Spectral Coarsening of Geometric Operators

HSUEH-TI DEREK LIU, University of Toronto, Canada
ALEC JACOBSON, University of Toronto, Canada
MAKS OVSJANIKOV, École Polytechnique, France

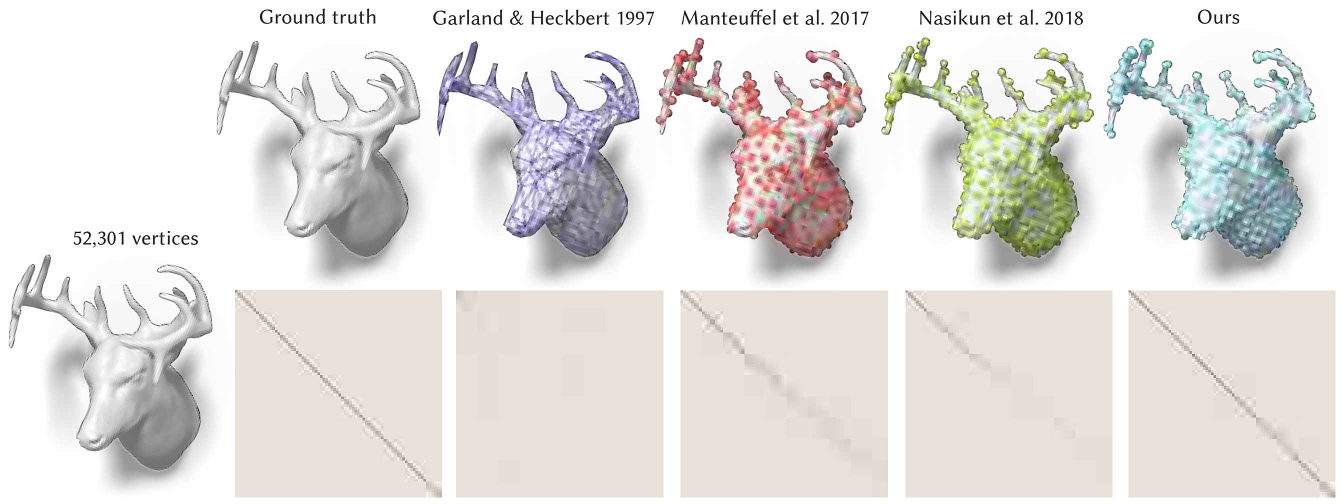


Fig. 1. There are many ways to coarsen a 52,301×52,301 sparse anisotropic Laplace matrix down to a sparse 500×500 matrix: simplify the mesh [Garland and Heckbert 1997] and rediscretize; apply algebraic multigrid coarsening [Manteuffel et al. 2017]; or approximate using radial-basis functions [Nasikun et al. 2018]. We introduce a way to measure how well the coarse operator maintains the original operator's eigenvectors (bottom row). The visualization shows deviation from a diagonal matrix indicating poor eigenvector preservation. In response, we introduce an optimization to coarsen geometric operators while preserving eigenvectors and maintaining sparsity and positive semi-definiteness.

We introduce a novel approach to measure the behavior of a geometric operator before and after coarsening. By comparing eigenvectors of the input operator and its coarsened counterpart, we can quantitatively and visually analyze how well the spectral properties of the operator are maintained. Using this measure, we show that standard mesh simplification and algebraic coarsening techniques fail to maintain spectral properties. In response, we introduce a novel approach for *spectral coarsening*. We show that it is possible to significantly reduce the sampling density of an operator derived from a 3D shape without affecting the low-frequency eigenvectors. By marrying techniques developed within the algebraic multigrid and the functional maps literatures, we successfully coarsen a variety of isotropic and anisotropic operators while maintaining sparsity and positive semi-definiteness. We demonstrate the utility of this approach for applications including operator-sensitive sampling, shape matching, and graph pooling for convolutional neural networks.

Authors' addresses: Hsueh-Ti Derek Liu, University of Toronto, 40 St. George Street, Toronto, ON, M5S 2E4, Canada; Alec Jacobson, University of Toronto, 40 St. George Street, Toronto, ON, M5S 2E4, Canada; Maks Ovsjanikov, École Polytechnique, 1 Rue Honoré d'Estienne d'Orves, Palaiseau, 91120, France.

0730-0301/2019/7-ART105 $15.00
https://doi.org/10.1145/3306346.3322953

CCS Concepts: • **Computing methodologies → Shape analysis**; • **Mathematics of computing →** *Computations on matrices.*

Additional Key Words and Phrases: geometry processing, numerical coarsening, spectral geometry

ACM Reference Format:
Hsueh-Ti Derek Liu, Alec Jacobson, and Maks Ovsjanikov. 2019. Spectral Coarsening of Geometric Operators. *ACM Trans. Graph.* 38, 4, Article 105 (July 2019), 13 pages. https://doi.org/10.1145/3306346.3322953

1 INTRODUCTION

Geometry processing relies heavily on building matrices to represent linear operators defined on geometric domains. While typically sparse, these matrices are often too large to work with efficiently when defined over high resolution representations. A common solution is to simplify or coarsen the domain. However, matrices built from coarse representations often do not behave the same way as their fine counterparts leading to inaccurate results and artifacts when resolution is restored. Quantifying and categorizing *how* this behavior is different is not straightforward and most often coarsening is achieved through operator-oblivious remeshing. The common appearance-based or geometric metrics employed by remeshers, such as the classical quadratic error metric [Garland and Heckbert 1997] can have very little correlation to maintaining operator behavior.

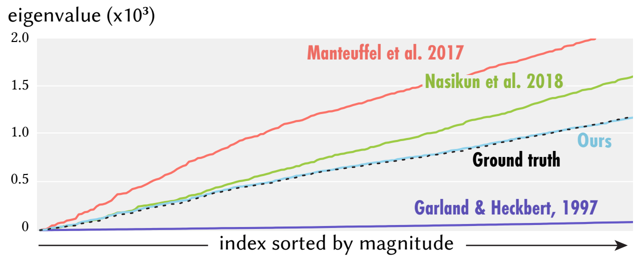
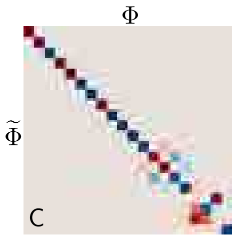


Fig. 2. Our coarsening directly preserves eigenvectors so eigenvalues are also implicitly preserved: eigenvalue plot of Fig. 1.

We propose a novel way to compare the spectral properties of a discrete operator before and after coarsening, and to guide the coarsening to preserve them. Our method is motivated by the recent success of spectral methods in shape analysis and processing tasks, such as shape comparison and non-rigid shape matching, symmetry detection, and vector field design to name a few. These methods exploit eigenfunctions of various operators, including the Laplace-Beltrami operator, whose eigenfunctions can be seen as a generalization of the Fourier basis to curved surfaces. Thus, spectral methods expand the powerful tools from Fourier analysis to more general domains such as shapes, represented as triangle meshes in 3D. We propose to measure how well the eigenvectors (and by extension eigenvalues) of a matrix $\mathsf{L} \in \mathbb{R}^{n\times n}$ on the high-resolution domain are maintained by its coarsened counterpart $\widetilde{\mathsf{L}} \in \mathbb{R}^{m\times m}$ $(m < n)$ by computing a dense matrix $\mathsf{C}^{k\times k}$, defined as the inner product of the first k eigenvectors $\Phi \in \mathbb{R}^{n\times k}$ and $\widetilde{\Phi} \in \mathbb{R}^{m\times k}$ of L and $\widetilde{\mathsf{L}}$ respectively:



$$\mathsf{C} = \widetilde{\Phi}^\top \widetilde{\mathsf{M}} \mathsf{P} \Phi, \qquad (1)$$

where $\widetilde{\mathsf{M}} \in \mathbb{R}^{m\times m}$ defines a mass-matrix on the coarse domain and $\mathsf{P} \in \mathbb{R}^{m\times n}$ is a restriction operator from fine to coarse. The closer C resembles the identity matrix the more the eigenvectors of the two operators before and after coarsening are equivalent.

We show through a variety of examples that existing geometric and algebraic coarsening methods fail to varying degrees to preserve the eigenvectors and the eigenvalues of common operators used in geometry processing (see Fig. 1 and Fig. 2).

In response, we propose a novel coarsening method that achieves much better preservation under this new metric. We present an optimization strategy to coarsen an input positive semi-definite matrix in a way that better maintains its eigenvectors (see Fig. 1, right) while preserving matrix sparsity and semi-definiteness. Our optimization is designed for operators occurring in geometry processing and computer graphics, but does not rely on access to a geometric mesh: our input is the matrix L, and an area measure M on the fine domain, allowing us to deal with non-uniform sampling. The output coarsened operator $\widetilde{\mathsf{L}}$ and an area measure $\widetilde{\mathsf{M}}$ on the coarse domain are defined for a subset of the input elements chosen carefully to respect anisotropy and irregular mass distribution defined by the input operator. The coarsened operator is optimized via a novel formulation of coarsening as a sparse semi-definite programming optimization based on the operator commutative diagram.

We demonstrate the effectiveness of our method at categorizing the failure of existing methods to maintain eigenvectors on a number of different examples of geometric domains including triangle meshes, volumetric tetrahedral meshes and point clouds. In direct comparisons, we show examples of successful spectral coarsening for isotropic and anisotropic operators. Finally, we provide evidence that spectral coarsening can improve downstream applications such as shape matching, graph pooling for graph convolutional neural networks, and data-driven mesh sampling.

