(\mathbf{v}) + \mathbf{K}(\mathbf{v}) \tag{41}$$

$$\mathbf{K}(\mathbf{v}) = \sum_{i=1}^{m} \mathcal{A}_i\mathbf{J}_i\mathbf{H}_i(\mathbf{v})\mathbf{J}_i^\top = \sum_{i=1}^{m} \mathcal{A}_i\mathbf{J}_i\mathbf{G}_i^\top(\mathbf{H}_\mathbf{f})_i\mathbf{G}_i\mathbf{J}_i^\top \tag{42}$$

where α_{damp} and β_{damp} are the Rayleigh damping coefficients and $\mathcal{A}_i$ is the rest volume of element i. Projected Newton's method seeks ways to project the stiffness matrix $\mathbf{K}(\mathbf{v})$ to the SPD space. Some strategies filter the eigenvalues of the per-element $\mathbf{F}$-based Hessian $\mathbf{H}_\mathbf{f}$ by either clamping them to zero [TSIF05] or taking their absolute values [CLL*24]. These methods often lead to unnecessary filtering, because they are not able to account for the regularization introduced by the mass and damping matrices. Projection using the global spectrum would also be theoretically closer to optimal, but would require an extremely costly computation.

We make the common lumped mass assumption that yields a diagonal matrix that accumulates element mass at each vertex. Expanding and combining $\mathbf{M}$ and $\mathbf{K}(\mathbf{v})$ gives:

$$\begin{aligned}\mathbf{A} &= \left(\frac{1}{\Delta t^2} + \frac{\alpha_{\text{damp}}}{\Delta t}\right)\sum_{i=1}^{m}\mathbf{J}_i\frac{\rho\mathcal{A}_i}{s}\mathbf{I}_{ds}\mathbf{J}_i^\top + \sum_{i=1}^{m}\left(1+\frac{\beta_{\text{damp}}}{\Delta t}\right)\mathcal{A}_i\mathbf{J}_i\mathbf{H}_i\mathbf{J}_i^\top \\ &= \sum_{i=1}^{m}\left(1+\frac{\beta_{\text{damp}}}{\Delta t}\right)\mathcal{A}_i\mathbf{J}_i\left(\omega\mathbf{I}_{ds}+\mathbf{H}_i\right)\mathbf{J}_i^\top \\ \omega &= \frac{\rho\left(1+\alpha_{\text{damp}}\Delta t\right)}{s\Delta t\left(\Delta t+\beta_{\text{damp}}\right)}\end{aligned} \tag{43}$$

where $\mathbf{I}_{ds}$ is an identity matrix the size of the local DoFs ds and ρ is the material density. Given the eigendecomposition $\mathbf{H}_i = \Gamma_i\Lambda_i\Gamma_i^\top$, the eigensystem of the middle term $(\omega\mathbf{I}_{ds}+\mathbf{H}_i)$ has the same eigendecomposition except that the eigenvalues are shifted by a positive offset ω. This is not achievable by the $\mathbf{F}$ based analysis on $\mathbf{H}_\mathbf{f}$ due to the non-orthonormal $\mathbf{G}_i$.

Using our spatial eigenanalysis, we can now filter the combined stiffness, mass, and damping terms simultaneously. Instead of filtering the negative eigenvalues during the assembly of $\mathbf{K}$, we propose a new strategy that filters them with respect to ω. For each element, we filter according to:

$$\lambda_i = \begin{cases}\lambda_i, & \lambda_i \geq -\omega \\ -\omega, & \lambda_i < -\omega\end{cases} \tag{44}$$

We show in §5 that this strategy substantially reduces the number of local eigenpair filters required, leading to a filtered matrix that is substantially closer to the original matrix, leading to improved stability, convergence, and performance of Projected Newton.

5. Implementation and Results

5.1. Implementation Details

The pseudo code for our pipeline is in Alg. 1. We use our spatial eigenanalysis from §3 and filtering strategy from Eq.44 to build the local Hessians, where the clamping threshold ω is a constant computed by Eq. 43. The SVD of $\mathbf{D}$ can be pre-computed at mesh initialization, as well as the vertex degree vector ξ.

We also validate and compare our analytical eigensystems

Algorithm 1: Newton Solver with Spectral Preconditioner

while *step* $<$ *maxStep* ***and*** *residual* $>$ *tolerance* **do**
  `// System assembly`
1 $\{\mathbf{H}_i, \Gamma_i\} \leftarrow$ computeLocalClampedHessian(ω)
2 $\mathbf{K} \leftarrow$ assembleGlobalStiffnessFromLocal($\{\mathbf{H}_i\}$)
3 $\mathbf{A}, \mathbf{b} \leftarrow$ buildSystem()
4 $\mathbf{B}_0 \leftarrow$ buildBasePreconditioner($\mathbf{A}$)
  `// Preconditioner construction`
5 **if** *step* $== 0$ **then**
6 $\quad \mathbf{Z}_0 \leftarrow$ numericalGlobalEigensolve()
7 $\quad \mathbf{Z} \leftarrow \mathbf{Z}_0$, $\Omega_i \leftarrow \Gamma_i^\top \mathbf{J}_i^\top \mathbf{Z}_0$
8 **end**
9 **else**
10 $\quad \mathbf{Z} \leftarrow \mathrm{diag}(\xi)^{-1} \mathcal{S} \sum_{i=1}^{m} \mathbf{J}_i(\Gamma_i \Omega_i)$
11 **end**
12 PCGSolver.buildSpectralPreconditioner($\mathbf{Z}, \mathbf{B}_0, \mathbf{A}$)
  `// Linear solve`
13 PCGSolver.solve($\mathbf{A}$, $\mathbf{b}$)
14 updateResidual()
15 *step* $\leftarrow$ *step* $+ 1$
end

against 6×6 numerical eigensolves. While our preconditioner and projected Newton strategy can be applied in the absence of analytical eigensystems, our experiments show that our analytical approach is roughly **4× faster** than a direct numerical eigendecomposition, and significantly reduces system assembly time.