2 RELATED WORK

Mesh Simplification and Hierarchical Representation. The use of multi-resolution shape representations based on mesh simplification has been extensively studied in computer graphics, with most prominent early examples including mesh decimation and optimization approaches [Hoppe et al. 1993; Schroeder et al. 1992] and their multiresolution variants e.g., *progressive meshes* [Hoppe 1996; Popović and Hoppe 1997] (see [Cignoni et al. 1998] for an overview and comparison of a wide range of mesh simplification methods). Among these classical techniques, perhaps the best-known and most widely used approach is based on the *quadratic error metrics* introduced in [Garland and Heckbert 1997] and extended significantly in follow-up works e.g., to incorporate texture and appearance attributes [Garland and Heckbert 1998; Hoppe 1999] to name a few. Other, more recent approaches have also included variational shape approximation [Cohen-Steiner et al. 2004] and wavelet-based methods especially prominent in shape compression [Peyré and Mallat 2005; Schroder 1996], as well as more flexible multi-resolution approaches such as those based on hybrid meshes [Guskov et al. 2002] among myriad others. Although mesh simplification is a very well-studied problem, the vast majority of approximation techniques is geared towards preservation of shape *appearance* most often formulated via the preservation of local geometric features. Li et al. [2015] conduct a *frequency-adaptive* mesh simplification to better preserve the acoustic transfer of a shape by appending a modal displacement as an extra channel during progressive meshes. In Fig. 3, we show that this method fails to preserve all low frequency eigenvectors (since it is designed for a single frequency). Our measure helps to reveal the accuracy of preserving *spectral* quantities, and to demonstrate that existing techniques often fail to achieve this objective.

Numerical Coarsening in Simulation. Coarsening the geometry of an elasticity simulation mesh without adjusting the material parameters (e.g., Young's modulus) leads to *numerical stiffening.* Kharevych et al. [2009] recognize this and propose a method to independently adjust the per-tetrahedron elasticity tensor of a coarse mesh to agree with the six smallest deformation modes of a fine-mesh inhomogeneous material object (see Fig. 3 for comparison). Chen et al. [2015] extend this idea via a data-driven lookup table. Chen et al. [2018] consider numerical coarsening for regular-grid domains, where matrix-valued basis functions on the coarse domain are optimized to again agree with the six smallest deformation modes of a fine mesh through a global quadratic optimization. To better capture vibrations, Chen et al. [2017] coarsen regular-grids of homogeneous

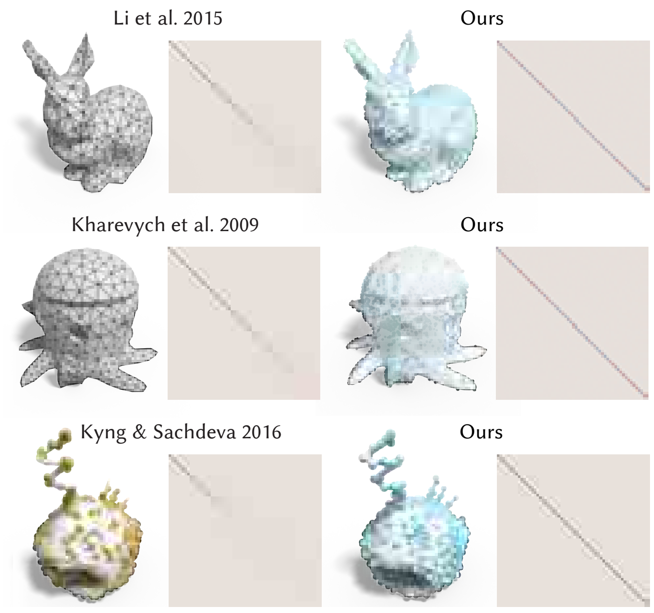


Fig. 3. As methods of Kharevych et al. [2009]; Kyng and Sachdeva [2016]; Li et al. [2015] are not designed for preserving spectral properties, they only preserve very low frequency eigenvectors (top-left corner of matrix images), but fails for subsequent modes.

materials until their low frequency vibration modes exceeding a Hausdorff distance threshold. The ratio of the first eigenvalue before and after coarsening is then used to rescale the coarse materials Young's modulus. In contrast to these methods, our proposed optimization is not restricted to regular grids or limited by adjusting physical parameters directly.

Algebraic Multigrid. Traditional multigrid methods coarsen the mesh of the geometric domain recursively to create an efficient iterative solver for large linear systems [Briggs et al. 2000]. For isotropic operators, each geometric level smooths away error at the corresponding frequency level [Burt and Adelson 1983]. *Algebraic* multigrid (AMG) does not see or store geometric levels, but instead defines a hierarchy of system matrices that attempt to smooth away error according to the input matrix's spectrum [Xu and Zikatanov 2017]. AMG has been successfully applied for anisotropic problems such as cloth simulation [Tamstorf et al. 2015]. Without access to underlying geometry, AMG methods treat the input sparse matrix as a graph with edges corresponding to non-zeros and build a coarser graph for each level by removing nodes and adding edges according to an *algebraic distance* determined by the input matrix. AMG like all multigrid hierarchies are typically measured according to their solver convergence rates [Xu and Zikatanov 2017]. While eigenvector preservation is beneficial to AMG, an efficient solver must also avoid adding too many new edges during coarsening (i.e., [Kahl and Rottmann 2018; Livne and Brandt 2012]). Meanwhile, to remain competitive to other blackbox solvers, AMG methods also strive to achieve very fast hierarchy construction performance [Xu and Zikatanov 2017]. Our analysis shows how state-of-the-art AMG coarsening methods such as [Manteuffel et al. 2017] which is designed for fast convergence fails to preserve eigenvectors and eigenvalues (see Fig. 1 and Fig. 2). Our subsequent optimization formulation in Section 3.1 and Section 3.2 is inspired by the "root node" selection and Galerkin projection approaches found in the AMG literature [Bell 2008; Manteuffel et al. 2017; Stuben 2000].

Spectrum Preservation. In contrast to geometry-based mesh simplification very few methods have been proposed targeting preservation of spectral properties. Öztireli and colleagues [Öztireli et al. 2010] introduced a technique for spectral sampling on surfaces. In a similar spirit to our approach, their method aims to compute samples on a surface that can approximate the Laplacian spectrum of the original shape. This method targets only isotropic sampling and is not well-suited to more diverse operators such as the anisotropic Laplace-Beltrami operator handled by our approach. More fundamentally, our goal is to construct a coarse representation that preserves an entire *operator*, and allows, for example to compute eigenfunctions and eigenvalues in the coarse domain, which is not addressed by a purely sampling-based strategy. More recently, an efficient approach for approximating the Laplace-Beltrami eigenfunctions has been introduced in [Nasikun et al. 2018], based on a combination of fast Poisson sampling and an adapted coarsening strategy. While very efficient, as we show below, this method unfortunately fails to preserve even medium frequencies, especially in the presence of high-curvature shape features or more diverse, including anisotropic Laplacian, operators.

We note briefly that spectrum preservation and optimization has also been considered in the context of sound synthesis, including [Bharaj et al. 2015], and more algebraically for efficient solutions of Laplacian linear systems [Kyng and Sachdeva 2016]. In Fig. 3, we show that the method of Kyng and Sachdeva [2016] only preserves very low frequency eigenvectors. Our approach is geared towards operators defined on non-uniform triangle meshes and does not have limitations of the approach of [Kyng and Sachdeva 2016] which only works on Laplacians where all weights are positive.

3 METHOD

The input to our method is a n-by-n sparse, positive semi-definite matrix $\mathsf{L} \in \mathbb{R}^{n \times n}$. We assume L is the Hessian of an energy derived from a geometric domain with n vertices and the sparsity pattern is determined by the connectivity of a mesh or local neighbor relationship. For example, L may be the discrete cotangent Laplacian, the Hessian of the discrete Dirichlet energy. However, we do not require direct access to the geometric domain or its spatial embedding. We also take as input a non-negative diagonal weighting or mass matrix $\mathsf{M} \in \mathbb{R}^{n \times n}$ (i.e., defining the inner-product space of vectors from the input domain). The main parameter of our method is the positive number $m < n$ which determines the size of our coarsened output.

Our method outputs a sparse, positive semi-definite matrix $\widetilde{\mathsf{L}} \in \mathbb{R}^{m \times m}$ that attempts to maintain the low-frequency eigenvalues and eigenvectors of the input matrix L (see Fig. 4).

Algorithm 1: Spectral Coarsening given L, M and m

```
$\mathsf{P}, \mathsf{K} \leftarrow$ *combinatorial coarsening*$(\mathsf{L}, \mathsf{M}, m)$;
$\widetilde{\mathsf{L}}, \widetilde{\mathsf{M}} \leftarrow$ *operator optimization*$(\mathsf{L}, \mathsf{M}, \mathsf{P}, \mathsf{K})$;
```

We propose coarsening in two steps (see Algorithm 1). First we treat the input matrix L as encoding a graph and select a subset of

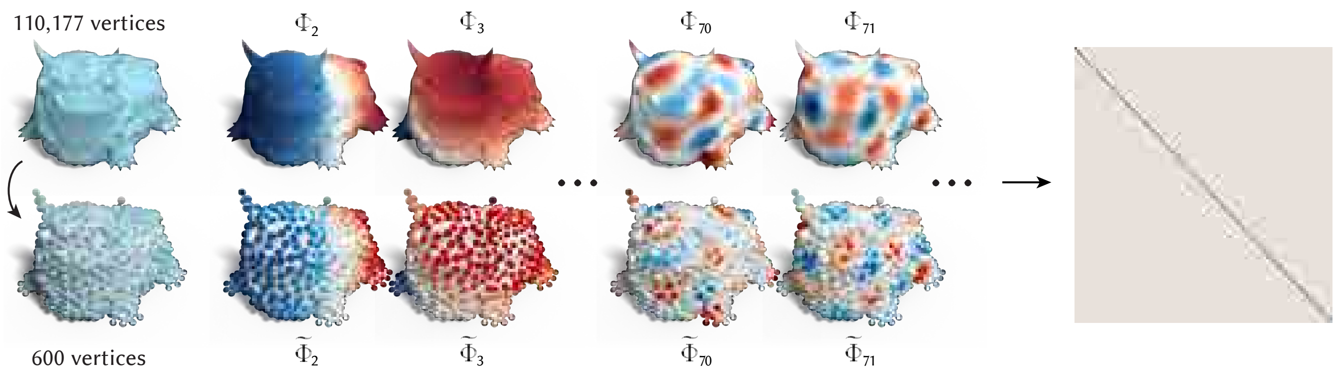


Fig. 4. When eigenvectors are equivalent (up to sign) before and after coarsening the operator, the matrix C (right) resembles the identity matrix.

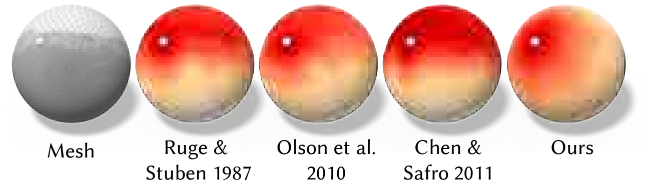


Fig. 6. We visualize the graph shortest path distance from the source point (gray) to all the other points, where the strength of connections between adjacent points is defined using different operator-dependent strength measures. In an isotropic problem, our resulting "distance" is more robust to different element sizes and grows more uniformly in all directions (right).

m "root" nodes, assigning all others to clusters based on a novel graph-distance. This clustering step defines a restriction operator (P in Eq. 1) and a cluster-assignment operator K that determines the sparsity pattern of our output matrix $\widetilde{\mathsf{L}}$. In the second stage, we optimize the non-zero values of $\widetilde{\mathsf{L}}$.

3.1 Combinatorial coarsening

Given an input operator $\mathsf{L} \in \mathbb{R}^{n\times n}$ and corresponding mass-matrix $\mathsf{M} \in \mathbb{R}^{n\times n}$, the goal of this stage is to construct two sparse binary matrices $\mathsf{K}, \mathsf{P} \in \{0,1\}^{m\times n}$ (see Fig. 5). Acting as a cluster-assignment operator, K has exactly one 1 per column, so that $\mathsf{K}_{ij} = 1$ indicates that element j on the input domain is assigned to element i on the coarsened domain. Complementarily, acting as a restriction or subset-selection operator, P has exactly one 1 per row and no more than one 1 per column, so that $\mathsf{P}_{ij} = 1$ indicates that element j on the input domain is selected as element i in the coarsened domain to *represent* its corresponding cluster. Following the terminology from the algebraic multigrid literature, we refer to this selected element as the "root node" of the cluster [Manteuffel et al. 2017]. In our figures, we visualize P by drawing large dots on the selected nodes and K by different color segments.

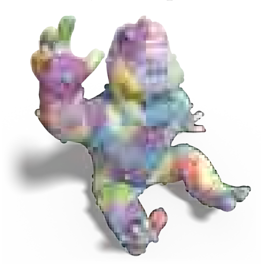


Fig. 5. Blue dots and colored regions indicate "root nodes" and clusters selected by P and K respectively.

Consider the graph with n nodes implied by interpreting non-zeros of L as undirected edges. Our node-clustering and root-node selection should respect how quickly information at one node *diffuses* to neighboring nodes according to L and how much mass or weight is associated with each node according to M. Although a variety of algebraic distances have been proposed [Chen and Safro 2011; Livne and Brandt 2012; Olson et al. 2010; Ruge and Stüben 1987], they are not directly applicable to our geometric tasks because they are sensitive to different finite-element sizes (see Fig. 6).

According to this diffusion perspective, the edge-distance of the edge between nodes i and j should be inversely correlated with $-\mathsf{L}_{ij}$ and positively correlated with $(\mathsf{M}_{ii} + \mathsf{M}_{jj})$. Given the *units* of L and M in terms of powers of length p and q respectively (e.g., the discrete cotangent Laplacian for a triangle mesh has units p=0, the barycentric mass matrix has units q=2), then we adjust these correlations so that our edge-distance has units of length. Putting these relations together and avoiding negative lengths due to positive off-diagonal entries in L, we define the edge-distance between connected nodes as:

$$\mathsf{D}_{ij} = \max\left(\frac{(\mathsf{M}_{ii} + \mathsf{M}_{jj})^{(p+1)/q}}{-\mathsf{L}_{ij}}, 0\right).$$

Compared to Euclidean or geodesic distance, shortest-path distances using this edge-distance will respect anisotropy of L (see Fig. 7, Fig. 8). Compared to state-of-the-art algebraic distances, our distance will account for irregular mass distribution, e.g., due to irregular meshing (see Fig. 6).

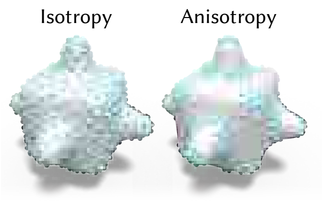


Fig. 7. Our coarsening is aware of the anisotropy of the underlying operator, resulting in a different set of selected root nodes.

Given this (symmetric) matrix of edge-distances, we compute the k-mediods clustering [Struyf et al. 1997] of the graph nodes according to shortest path distances (computed efficiently using the modified Bellman-Ford method and Lloyd aggregation method of Bell [2008]). We initialize this iterative optimization with a random set of k root nodes. Unlike k-means where the *mean* of each cluster is not restricted to the set of input points in space, k-mediods chooses the cluster root as the mediod-node of the cluster (i.e., the node with minimal total distance to all

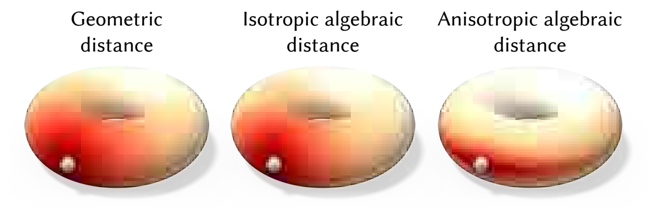


Fig. 8. We visualize the graph shortest path distance from the source point (gray) to all the other points. Our operator-dependent distance can handle both isotropic and anisotropic problems, whereas standard geometry-based measure (e.g. edge length) is limited to isotropic problems.

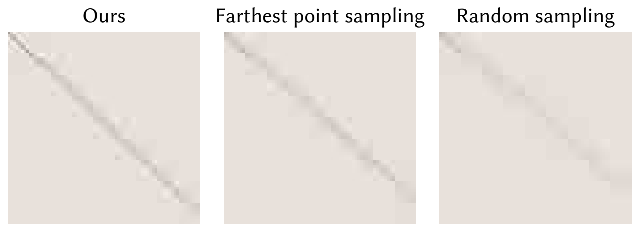


Fig. 9. Coarsening with our operator-aware distance (left) results in a better eigenfunction preservation compared to the farthest point sampling (middle) and the random sampling (right) on an anisotropic operator.

other nodes in the cluster). All other nodes are then re-assigned to their closest root. This process is iterated until convergence. Cluster assignments and cluster roots are stored as K and P accordingly. Comparing to the farthest point sampling and the random sampling, our approach results in a better eigenfunction preservation image for anisotropic operators (Fig. 9).

We construct a sparsity pattern for $\widetilde{\mathsf{L}}$ so that $\widetilde{\mathsf{L}}_{ij}$ may be non-zero if the cluster j is in the three-ring neighborhood of cluster i as determined by cluster-cluster adjacency. If we let $\mathsf{S_L} \in \{0,1\}^{n\times n}$ be a binary matrix containing a 1 if and only if the corresponding element of L is non-zero, then we can compute the "cluster adjacency" matrix $\widetilde{\mathsf{A}} = \mathsf{K}\mathsf{S_L}\mathsf{K}^\top \in \{0,1\}^{m\times m}$ so that $\widetilde{\mathsf{A}}_{ij} = 1$ if and only if the clusters i and j contain some elements u and v such that $\mathsf{L}_{uv} \neq 0$. Using this adjacency matrix, we create a sparse restriction matrix with wider connectivity $\mathsf{S_G} = \mathsf{K}^\top\widetilde{\mathsf{A}} \in \{0,1\}^{n\times m}$. Finally, our predetermined sparsity pattern for $\widetilde{\mathsf{L}}$ is defined to be that of $\mathsf{S}_{\widetilde{\mathsf{L}}} = \mathsf{S_G}^\top\mathsf{S_L}\mathsf{S_G} = \widetilde{\mathsf{A}}^3 \in \{0,1\}^{m\times m}$. We found that using the cluster three-ring sparsity is a reasonable trade-off between in-fill density and performance of the optimized operator. Assuming the cluster graph is 2-manifold with average valence 6, the three-ring sparsity implies that $\widetilde{\mathsf{L}}$ will have 37 non-zeros per row/column on average, independent to m and n. In practice, our cluster graph is nearly 2-manifold. The $\widetilde{\mathsf{L}}$ in Fig. 5, for instance, has approximately 39 non-zeros per row/column.

3.2 Operator optimization

Given a clustering, root node selection and the desired sparsity pattern, our second step is to compute a coarsened matrix $\widetilde{\mathsf{L}}$ that maintains the eigenvectors of the input matrix L as much as possible. Since L and $\widetilde{\mathsf{L}}$ are of different sizes, their corresponding eigenvectors are also of different lengths. To compare them in a meaningful way we will use the functional map matrix C defined in Eq. 1 implied by the restriction operator P (note that: prolongation from coarse to fine is generally ill-defined). This also requires a mass-matrix on the coarsened domain, which we compute by lumping cluster masses: $\widetilde{\mathsf{M}} = \mathsf{KMK}^\top$. The first k eigenvectors for the input operator and yet-unknown coarsened operator may be computed as solutions to the generalized eigenvalue problems $\mathsf{L}\Phi = \Lambda\mathsf{M}\Phi$ and $\widetilde{\mathsf{L}}\widetilde{\Phi} = \widetilde{\Lambda}\widetilde{\mathsf{M}}\widetilde{\Phi}$, where $\Lambda, \widetilde{\Lambda}$ are eigenvalue matrices.