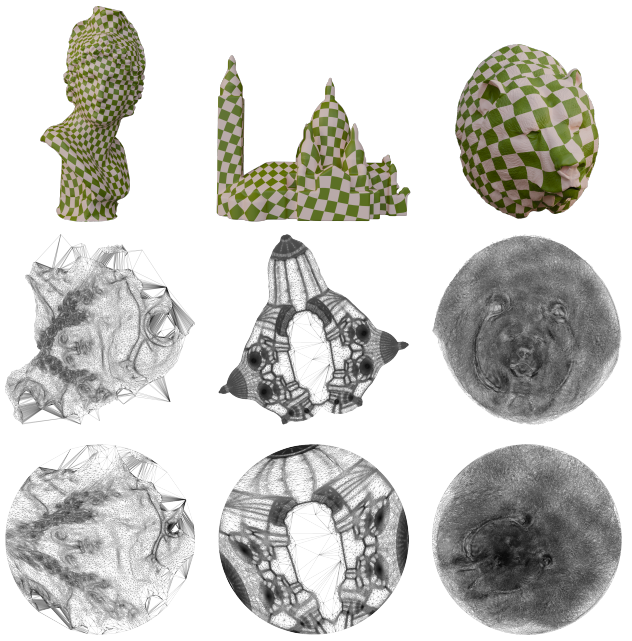
At the first Newton step, we use the shift-invert solver `Spectra::SparseSymShiftSolve` [QGN15] to compute the eigenvectors with the smallest $k = 13$ eigenvalues. Multiplications with scatter matrices $\mathbf{J}_i$ are done implicitly by visiting the entries specified by the vertex indices of each element in the global matrix. When constructing the preconditioner matrix $\mathbf{B}$, Eq. 36 should never be evaluated explicitly, as it involves dense matrix multiplication and an inverse. Instead, we store $\mathbf{Z}$ and $\mathbf{B}_0^{-1}$. Depending on the type of the base preconditioner, $\mathbf{B}_0^{-1}$ might also be stored implicitly. The inverse of the system matrix $\mathbf{A}$ on the deflated basis $\mathbf{N}^{-1}$ can be stored as its Cholesky factorization. During each PCG iteration, $\mathbf{B}^{-1}$ is applied per-component with the residual vector from right to left, avoiding a dense multiplication. Since these operations are repeated at each PCG iteration, keeping them optimized and minimal can significantly improve efficiency.

5.2. Parameterization

Surface parameterization provides a representative quasi-static 2D elasticity benchmark to evaluate our spectral preconditioner.

5.2.1. Setup

We solve single-patch, free-boundary parameterization problems with an open boundary or a given cut. To initialize the $\mathbf{D}_m$ matrices, we construct a local two-dimensional reference frame embedded in $\mathbb{R}^3$. The edge vectors are then projected onto this local tangent basis to form $\mathbf{D}_m \in \Re^{2\times 2}$, which encodes the geometry in the local intrinsic metric. The 2D vertex positions are initialized using harmonic parameterization with the boundary projected to a circle. To solve with zero-inertia and a free boundary, we remove the matrix null space by fixing three arbitrary degrees of freedom. In our examples, we use ARAP for the energy minimization and 2×2 block-Jacobi as the base preconditioner $\mathbf{B}_0$.



(a) Bust, 72k Faces (b) Building, 204k Faces (c) Bear, 516k Faces

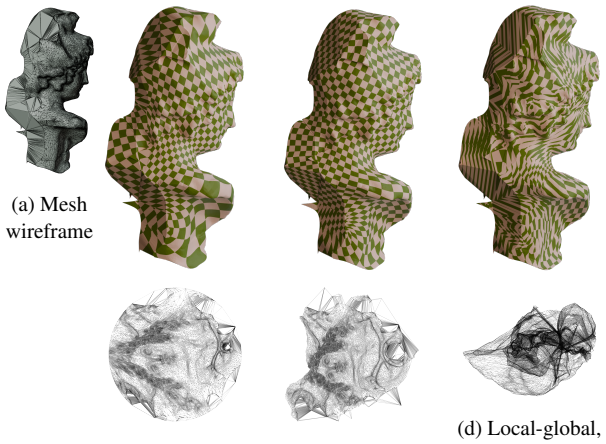
Figure 2: Surface parameterization results using our method. The top line shows the textured 3D meshes with our computed UV. The middle line shows the planar UV distribution of our output, and the third line shows the initial guess obtained from harmonic parameterization. The number of faces increases from left to right. The left two models are from the Thingi10K dataset [ZJ16]

5.2.2. Results and Comparison

We evaluate our method on three meshes exhibiting features such as irregular triangle distributions and ill-conditioned elements. The results are visualized in Fig. 2, where our method converged to low-deformation configurations. The pre-convergence middle steps are shown in the supplementary video. Local-global methods are also a popular elasticity solver [LZX*08], so we show the comparison between our Newton solver and the *libigl* implementation of the local-global-based ARAP solver on the example in Fig. 3. Even given twice the runtime, the local-global solver struggles to converge. This is expected given the mesh's large, ill-conditioned region, as local-global solvers are known to perform poorly on irregular meshes and are particularly vulnerable to inverted elements that can spread instability. In contrast, our method robustly solves a global non-linear energy minimization problem. More advanced parameterization frameworks such as Progressive Parameterizations [LYNF18] and Composite Majorization [SPSH*17] can improve robustness through continuation strategies or surrogate majorization. As our parameterization examples are intended as a test

	Ours			Block-Jacobi		AMGCL	
	Precomp.	Construct	Total	Time (s)	Our Speedup	Time (s)	Our Speedup
Bust	8.6%	2.3%	11.5s	24.7	**2.15**×	34.3	**2.98**×
Building	6.5%	1.9%	47.9s	152.2	**3.18**×	181.2	**3.78**×
Bear	5.4%	1.5%	390.0s	870.2	**2.23**×	1023.9	**2.63**×

Table 1: Wall-clock timings of our method, Block-Jacobi, and AMGCL. **Precomp.**: one-time global numerical eigensolve for $\mathbf{Z}_0$ and Ω_i fitting. **Construct**: total preconditioner construction time over all Newton steps. Both are presented as a total runtime percentage of our's method. **Our Speedup**: other methods' runtime divided by our runtime..



(a) Mesh wireframe (b) Initial guess (c) Ours, 11.5s (d) Local-global, 21.9s

Figure 3: Comparison with local-global ARAP. The leftmost wireframe shows the irregular triangles on the back. The initial guess is obtained via a harmonic parameterization. The local-global solver failed to converge, even when given almost twice the time.

bed for our Newton solver, we do not compare our visual results against these alternative strategies.

We numerically compute the exact $\mathbf{Z}$ at each Newton step and compute the relative error of the subspace $e = \|\tilde{\mathbf{Z}}\tilde{\mathbf{Z}}^\top - \mathbf{Z}\mathbf{Z}^\top\| / \|\mathbf{Z}\mathbf{Z}^\top\|$. In our examples, $e \approx 0.3$, which is expected since our basis is more aware of the local-spectrum than the global eigenspace. For preconditioning, the basis is only intended to provide low-rank corrections, and our experiments show that it effectively reduces the the system matrix's condition number by $\sim 100\times$.

We compare against other preconditioners in Figs. 4 and 5 using the base preconditioner $\mathbf{B}_0$ of Block-Jacobi, and the algebraic multigrid preconditioner from AMGCL [Dem19]. We use the true residual for both the convergence criteria and the plots. The PCG tolerance and Newton relative tolerance were set to $1e^{-5}$. The residual with our preconditioner decreases steadily on a logarithmic scale, whereas the block-Jacobi method stagnates and fluctuates. The right column shows that even beyond the initial Newton step, where the low-eigenvalue basis $\mathbf{Z}$ is computed accurately using a numerical solver, the performance gain with our estimated eigenbasis persists.

AMGCL exhibits a convergence rate similar to that of our method. However, shown in Table 1, the wall-clock time of AMGCL is slower than even block-Jacobi due to the overhead of applying the preconditioner. This is likely because AMG methods rely on elliptic PDE properties, such as strong local coupling and smooth errors, which may not hold in parameterization problems. Using AMGCL as the base preconditioner $\mathbf{B}_0$ is also infeasible, as applying it will dominate the runtime. By comparison, our algorithm is efficient and more lightweight.

We also ran our examples with a recent incomplete Cholesky (IC) preconditioner [CSHD21], but it failed due to repeated pivot breakdowns at supernodes during the factorization stage. This is a well-known limitation of IC preconditioners under large deformations [CSHD21, GVL13], and further underscores that our method does not depend on problem-specific assumptions or well-conditioned spatial structures.

Overall, we observed a more than 2× reduction in total runtime compared with block-Jacobi and a 2×-3× reduction compared with AMGCL.

5.3. Dynamic Simulation

We test our unified projected Newton method on a dynamic simulation example with a hairy ball and a square with holes (cheese) dropped onto obstacles. Fig. 6 shows the intricate structure of the meshes. The thin triangles and the wide range of element sizes make the system matrices ill-conditioned enough to stress test our solver. We use ARAP and stable Neo-Hookean (SNH) on a fixed time-step Newton solver with a tolerance of $1e^{-10}$ and $\Delta t = 1/30$s. The material density ρ is set to 1.0 and the damping parameters are $\alpha_{\text{damp}} = \beta_{\text{damp}} = 1e^{-3}$. We use material parameters $E_Y = 100, \nu = 0.45$ and solve with 8 substeps for ARAP and 20 for SNH on the hairy ball example. On the cheese example, we use substeps = 2, $E_Y = 100$ for ARAP and substeps = 5, $E_Y = 50$ for SNH. Poisson's ratio is set to 0.48. We choose these parameters to balance computational efficiency and visual quality while producing sufficiently challenging test cases for our solver. Kinematic collisions are implemented using constraint filtering [BW98], and self-collisions are turned off. The results are compared with **F**-based projection, using both the clamp-to-0 [TSIF05] and absolute value [CLL*24] filtering strategy.