Knowing that the proximity of the functional map matrix C to an identity matrix encodes eigenvector preservation, it might be tempting to try to enforce $\|\mathsf{C} - \mathsf{I}\|_F$ directly. This however is problematic because it does not handle sign flips or multiplicity (see Fig. 10). More importantly, recall that in our setting C is not a free variable, but rather a non-linear function (via eigen decomposition) of the unknown sparse matrix $\widetilde{\mathsf{L}}$.

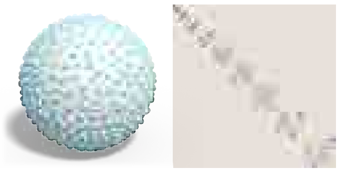


Fig. 10. The optimized C should be block diagonal when the operator has algebraic multiplicity.

Instead, we propose to minimize the failure to realize the commutative diagram of a functional map. Ideally, for any function on the input domain $\mathbf{f} \in \mathbb{R}^n$ applying the input operator $\mathsf{M}^{-1}\mathsf{L}$ and then the restriction matrix P is equivalent to applying P then $\widetilde{\mathsf{M}}^{-1}\widetilde{\mathsf{L}}$, resulting in the same function $\widetilde{\mathbf{f}} \in \mathbb{R}^m$ on the coarsened domain:

$$\begin{tikzcd} \mathbf{f} \arrow[r, "\mathsf{M}^{-1}\mathsf{L}"] \arrow[d, "\mathsf{P}"'] & \bullet \arrow[d, "\mathsf{P}"] \\ \bullet \arrow[r, "\widetilde{\mathsf{M}}^{-1}\widetilde{\mathsf{L}}"'] & \widetilde{\mathbf{f}} \end{tikzcd} \tag{2}$$

This leads to a straightforward energy that minimizes the difference between the two paths in the commutative diagram for all possible functions $\mathbf{f}$:

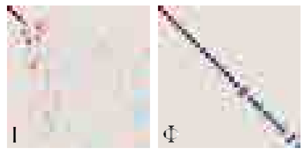
$$E(\widetilde{\mathsf{L}}) = \|\mathsf{PM}^{-1}\mathsf{LI} - \widetilde{\mathsf{M}}^{-1}\widetilde{\mathsf{L}}\mathsf{PI}\|^2_{\widetilde{\mathsf{M}}}, \tag{3}$$

where $\mathsf{I} \in \mathbb{R}^{n\times n}$ is the identity matrix (included didactically for the discussion that follows) and

$$\|\mathsf{X}\|^2_{\widetilde{\mathsf{M}}} = \operatorname{tr}(\mathsf{X}^\top\widetilde{\mathsf{M}}\mathsf{X})$$

computes the Frobenius inner-product defined by $\widetilde{\mathsf{M}}$.

By using I as the spanning matrix, we treat all functions equally in an L2 sense. Inspired by the functional maps literature, we can instead compute this energy over only lower frequency functions spanned by the first k eigenvectors $\Phi \in \mathbb{R}^{n\times k}$ of the operator L. Since high frequency functions naturally cannot live on a coarsened domain, this parameter k allows the optimization to focus on functions that matter. Consequently, preservation of low frequency eigenvectors dramatically improves (see inset).



Substituting Φ for I in Eq. 3, we now consider minimizing this reduced energy E_k over all possible *sparse* positive semi-definite

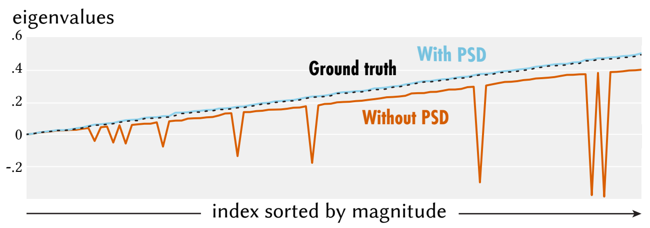


Fig. 11. Dropping the PSD constraint leads to a simple quadratic optimization problem which can be solved directly, but it produces a non-PSD $\widetilde{L}$ that contains negative eigenvalues.

(PSD) matrices $\widetilde{L}$:

$$\underset{\widetilde{L} \doteq S_{\widetilde{L}}}{\text{minimize}} \ \underbrace{\frac{1}{2}\|PM^{-1}L\Phi - \widetilde{M}^{-1}\widetilde{L}P\Phi\|^2_{\widetilde{M}}}_{E_k(\widetilde{L})} \tag{4}$$

$$\text{subject to} \quad \widetilde{L} \text{ is positive semi-definite} \tag{5}$$

$$\text{and} \quad \widetilde{L}P\Phi_0 = 0 \tag{6}$$

where we use $X \doteq Y$ to denote that X has the same sparsity pattern as Y. The final linear-equality constraint in Eq. 6 ensures that the eigen-vectors Φ_0 corresponding to zero eigenvalues are exactly preserved ($PL\Phi_0 = PM\Phi_0 0 = 0$). Note that while it might seem that Eq. 4 is only meant to preserve the eigen*vectors*, a straightforward calculation (See Appendix C) shows that it promotes the preservation of eigen*values* as well.

This optimization problem is convex [Boyd and Vandenberghe 2004], but the sparsity constraint makes it challenging to solve efficiently. Most efficient semi-definite programming (SDP) solvers (e.g., Mosek, cvxopt, Gurobi) only implement *dense* PSD constraints. The academic community has studied SDPs over sparse matrices, yet solutions are not immediately applicable (e.g., those based on chordal sparsity [Vandenberghe and Andersen 2015; Zheng et al. 2017]) or practically efficient (e.g., [Andersen et al. 2010]). Even projecting a sparse matrix X on to the set of PSD matrices with the same sparsity pattern is a difficult sub-problem (the so-called sparse matrix nearness problem, e.g., [Sun and Vandenberghe 2015]), so that proximal methods such as ADMM lose their attractiveness.

If we drop the PSD constraint, the result is a simple quadratic optimization with linear constraints and can be solved directly. While this produces solutions with very low objective values E_k, the eigenvector preservation is sporadic and negative eigenvalues appear (see Fig. 11). Conversely, attempting to replace the PSD constraint with the *stricter* but more amenable diagonal dominance linear inequality constraint (i.e., $\widetilde{L}_{ii} \geq \sum_{j \neq i} \widetilde{L}_{ji}$) produces a worse objective value and poor eigenvector preservation.

Instead, we propose introducing an auxiliary sparse matrix variable $G \in \mathbb{R}^{n \times m}$ and restricting the coarsened operator to be created by using G as an interpolation operator: $\widetilde{L} := G^\top LG$. Substituting this into Eq. 4, we optimize

$$\underset{G \doteq S_G}{\text{minimize}} \ \underbrace{\frac{1}{2}\|PM^{-1}L\Phi - \widetilde{M}^{-1}G^\top LGP\Phi\|^2_{\widetilde{M}}}_{E_k(G)}, \tag{7}$$

$$\text{subject to} \quad GP\Phi_0 = \Phi_0$$

where the sparsity of $\widetilde{L}$ is maintained by requiring sparsity of G. The null-space constraint remains linear because $GP\Phi_0 = \Phi_0 \Rightarrow G^\top LGP\Phi_0 = 0$ implies that $\widetilde{L}$ contains the null-space of L. The converse will not necessarily be true, but is unlikely to happen because this would represent inefficient minimization of the objective. In practice, we never found that spurious null-spaces occurred. While we get to remove the PSD constraint ($L \succcurlyeq 0$ implies $G^\top LG \succcurlyeq 0$), the price we have paid is that the energy is no longer quadratic in the unknowns, but quartic.

Therefore in lieu of convex programming, we optimize this energy over the non-zeros of G using a gradient-based algorithm with a fixed step size γ. Specifically, we use nadam [Dozat 2016] optimizer which is a variant of gradient descent that combines momentum and Nesterov's acceleration. For completeness, we provide the *sparse* matrix-valued gradient $\partial E_k / \partial G$ in Appendix A. The sparse linear equality constraints are handled with the orthogonal projection in Appendix B). We summarize our optimization in pseudocode Algorithm 2. We stop the optimization if it stalls (i.e., does not decrease the objective after 10 iterations) and use a fixed step size $\gamma = 0.02$. This rather straightforward application of a gradient-based optimization to maintaining the commutative diagram in Eq. 2 performs quite well for a variety of domains and operators.

Algorithm 2: Operator optimization using nadam

```
$G \leftarrow K^\top$;                                   // initialization
while not stalled do
    $\partial E_k / \partial G \leftarrow$ sparse gradient (G);
    $\Delta G \leftarrow$ NADAM($\partial E_k / \partial G$);
    $G \leftarrow G - \gamma \, \Delta G$;
    $G \leftarrow$ orthogonal projection (G, $\Phi_0$);    // see Appendix B
```

4 EVALUATION & VALIDATION

Our input is a matrix L which can be derived from a variety of geometric data types. In Fig. 13 we show that our method can preserve the property of the Laplace operators defined on triangle meshes [Desbrun et al. 1999; MacNeal 1949; Pinkall and Polthier 1993], point clouds [Belkin et al. 2009], graphs, and tetrahedral meshes [Sharf et al. 2007]. We also evaluate our method on a variety of operators, including the offset surface Laplacian [Corman et al. 2017], the Hessian of the Ricci energy [Jin et al. 2008], anisotropic Laplacian [Andreux et al. 2014], and the intrinsic Delaunay Laplacian [Fisher et al. 2006] (see Fig. 14).

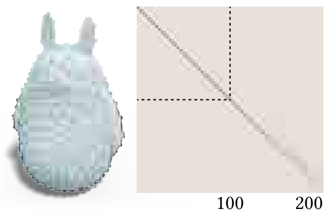


Fig. 12. We optimize for the first 100 eigenfunctions and visualize the 200×200 functional map, demonstrating a graceful generalization beyond the optimized eigenfunctions.

We further evaluate how the coarsening generalizes beyond the optimized eigenfunctions. In Fig. 12, we coarsen the shape using the first 100 eigenfunctions and visualize the 200×200 functional map image. This shows a strong diagonal for the upper 100×100 block and a slowly blurring off-diagonal for the bottom block, demonstrating a graceful generalization beyond the optimized eigenfunctions.

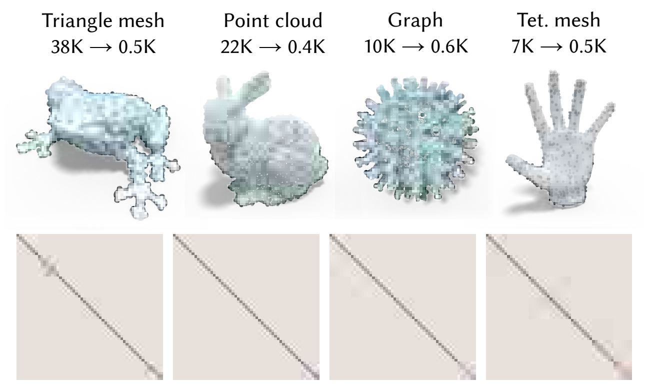


Fig. 13. Our algebraic formulation is directly applicable to different data types, such as triangle meshes, point clouds, graphs, and tetrahedral meshes.

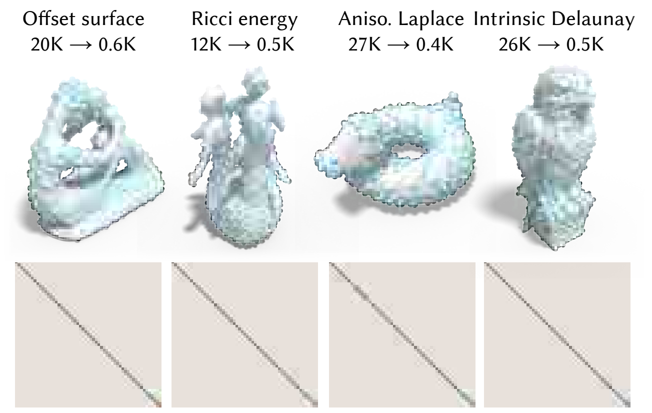


Fig. 14. Our method preserves the eigenfunctions of the offset surface Laplacian, the Hessian of the Ricci energy, the anisotropic Laplace, and the the intrinsic Delaunay Laplacian.

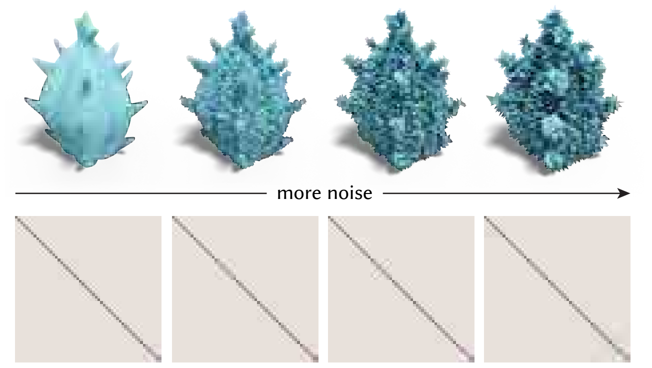


Fig. 15. Our coarsening takes the operator as the input, thus the output quality is robust to noise and sharp geometric features.

Our algebraic approach takes the operator as the input, instead of the mesh, thus the output quality is robust to noise or sharp features (see Fig. 15). In addition, we can apply our method recursively to the output operator to construct a multilevel hierarchy (see Fig. 16).

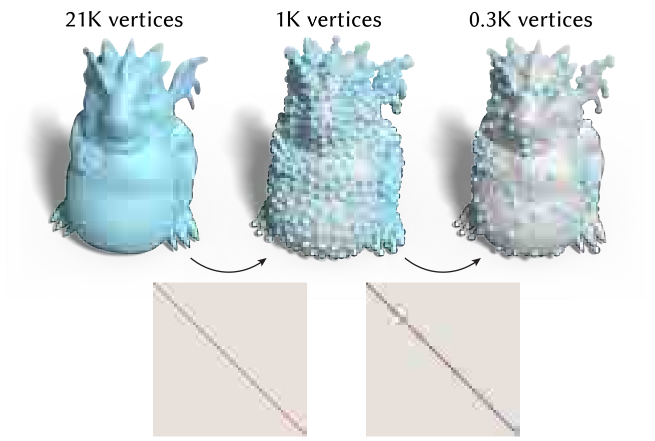


Fig. 16. We apply our approach recursively to construct a multilevel hierarchy: from 21,000 rows through 1,000 rows to finally 300 rows.

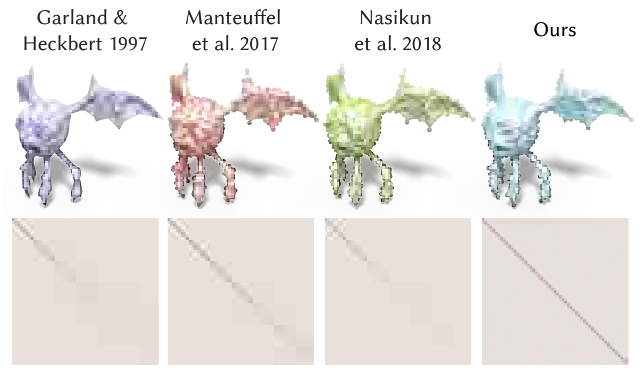


Fig. 17. We simplify the cotangent Laplacian from $n = 30,000$ to $m = 500$. Our coarsening preserve the first 200 eigenfunctions better than the QSlim [Garland and Heckbert 1997], the root-node algebraic multigrid [Manteuffel et al. 2017], and the fast approximation [Nasikun et al. 2018].

4.1 Comparisons

Existing coarsening methods are usually not designed for preserving the spectral property of operators. Geometry-based mesh decimation (i.e., *QSlim* [Garland and Heckbert 1997]) is formulated to preserve the appearance of the geometry, and results in poor performance in preserving the operator (see Fig. 1). As an iterative solver, algebraic multigrid, i.e., root-node method [Manteuffel et al. 2017], optimizes the convergence rate and does not preserve the spectral properties either. Recently, Nasikun et al. [2018] propose approximating the isotropic Laplacian based on constructing locally supported basis functions. However, this approach falls short in preserving the spectral properties of shapes with high-curvature thin structures and anisotropic operators (see Fig. 17, Fig. 18). In contrast, our proposed method can effectively preserve the eigenfunctions for both isotropic and anisotropic operators.

In addition, a simple derivation (Appendix C) can show that minimizing the proposed energy implies eigenvalue preservation (see Fig. 2 and Fig. 19).

In Fig. 20, we show that our method handles anisotropy in the input operator better than existing methods. This example also

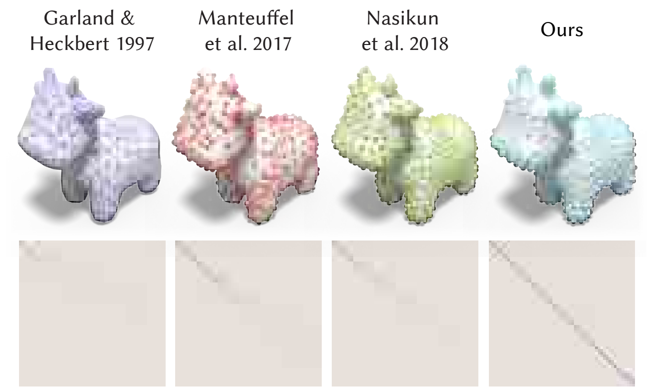


Fig. 18. We simplify the anisotropic Laplacian [Andreux et al. 2014] (with parameter 70) from $n = 25,000$ to $m = 600$. Our approach can preserve eigenfunctions of anisotropic operators better than the existing approaches.

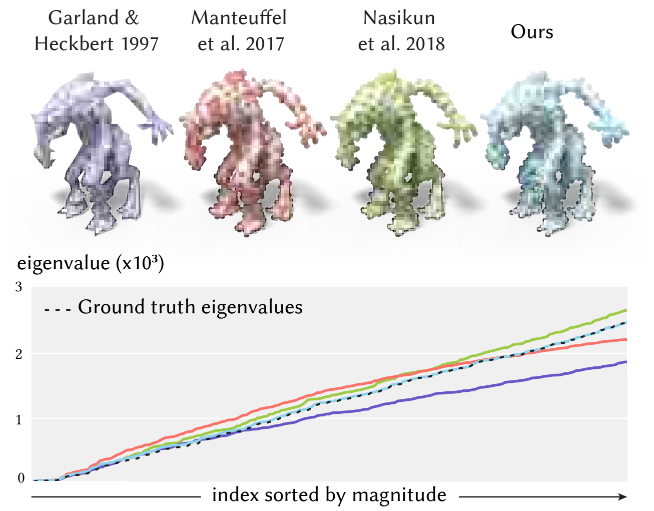


Fig. 19. We compare the performance of preserving eigenvalues with different simplification methods. As optimizing our proposed energy implies eigenvalue preservation, we show that the eigenvalues of the simplified operator is well-aligned with the original eigenvalues.

demonstrates how our method gracefully degrades as anisotropy increases. Extreme anisotropy (far right column) eventually causes our method to struggle to maintain eigenvectors.

4.2 Implementation

In general, let k be the number of eigenvectors/eigenvalues in use, we recommend to use the number of root nodes $m > k \times 2$. In Fig. 21 we show that if m is too small, the degrees of freedom are insufficient to capture the eigenfunctions with higher frequencies.