In Fig. 7c, our eigenvalue clamping rule (Eq. 44) filters much fewer negative eigenvalues compared to the **F**-based method with up to 30× reductions. This indicates that the traditional **F**-based filtering can introduce substantial deviations from the original physics that, in the end, do not contribute to solver stability. Our projected system matrix is much closer to the original matrix, which can improve Newton convergence by roughly 2× (Fig. 7a).

In the first few frames, the models fall under gravity nearly undeformed. Our position-based unified clamping strategy especially

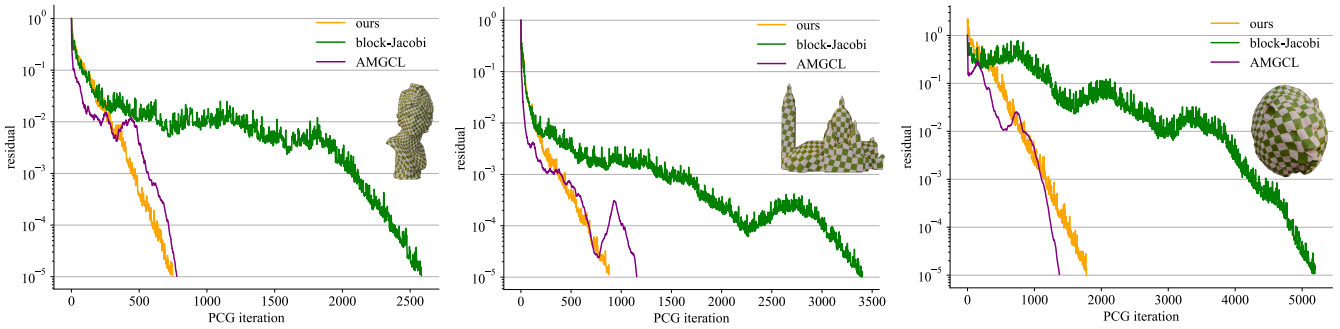


Figure 4: PCG convergence curves on surface parameterization examples of a bust, building, and bear. Convergence in shown for a single Newton step, where the residual is defined as $\|\mathbf{Av}-\mathbf{b}\|/\|\mathbf{b}\|$. Even when AMGCL converges faster, because its overhead is higher, our approach is more than twice as fast.

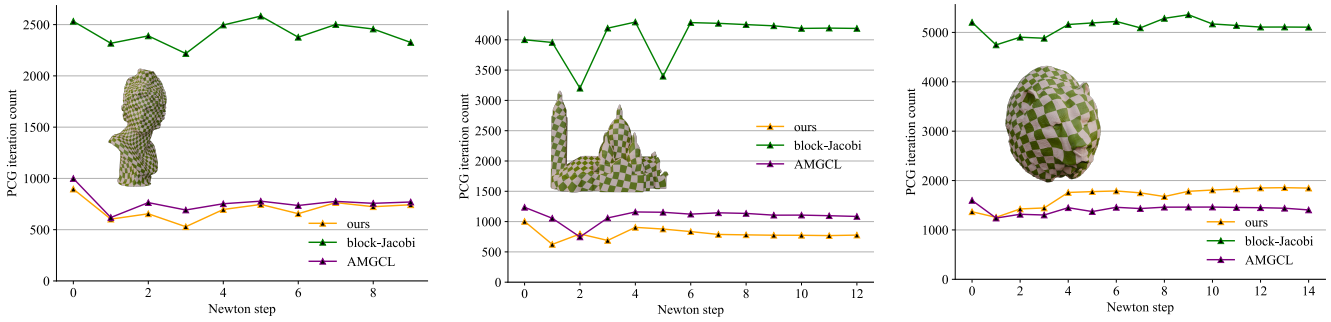


Figure 5: PCG iteration counts on surface parameterization examples of a bust, building, and bear. Similar to Fig. 4, even when AMGCL has lower iteration counts, because its overhead is higher, our approach is more than twice as fast.

outperforms the **F**-base under severe and nonuniform deformation. The advantage is more prominent on more non-linear energy, such as SNH, and more complicated geometries, such as the hairy ball. As the condition number of the system matrix increases, the clamped eigenvalue count is up to orders of magnitude smaller than **F**-based. We selected one time step in Fig. 7b that shows the superior convergence of our method. Table 2 shows the total solver speedup.

Absolute value filtering resulted in slower performance than the baseline clamp-to-0 method. This is expected because the method [CLL*24] is designed for quasi-statics, whereas we are simulating dynamics. Our performance persists even when averaged over all time frames. The animations are in the supplementary video.

6. Conclusions and Future Work

We have shown analytic spatial eigensystems can be obtained for a variety of 2D elastic energies. We use this system to design a fast spectral preconditioner for PCG, and a robust projected Newton approach. The results on parameterization and 2D dynamic simulation present consistent performance improvements. We provide only closed-form results in 2D, and generalizing to 3D is left to future work. Our analysis also assumes that the energy is linear in I_2 and contains no cross-invariant terms, but extending it to the non-linear regime would allow this approach to be applied to Mooney-Rivlin [Moo40, Riv48], Arruda-Boyce [AB93], Symmetric Dirichlet [SS15], and MIPS [HG99] energies.

Finally, with per-element spatial eigensystems in hand, we can begin asking how they relate to the global spatial eigensystem.

References

[AB93] ARRUDA E. M., BOYCE M. C.: A three-dimensional constitutive model for the large stretch behavior of rubber elastic materials. *Journal of the Mechanics and Physics of Solids 41*, 2 (1993), 389–412. 9

[AB03] ASCHER U. M., BOXERMAN E.: On the modified conjugate gradient method in cloth simulation. *The Visual Computer 19*, 7 (2003), 526–531. 6

[BHM00] BRIGGS W. L., HENSON V. E., MCCORMICK S. F.: *A multigrid tutorial*. SIAM, 2000. 2

[BJ05] BARBIČ J., JAMES D. L.: Real-time subspace integration for st. venant-kirchhoff deformable models. *ACM transactions on graphics (TOG) 24*, 3 (2005), 982–990. 1

[BML*14] BOUAZIZ S., MARTIN S., LIU T., KAVAN L., PAULY M.: Projective dynamics: fusing constraint projections for fast simulation. *ACM Transactions on Graphics (TOG) 33*, 4 (2014), 1–11. 2

[BW98] BARAFF D., WITKIN A.: Large steps in cloth simulation. In *Proceedings of the 25th Annual Conference on Computer Graphics and Interactive Techniques* (New York, NY, USA, 1998), SIGGRAPH '98, Association for Computing Machinery, p. 43–54. 5, 8

[CBC*25] CHANG Y., BENCHEKROUN O., CHIARAMONTE M. M., CHEN P. Y., GRINSPUN E.: Shape space spectra. *ACM Transactions on Graphics (TOG) 44*, 4 (2025), 1–16. 1

[CDG03] CARPENTIERI B., DUFF I. S., GIRAUD L.: A class of spectral two-level preconditioners. *SIAM Journal on Scientific Computing 25*, 2 (2003), 749–765. 5

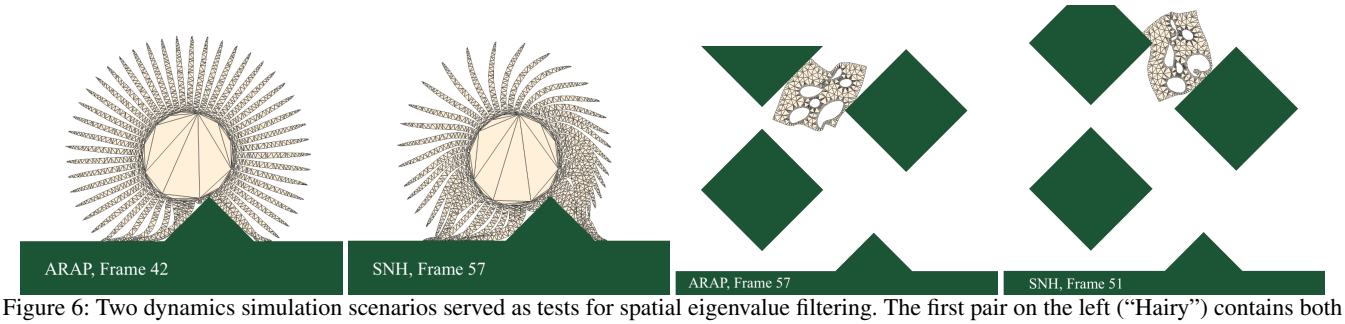


Figure 6: Two dynamics simulation scenarios served as tests for spatial eigenvalue filtering. The first pair on the left ("Hairy") contains both large and small elements, creating a challenging conditioning scenario. The second ("Cheese") undergoes challenging kinematic collisions. We tested using both the SNH [SGK18] and ARAP [SA07] elastic energies.

	Energy	Clamp Reduction	Reduction over **F**-based 0 Steps	Reduction over **F**-based Abs. Steps	Time (s) Ours	**F**-based 0	**F**-based Abs.	Our Speedup over **F**-based 0	**F**-based Abs.
Hairy	ARAP	**30.42**×	1.81×	2.73×	25.7	47.1	67.8	**1.83**×	**2.64**×
	SNH	**20.0**×	1.99×	2.90×	60.8	123.6	172.8	**2.03**×	**2.84**×
Cheese	ARAP	**2.07**×	1.96×	2.26×	7.7	16.0	17.9	**2.08**×	**2.33**×
	SNH	**5.01**×	1.63×	2.70×	25.8	46.4	73.0	**1.80**×	**2.83**×

Table 2: Statistics of our method vs. **F**-based projection using both the clamp-to-0 and absolute value strategies. **Clamp**: clamped eigenvalue count across all frames. **F**-based divided by Ours. **Steps**: average Newton step count throughout all frames. **F**-based divided by Ours. The last two columns are the percentage speedup in total solver runtime.

[CLJ*24] CHEN H., LIU H.-T. D., JACOBSON A., LEVIN D. I., ZHENG C.: Trust-region eigenvalue filtering for projected newton. In *SIGGRAPH Asia 2024 Conference Papers* (New York, NY, USA, 2024), SA '24, Association for Computing Machinery. 2

[CLL*24] CHEN H., LIU H.-T. D., LEVIN D. I., ZHENG C., JACOBSON A.: Stabler neo-hookean simulation: Absolute eigenvalue filtering for projected newton. In *ACM SIGGRAPH 2024 Conference Papers* (New York, NY, USA, 2024), SIGGRAPH '24, Association for Computing Machinery. 2, 6, 8, 9, 11

[CPSS10] CHAO I., PINKALL U., SANAN P., SCHRÖDER P.: A simple geometric model for elastic deformations. *ACM Trans. Graph. 29*, 4 (July 2010). 2