Our serial C++ implementation is built on top of LIBIGL [Jacobson et al. 2018] and SPECTRA [Qiu 2018]. We test our implementation on a Linux workstation with an Intel Xeon 3.5GHz CPU, 64GB of RAM, and an NVIDIA GeForce GTX 1080 GPU. We evaluate our runtime using the mesh from Fig. 4 in three different cases: (1) varying the size of input operators n, (2) varying the size of output operators m, and (3) varying the number of eigenvectors in use k. All experiments converge in 100-300 iterations. We report our runtime in Fig. 22. We obtain 3D shapes mainly from *Thingi10K* [Zhou and Jacobson 2016] and clean them with the method of [Hu et al. 2018].

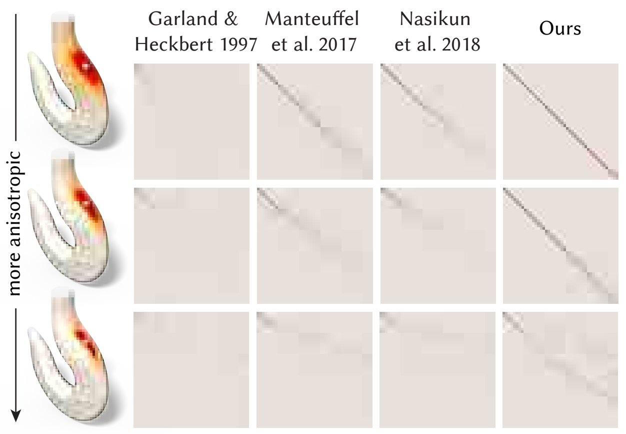


Fig. 20. We increase the anisotropy parameter of [Andreux et al. 2014] (60, 120, 180) while simplifying an operator from 21,000 rows down to 500. Our approach handles anisotropy better than existing approaches but still struggles to preserve extreme anisotropic operators.

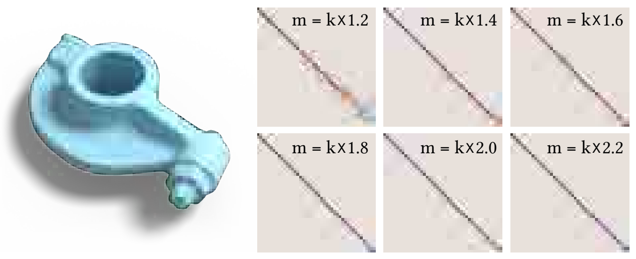


Fig. 21. Let k be the number of eigenvectors we want to preserve, experimentally we observed that $m > k \times 2$ leads to desired results.

4.3 Difference-Driven Coarsening

We also validate our combinatorial coarsening by applying it to the shape difference operator [Rustamov et al. 2013] which provides an informative representation of how two shapes differ from each other. As a positive definite operator it fits naturally into our framework. Moreover, since the difference is captured via functional maps, it does not require two shapes to have the same triangulation. We therefore take a pair of shapes with a known functional map between them, compute the shape difference operator and apply our combinatorial coarsening, while trying to best preserve this computed operator. Intuitively, we expect the samples to be informed by the shape difference and thus capture the areas of distortion between the shapes (see Appendix D for more detail). As shown in Fig. 23, our coarsening indeed leads to samples in areas where the intrinsic distortion happens, thus validating the ability of our approach to capture and reveal the characteristics of the input operator.

We can further take the element-wise maximum from a collection of shape difference operators to obtain a data-driven coarsening informed by many shape differences (see Fig. 24).

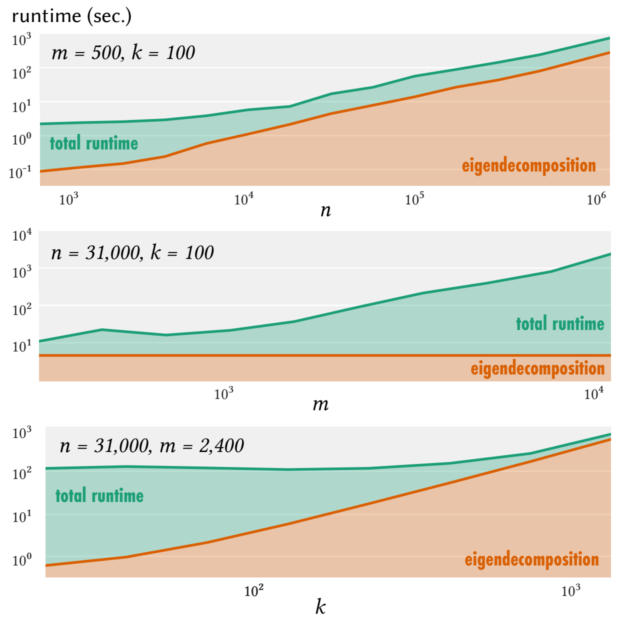


Fig. 22. Our runtime shows that our approach is more suitable for aggressive coarsening (middle). For large input meshes and many eigenvectors in use (top, bottom), computing eigendecomposition is the bottleneck.

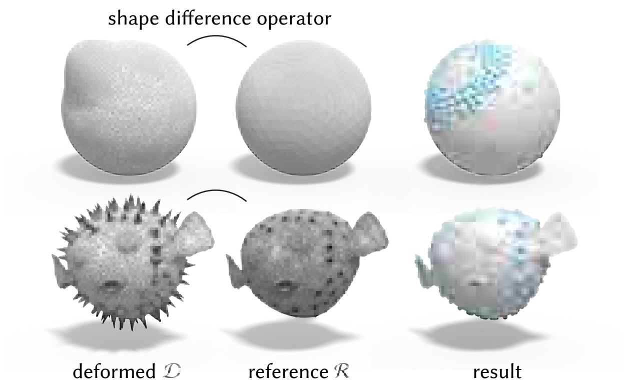


Fig. 23. Given a reference shape $\mathcal{R}$ and its deformed version $\mathcal{D}$, we combine the shape difference operator with our coarsening to compute a set of samples that capture the areas of highest distortion between the shapes.

4.4 Efficient Shape Correspondence

A key problem in shape analysis is computing correspondences between pairs of non-rigid shapes. In this application we show how our coarsening can significantly speed up existing shape matching methods while also leading to comparable or even higher accuracy. For this we use a recent iterative method based on the notion of *Product Manifold Filter* (PMF), which has shown excellent results in different shape matching applications [Vestner et al. 2017]. This method, however, suffers from high computational complexity, since it is based on solving a linear assignment problem, $O(n^3)$, at each iteration. Moreover, it requires the pair of shapes to have the same number of vertices. As a result, in practice before applying PMF

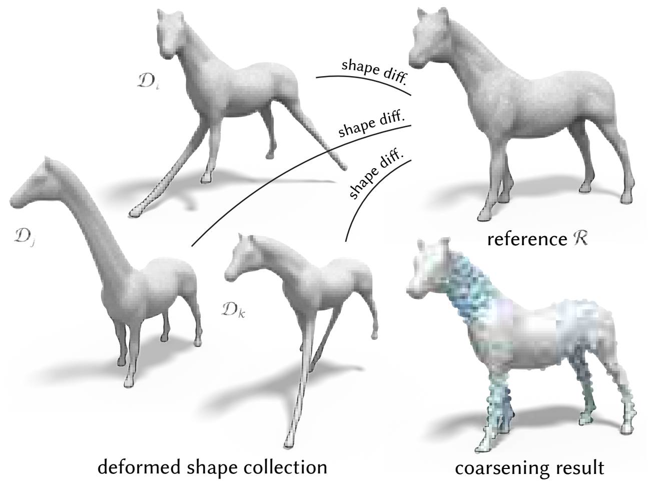


Fig. 24. By combining the shape difference operators from the reference shape $\mathcal{R}$ to a collection of deformed shapes $\mathcal{D}$, algebraic coarsening can simplify a mesh based on the “union” of all the deformations.

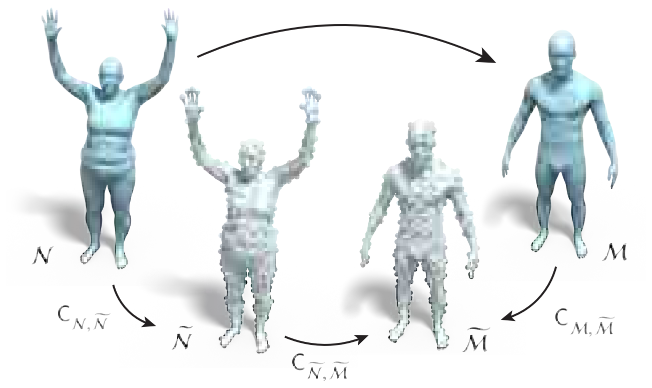

$\mathcal{N}$
$\mathcal{M}$
$C_{\mathcal{N},\widetilde{\mathcal{N}}}$
$\widetilde{\mathcal{N}}$
$C_{\widetilde{\mathcal{N}},\widetilde{\mathcal{M}}}$
$\widetilde{\mathcal{M}}$
$C_{\mathcal{M},\widetilde{\mathcal{M}}}$

Fig. 25. Efficient matching computes the map $C_{\mathcal{N},\mathcal{M}}$ between the original shapes by (1) applying our proposed coarsening to the shape pair and obtain $C_{\mathcal{N},\widetilde{\mathcal{N}}}$, $C_{\mathcal{M},\widetilde{\mathcal{M}}}$, (2) compute shape correspondences $C_{\widetilde{\mathcal{N}},\widetilde{\mathcal{M}}}$ in the reduced space, and (3) solve a linear system based on the commutative diagram.

shapes are typically subsampled to a coarser resolution and the result is then propagated back to the original meshes. For example in [Litany et al. 2017], the authors used the standard QSlim [Garland and Heckbert 1997] to simplify the meshes before matching them using PMF. Unfortunately, since standard appearance-based simplification methods can severely distort the spectral properties this can cause problems for spectral methods such as [Vestner et al. 2017] both during matching between coarse domains and while propagating back to the dense ones. Instead our spectral-based coarsening, while not resulting in a mesh provides all the necessary information to apply a spectral technique via the eigen-pairs of the coarse operator, and moreover provides an accurate way to propagate the information back to the original shapes.