[CSHD21] CHEN J., SCHÄFER F., HUANG J., DESBRUN M.: Multiscale cholesky preconditioning for ill-conditioned problems. *ACM Trans. Graph. 40*, 4 (July 2021). 2, 8

[CW17] CHEN R., WEBER O.: Gpu-accelerated locally injective shape deformation. *ACM Trans. Graph. 36*, 6 (Nov. 2017). URL: https://doi.org/10.1145/3130800.3130843, doi:10.1145/3130800.3130843. 2

[Dem19] DEMIDOV D.: Amgcl: An efficient, flexible, and extensible algebraic multigrid implementation. *Lobachevskii Journal of Mathematics 40*, 5 (2019), 535–546. 2, 8

[FH05] FLOATER M. S., HORMANN K.: Surface parameterization: a tutorial and survey. In *Advances in Multiresolution for Geometric Modelling* (Berlin, Heidelberg, 2005), Dodgson N. A., Floater M. S., Sabin M. A., (Eds.), Springer Berlin Heidelberg, pp. 157–186. 2

[FL16] FU X.-M., LIU Y.: Computing inversion-free mappings by simplex assembly. *ACM Trans. Graph. 35*, 6 (Dec. 2016). 2

[FLG15] FU X.-M., LIU Y., GUO B.: Computing locally injective mappings by advanced mips. *ACM Trans. Graph. 34*, 4 (July 2015). 2

[GSC18] GOLLA B., SEIDEL H.-P., CHEN R.: Piecewise linear mapping optimization based on the complex view. *Computer Graphics Forum 37*, 7 (2018), 233–243. doi:10.1111/cgf.13563. 2

[GVL13] GOLUB G. H., VAN LOAN C. F.: *Matrix Computations - 4th Edition*. Johns Hopkins University Press, Philadelphia, PA, 2013. 8

[HCLK24] HUANG K., CHITALU F. M., LIN H., KOMURA T.: Gipc: Fast and stable gauss-newton optimization of ipc barrier energy. *ACM Transactions on Graphics 43*, 2 (2024), 1–18. 1, 2

[HG99] HORMANN K., GREINER G.: Mips: An efficient global parametrization method. *Curve and Surface Design: Saint-Malo 2000* (11 1999), 10. 2, 4, 9

[HGMRT20] HYDE D. A., GAGNIERE S. W., MARQUEZ-RAZON A., TERAN J.: An implicit updated lagrangian formulation for liquids with large surface energy. *ACM Transactions on Graphics (TOG) 39*, 6 (2020), 1–13. 1, 2

[ITF04] IRVING G., TERAN J., FEDKIW R.: Invertible finite elements for robust simulation of large deformation. In *Proceedings of the 2004 ACM SIGGRAPH/Eurographics Symposium on Computer Animation* (Goslar, DEU, 2004), SCA '04, Eurographics Association, p. 131–140. 2

[JTPSH15] JAKOB W., TARINI M., PANOZZO D., SORKINE-HORNUNG O.: Instant field-aligned meshes. *ACM Trans. Graph. 34*, 6 (Nov. 2015). 2

[KE22] KIM T., EBERLE D.: Dynamic deformables: implementation and production practicalities (now with code!). In *ACM SIGGRAPH 2022 Courses* (New York, NY, USA, 2022), SIGGRAPH '22, Association for Computing Machinery. 2

[KGL16] KOVALSKY S. Z., GALUN M., LIPMAN Y.: Accelerated quadratic proxy for geometric optimization. *ACM Trans. Graph. 35*, 4 (July 2016). 2

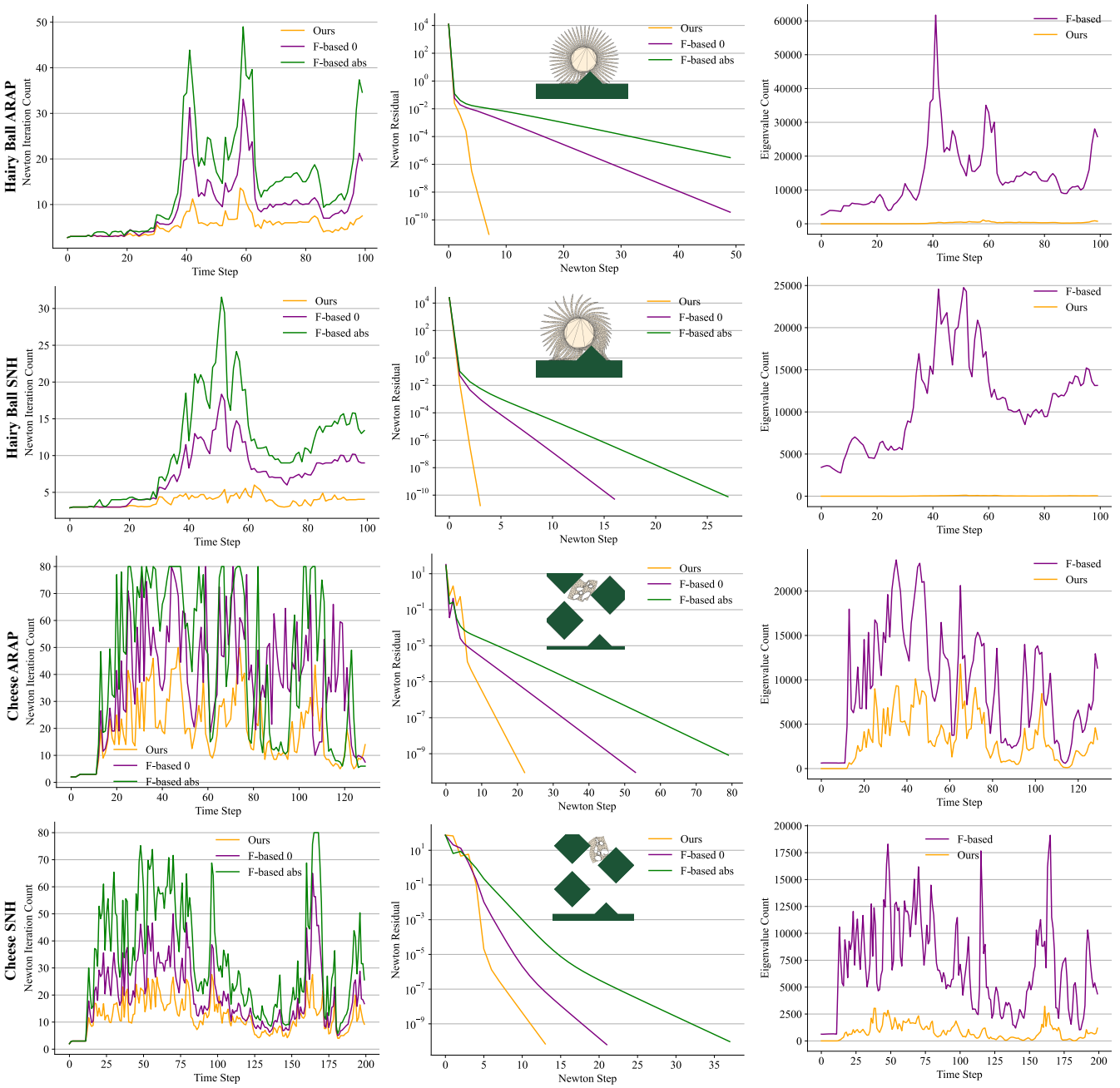
[Kim20] KIM T.: A finite element formulation of baraff-witkin cloth. In *Proceedings of the ACM SIGGRAPH/Eurographics Symposium on Computer Animation* (Goslar, DEU, 2020), SCA '20, Eurographics Association. 1, 2

[KNP07] KÄLBERER F., NIESER M., POLTHIER K.: Quadcover - surface parameterization using branched coverings. *Computer Graphics Forum 26*, 3 (2007), 375–384. 2

[LBK17] LIU T., BOUAZIZ S., KAVAN L.: Quasi-newton methods for real-time simulation of hyperelastic materials. *ACM Trans. Graph. 36*, 4 (July 2017). 2

[LCK22] LIN H., CHITALU F. M., KOMURA T.: Isotropic arap energy using cauchy-green invariants. *ACM Trans. Graph. 41*, 6 (Nov. 2022). 2

[LYL*25] LI C., YU P., LIU T., YU S., XIAO Y., LI S., HAO A., GAO



(a) Newton Step Count (b) Newton Convergence, single frame (c) Clamped Count

Figure 7: Dynamic simulation of a hairy ball (top two rows) and a square with holes (bottom two rows). Column (a) shows the number of Newton iterations at convergence for each time step, averaged over the sub-steps. Column (b) is the zoomed-in Newton Convergence curve at a single frame. Column (c) shows the total number of clamped eigenvalues using our unified projection strategy versus the **F**-based projection to zero [TSIF05] or absolute value [CLL*24]. Our algorithm consistently, and often dramatically, outperforms the other strategies.

Y., ZHAO Q.: Mgpbd: A multigrid accelerated global xpbd solver. In *Proceedings of the Special Interest Group on Computer Graphics and Interactive Techniques Conference Conference Papers* (New York, NY, USA, 2025), SIGGRAPH Conference Papers '25, Association for Computing Machinery. 2

[LYNF18] LIU L., YE C., NI R., FU X.-M.: Progressive parameterizations. *ACM Transactions on Graphics(SIGGRAPH) 37*, 4 (2018). 2, 7

[LZX*08] LIU L., ZHANG L., XU Y., GOTSMAN C., GORTLER S. J.: A local/global approach to mesh parameterization. SGP '08, Eurographics Association, p. 1495–1504. 2, 7

[MDM*02] MÜLLER M., DORSEY J., MCMILLAN L., JAGNOW R., CUTLER B.: Stable real-time deformations. In *Proceedings of the 2002 ACM SIGGRAPH/Eurographics Symposium on Computer Animation* (New York, NY, USA, 2002), SCA '02, Association for Computing Machinery, p. 49–54. 2

[Moo40] MOONEY M.: A theory of large elastic deformation. *Journal of applied physics 11*, 9 (1940), 582–592. 9

[MZS*11] MCADAMS A., ZHU Y., SELLE A., EMPEY M., TAMSTORF R., TERAN J., SIFAKIS E.: Efficient elasticity for character skinning with contact and collisions. *ACM Trans. Graph. 30*, 4 (July 2011). 2, 3

[NKJF09] NESME M., KRY P. G., JEŘÁBKOVÁ L., FAURE F.: Preserving topology and elasticity for embedded deformable models. In *ACM SIGGRAPH 2009 Papers* (New York, NY, USA, 2009), SIGGRAPH '09, Association for Computing Machinery. 2