More concretely, we aim to find correspondences between the coarsened shapes $\widetilde{\mathcal{N}}, \widetilde{\mathcal{M}}$ and to propagate the result back to the original domains $\mathcal{N}, \mathcal{M}$ by following a commutative diagram (see

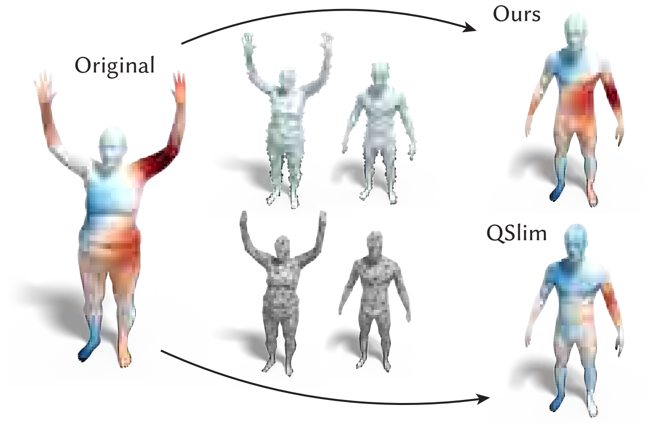


Fig. 26. Using our coarsening (top) to infer functional maps between the original pair from the coarse pair introduces less error than using the appearance-based mesh simplification (bottom), QSlim [Garland and Heckbert 1997].

Fig. 25). When all correspondences are encoded as functional maps this diagam can be written in matrix form as:

$$C_{\mathcal{M},\widetilde{\mathcal{M}}} C_{\mathcal{N},\mathcal{M}} = C_{\widetilde{\mathcal{N}},\widetilde{\mathcal{M}}} C_{\mathcal{N},\widetilde{\mathcal{N}}}, \tag{8}$$

where $C_{\mathcal{X},\mathcal{Y}}$ denotes the functional map from $\mathcal{X}$ to $\mathcal{Y}$. Using Eq. 8, the functional map $C_{\mathcal{N},\mathcal{M}}$ can be computed by solving a simple least squares problem, via a single linear solve. Our main observation is that if the original function space is preserved during the coarsening, less error will be introduced when moving across domains.

We tested this approach by evaluating a combination of our coarsening with [Vestner et al. 2017] and compared it to several baselines on a challenging non-rigid non-isometric dataset containing shapes from the SHREC 2007 contest [Giorgi et al. 2007], and evaluated the results using the landmarks and evaluation protocol from [Kim et al. 2011] (please see the details on both the exact parameters and the evaluation in the Appendix). Figure 26 shows the accuracy of several methods, both that directly operate on the dense meshes [Kim et al. 2011; Nogneng and Ovsjanikov 2017] as well as using kernel matching [Vestner et al. 2017] with QSlim and with our coarsening. The results in Figure 26 show that our approach produces maps with comparable quality or superior quality to existing methods on these non-isometric shape pairs, and results in significant improvement compared to coarsening the shapes with QSlim. At the same time, in Table 1 we report the runtime of different methods, which shows that our approach leads to a significant speed-up compared to existing techniques, and enables an efficient and accurate PMF-based matching method (see Fig. 27) with significantly speedup.

4.5 Graph Pooling

Convolutional neural networks [LeCun et al. 1998] have led to breakthroughs in image, video, and sound recognition tasks. The success of CNNs has inspired a growth of interest in generalizing CNNs to graphs and curved surfaces [Bronstein et al. 2017]. The fundamental components of a graph CNN are the *pooling* and the *convolution*. Our root node representation P, K defines a way of performing pooling on graphs. Meanwhile, our output $\widetilde{L}$ facilitates

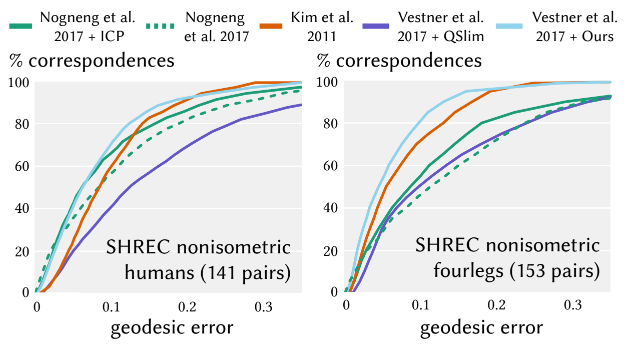


Fig. 27. Our efficient shape correspondence with kernel matching [Vestner et al. 2017] achieves comparable matching quality on many non-isometric shape pairs from SHREC [Giorgi et al. 2007] dataset with methods that directly operator on dense meshes [Kim et al. 2011; Nogneng and Ovsjanikov 2017].

graph convolution on the coarsened graph due to the convolution theorem [Arfken and Weber 1999].

To evaluate the performance of graph pooling, we construct several mesh EMNIST datasets where each mesh EMNIST digit is stored as a real-value function on a triangle mesh. Each mesh EMNIST dataset is constructed by overlaying a triangle mesh with the original EMNIST letters [Cohen et al. 2017]. We compare our graph pooling with the graph pooling IN [Defferrard et al. 2016] by evaluating the classification performance. For the sake of fair comparisons, we use the same graph Laplacian, the same architecture, and the smae hyperparameters. The only difference is the graph pooling module. In addition to EMNIST, we evaluate the performance on the fashion-MNIST dataset [Xiao et al. 2017] under the same settings. In Fig. 28, we show that our graph pooling results in better training and testing performance. We provide implementation details in Appendix F.

5 LIMITATIONS & FUTURE WORK

Reconstructing a valid mesh from our output coarsened operator would enable more downstream applications. Incorporating a fast eigen-approximation or removing the use of eigen decomposition would further scale the spectral coarsening. Moreover, exploring sparse SDP methods (e.g. [Sun and Vandenberghe 2015]) could improve our operator optimization. Jointly optimizing the sparsity and the operator entries may lead to even better solutions. Further restricting the sparsity pattern of the coarsened operator while maintaining the performance would aid to the construction of a deeper

Table 1. We report the total time, pre-computation time + runtime, for computing a 60-by-60 functional map on a shape pair with 14,000 vertices each. Our pre-computation time will be amortized by the number of pairs because we apply coarsening on each shape independently, and the number of combinations is quadratic in the number of shapes. Our runtime is orders of magnitude faster because we only need to perform shape matching in the coarsened domain (i.e., 300 vertices).

[Nogneng 17]+ICP	*[Nogneng 17]*	*[Kim 11]*	*[Vestner 17]+ours*
32.4 *sec*	4.6 *sec*	90.6 *sec*	10.8+**0.3** *sec*

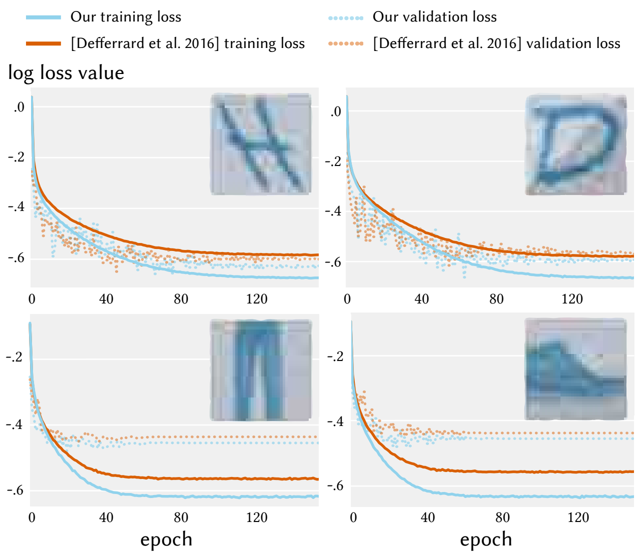


Fig. 28. Graph pooling using our coarsening performs better than the pooling presented in [Defferrard et al. 2016] on classifying the mesh EMNIST (top row) and the mesh fashion-MNIST (bottom row) datasets.

multilevel representation, which could aid in developing a hierarchical graph representation for graph neural networks. A scalable coarsening with a deeper multilevel representation could promote a multigrid solver for geometry processing applications. Generalizing to quaternions would extend the realm of our coarsening to the fields that deal with quaternionic operators.

6 ACKNOWLEDGMENTS

This work is funded in part by by a Google Focused Research Award, KAUST OSR Award (OSR-CRG2017-3426), ERC Starting Grant No. 758800 (EXPROTEA), NSERC Discovery Grants (RGPIN2017-05235), NSERC DAS (RGPAS-2017-507938), the Canada Research Chairs Program, Fields Institute CQAM Labs, Mitacs Globalink, and gifts from Autodesk, Adobe, MESH, and NVIDIA Corporation. We thank members of Dynamic Graphics Project at the University of Toronto and STREAM group at the École Polytechnique for discussions and draft reviews; Eitan Grinspun, Etienne Corman, Jean-Michel Roufosse, Maxime Kirgo, and Silvia Sellán for proofreading; Ahmad Nasikun, Desai Chen, Dingzeyu Li, Gavin Barill, Jing Ren, and Ruqi Huang for sharing implementations and results; Zih-Yin Chen for early discussions.

REFERENCES

Martin S. Andersen, Joachim Dahl, and Lieven Vandenberghe. 2010. Implementation of nonsymmetric interior-point methods for linear optimization over sparse matrix cones. *Mathematical Programming Computation* 2, 3 (Dec 2010).

Mathieu Andreux, Emanuele Rodola, Mathieu Aubry, and Daniel Cremers. 2014. Anisotropic Laplace-Beltrami operators for shape analysis. In *European Conference on Computer Vision*. Springer, 299–312.

George B Arfken and Hans J Weber. 1999. Mathematical methods for physicists.

Mathieu Aubry, Ulrich Schlickewei, and Daniel Cremers. 2011. The wave kernel signature: A quantum mechanical approach to shape analysis. In *Computer Vision Workshops (ICCV Workshops), 2011 IEEE International Conference on*. IEEE, 1626–1633.