[OBLN17] OVERBY M., BROWN G. E., LI J., NARAIN R.: Admm $\supseteq$ projective dynamics: Fast simulation of hyperelastic models with dynamic constraints. *IEEE Transactions on Visualization and Computer Graphics 23*, 10 (2017), 2222–2234. 2

[PDA00] PICINBONO G., DELINGETTE H., AYACHE N.: Real-time large displacement elasticity for surgery simulation: Non-linear tensor-mass model. In *Proceedings of the Third International Conference on Medical Image Computing and Computer-Assisted Intervention* (Berlin, Heidelberg, 2000), MICCAI '00, Springer-Verlag, p. 643–652. 2

[QGN15] QIU Y., GUENNEBAUD G., NIESEN J.: Spectra: C++ library for large scale eigenvalue problems. *Retrieved from https://spectralib.org/.* (2015). 7

[Riv48] RIVLIN R. S.: Large elastic deformations of isotropic materials iv. further developments of the general theory. *Philosophical transactions of the royal society of London. Series A, Mathematical and physical sciences 241*, 835 (1948), 379–397. 9

[RPPSH17] RABINOVICH M., PORANNE R., PANOZZO D., SORKINE-HORNUNG O.: Scalable locally injective mappings. *ACM Trans. Graph. 36*, 2 (Apr. 2017). 2

[SA07] SORKINE O., ALEXA M.: As-rigid-as-possible surface modeling. In *Proceedings of the Fifth Eurographics Symposium on Geometry Processing* (Goslar, DEU, 2007), SGP '07, Eurographics Association, p. 109–116. 2, 3, 10

[SB12] SIFAKIS E., BARBIC J.: Fem simulation of 3d deformable solids: a practitioner's guide to theory, discretization and model reduction. In *ACM SIGGRAPH 2012 Courses* (New York, NY, USA, 2012), SIGGRAPH '12, Association for Computing Machinery. 2

[SGK18] SMITH B., GOES F. D., KIM T.: Stable neo-hookean flesh simulation. *ACM Trans. Graph. 37*, 2 (Mar. 2018). 2, 4, 10

[SGK19] SMITH B., GOES F. D., KIM T.: Analytic eigensystems for isotropic distortion energies. *ACM Trans. Graph. 38*, 1 (Feb. 2019). 1, 2, 4

[SK23] SHI A., KIM T.: A unified analysis of penalty-based collision energies. *Proc. ACM Comput. Graph. Interact. Tech. 6*, 3 (Aug. 2023). 1, 2

[SKPSH13] SCHÜLLER C., KAVAN L., PANOZZO D., SORKINE-HORNUNG O.: Locally injective mappings. *Computer Graphics Forum (proceedings of EUROGRAPHICS/ACM SIGGRAPH Symposium on Geometry Processing) 32*, 5 (2013), 125–135. 2

[SPSH*17] SHTENGEL A., PORANNE R., SORKINE-HORNUNG O., KOVALSKY S. Z., LIPMAN Y.: Geometric optimization via composite majorization. *ACM Trans. Graph. 36*, 4 (July 2017). 2, 7

[SS15] SMITH J., SCHAEFER S.: Bijective parameterization with free boundaries. *ACM Trans. Graph. 34*, 4 (July 2015). 4, 9

[TSIF05] TERAN J., SIFAKIS E., IRVING G., FEDKIW R.: Robust quasistatic finite elements and flesh simulation. In *Proceedings of the 2005 ACM SIGGRAPH/Eurographics Symposium on Computer Animation* (New York, NY, USA, 2005), SCA '05, Association for Computing Machinery, p. 181–190. 2, 6, 8, 11

[VCD*17] VAXMAN A., CAMPEN M., DIAMANTI O., BOMMES D., HILDEBRANDT K., TECHNION M. B.-C., PANOZZO D.: Directional field synthesis, design, and processing. In *ACM SIGGRAPH 2017 Courses* (New York, NY, USA, 2017), SIGGRAPH '17, Association for Computing Machinery. 2

[VDDKP01] VAN DEN DOEL K., KRY P. G., PAI D. K.: Foleyautomatic: physically-based sound effects for interactive simulation and animation. In *Proceedings of the 28th annual conference on Computer graphics and interactive techniques* (2001), pp. 537–544. 1

[Wan15] WANG H.: A chebyshev semi-iterative approach for accelerating projective and position-based dynamics. *ACM Trans. Graph. 34*, 6 (Nov. 2015). 2

[WK23] WU H., KIM T.: An eigenanalysis of angle-based deformation energies. *Proc. ACM Comput. Graph. Interact. Tech. 6*, 3 (Aug. 2023). 4

[WLF*20] WANG X., LI M., FANG Y., ZHANG X., GAO M., TANG M., KAUFMAN D. M., JIANG C.: Hierarchical optimization time integration for cfl-rate mpm stepping. *ACM Transactions on Graphics (TOG) 39*, 3 (2020), 1–16. 2

[WWW22] WU B., WANG Z., WANG H.: A gpu-based multilevel additive schwarz preconditioner for cloth and deformable body simulation. *ACM Trans. Graph. 41*, 4 (July 2022). 2

[ZBK18] ZHU Y., BRIDSON R., KAUFMAN D. M.: Blended cured quasi-newton for distortion optimization. *ACM Trans. Graph. 37*, 4 (July 2018). 2

[ZJ16] ZHOU Q., JACOBSON A.: Thingi10k: A dataset of 10,000 3d-printing models. *arXiv preprint arXiv:1605.04797* (2016). 7