Mikhail Belkin, Jian Sun, and Yusu Wang. 2009. Constructing Laplace operator from point clouds in R d. In *Proceedings of the twentieth annual ACM-SIAM symposium on Discrete algorithms*. Society for Industrial and Applied Mathematics, 1031–1040.

William N Bell. 2008. Algebraic multigrid for discrete differential forms. (2008).

Gaurav Bharaj, David IW Levin, James Tompkin, Yun Fei, Hanspeter Pfister, Wojciech Matusik, and Changxi Zheng. 2015. Computational design of metallophone contact sounds. *ACM Transactions on Graphics (TOG)* 34, 6 (2015), 223.

Stephen Boyd and Lieven Vandenberghe. 2004. *Convex optimization*. Cambridge university press.

William L Briggs, Steve F McCormick, et al. 2000. *A multigrid tutorial*. Vol. 72. Siam.

Michael M Bronstein, Joan Bruna, Yann LeCun, Arthur Szlam, and Pierre Vandergheynst. 2017. Geometric deep learning: going beyond euclidean data. *IEEE Signal Processing Magazine* 34, 4 (2017), 18–42.

Peter Burt and Edward Adelson. 1983. The Laplacian pyramid as a compact image code. *IEEE Transactions on communications* 31, 4 (1983), 532–540.

Desai Chen, David I. Levin, Wojciech Matusik, and Danny M. Kaufman. 2017. Dynamics-Aware Numerical Coarsening for Fabrication Design. *ACM Trans. Graph.* 34, 4 (2017).

Desai Chen, David I. W. Levin, Shinjiro Sueda, and Wojciech Matusik. 2015. Data-driven Finite Elements for Geometry and Material Design. *ACM Trans. Graph.* 34, 4 (2015).

Jiong Chen, Hujun Bao, Tianyu Wang, Mathieu Desbrun, and Jin Huang. 2018. Numerical Coarsening Using Discontinuous Shape Functions. *ACM Trans. Graph.* 37, 4 (2018).

Jie Chen and Ilya Safro. 2011. Algebraic distance on graphs. *SIAM Journal on Scientific Computing* 33, 6 (2011), 3468–3490.

Paolo Cignoni, Marco Callieri, Massimiliano Corsini, Matteo Dellepiane, Fabio Ganovelli, and Guido Ranzuglia. 2008. Meshlab: an open-source mesh processing tool.. In *Eurographics Italian chapter conference*, Vol. 2008. 129–136.

Paolo Cignoni, Claudio Montani, and Roberto Scopigno. 1998. A comparison of mesh simplification algorithms. *Computers & Graphics* 22, 1 (1998), 37–54.

Gregory Cohen, Saeed Afshar, Jonathan Tapson, and André van Schaik. 2017. EMNIST: an extension of MNIST to handwritten letters. *arXiv preprint arXiv:1702.05373* (2017).

David Cohen-Steiner, Pierre Alliez, and Mathieu Desbrun. 2004. Variational shape approximation. *ACM Trans. on Graph.* (2004).

Etienne Corman, Justin Solomon, Mirela Ben-Chen, Leonidas Guibas, and Maks Ovsjanikov. 2017. Functional characterization of intrinsic and extrinsic geometry. *ACM Transactions on Graphics (TOG)* 36, 2 (2017), 14.

Michaël Defferrard, Xavier Bresson, and Pierre Vandergheynst. 2016. Convolutional neural networks on graphs with fast localized spectral filtering. In *Advances in Neural Information Processing Systems*. 3844–3852.

Mathieu Desbrun, Mark Meyer, Peter Schröder, and Alan H Barr. 1999. Implicit fairing of irregular meshes using diffusion and curvature flow. In *Proceedings of the 26th annual conference on Computer graphics and interactive techniques*. ACM Press/Addison-Wesley Publishing Co., 317–324.

Timothy Dozat. 2016. Incorporating nesterov momentum into adam. (2016).

Matthew Fisher, Boris Springborn, Alexander I Bobenko, and Peter Schroder. 2006. An algorithm for the construction of intrinsic Delaunay triangulations with applications to digital geometry processing. In *ACM SIGGRAPH 2006 Courses*. ACM, 69–74.

Michael Garland and Paul S Heckbert. 1997. Surface simplification using quadric error metrics. In *Proceedings of the 24th annual conference on Computer graphics and interactive techniques*. ACM Press/Addison-Wesley Publishing Co., 209–216.

Michael Garland and Paul S Heckbert. 1998. Simplifying surfaces with color and texture using quadric error metrics. In *Visualization'98. Proceedings*. IEEE, 263–269.

Daniela Giorgi, Silvia Biasotti, and Laura Paraboschi. 2007. SHape REtrieval Contest 2007: Watertight Models Track. http://watertight.ge.imati.cnr.it/.

Igor Guskov, Andrei Khodakovsky, Peter Schröder, and Wim Sweldens. 2002. Hybrid meshes: multiresolution using regular and irregular refinement. In *Proceedings of the eighteenth annual symposium on Computational geometry*. ACM, 264–272.

Hugues Hoppe. 1996. Progressive meshes. In *Proceedings of the 23rd annual conference on Computer graphics and interactive techniques*. ACM, 99–108.

Hugues Hoppe. 1999. New quadric metric for simplifying meshes with appearance attributes. In *Visualization'99. Proceedings*. IEEE, 59–510.

Hugues Hoppe, Tony DeRose, Tom Duchamp, John McDonald, and Werner Stuetzle. 1993. Mesh Optimization. In *Proc. SIGGRAPH*. 19–26.

Yixin Hu, Qingnan Zhou, Xifeng Gao, Alec Jacobson, Denis Zorin, and Daniele Panozzo. 2018. Tetrahedral meshing in the wild. *ACM Transactions on Graphics (TOG)* 37, 4 (2018), 60.

Alec Jacobson, Daniele Panozzo, et al. 2018. libigl: A simple C++ geometry processing library. http://libigl.github.io/libigl/.

Miao Jin, Junho Kim, Feng Luo, and Xianfeng Gu. 2008. Discrete surface Ricci flow. *IEEE Transactions on Visualization and Computer Graphics* 14, 5 (2008), 1030–1043.

Karsten Kahl and Matthias Rottmann. 2018. Least Angle Regression Coarsening in Bootstrap Algebraic Multigrid. *arXiv preprint arXiv:1802.00595* (2018).

Lily Kharevych, Patrick Mullen, Houman Owhadi, and Mathieu Desbrun. 2009. Numerical coarsening of inhomogeneous elastic materials. *ACM Trans. on Graph.* (2009).

Vladimir G Kim, Yaron Lipman, and Thomas Funkhouser. 2011. Blended intrinsic maps. In *ACM Transactions on Graphics (TOG)*, Vol. 30. ACM, 79.

Rasmus Kyng and Sushant Sachdeva. 2016. Approximate gaussian elimination for laplacians-fast, sparse, and simple. In *Foundations of Computer Science (FOCS), 2016 IEEE 57th Annual Symposium on.* IEEE, 573–582.

Yann LeCun, Léon Bottou, Yoshua Bengio, and Patrick Haffner. 1998. Gradient-based learning applied to document recognition. *Proc. IEEE* 86, 11 (1998), 2278–2324.

Dingzeyu Li, Yun Fei, and Changxi Zheng. 2015. Interactive Acoustic Transfer Approximation for Modal Sound. *ACM Trans. Graph.* 35, 1 (2015).

Or Litany, Tal Remez, Emanuele Rodolà, Alexander M Bronstein, and Michael M Bronstein. 2017. Deep Functional Maps: Structured Prediction for Dense Shape Correspondence.. In *ICCV*. 5660–5668.

Oren E Livne and Achi Brandt. 2012. Lean algebraic multigrid (LAMG): Fast graph Laplacian linear solver. *SIAM Journal on Scientific Computing* 34, 4 (2012), B499–B522.

Richard H MacNeal. 1949. *The solution of partial differential equations by means of electrical networks.* Ph.D. Dissertation. California Institute of Technology.

Thomas A Manteuffel, Luke N Olson, Jacob B Schroder, and Ben S Southworth. 2017. A Root-Node–Based Algebraic Multigrid Method. *SIAM Journal on Scientific Computing* 39, 5 (2017), S723–S756.

Ahmad Nasikun, Christopher Brandt, and Klaus Hildebrandt. 2018. Fast Approximation of Laplace-Beltrami Eigenproblems. In *Computer Graphics Forum*, Vol. 37. Wiley Online Library, 121–134.

Dorian Nogneng and Maks Ovsjanikov. 2017. Informative descriptor preservation via commutativity for shape matching. In *Computer Graphics Forum*, Vol. 36. Wiley Online Library, 259–267.

Luke N Olson, Jacob Schroder, and Raymond S Tuminaro. 2010. A new perspective on strength measures in algebraic multigrid. *Numerical Linear Algebra with Applications* 17, 4 (2010), 713–733.

Maks Ovsjanikov, Mirela Ben-Chen, Justin Solomon, Adrian Butscher, and Leonidas Guibas. 2012. Functional maps: a flexible representation of maps between shapes. *ACM Transactions on Graphics (TOG)* 31, 4 (2012), 30.

A Cengiz Öztireli, Marc Alexa, and Markus Gross. 2010. Spectral sampling of manifolds. In *ACM Transactions on Graphics (TOG)*, Vol. 29. ACM, 168.

Gabriel Peyré and Stéphane Mallat. 2005. Surface compression with geometric bandelets. In *ACM Transactions on Graphics (TOG)*, Vol. 24. ACM, 601–608.

Ulrich Pinkall and Konrad Polthier. 1993. Computing discrete minimal surfaces and their conjugates. *Experimental mathematics* 2, 1 (1993), 15–36.

Jovan Popović and Hugues Hoppe. 1997. Progressive simplicial complexes. In *Proceedings of the 24th annual conference on Computer graphics and interactive techniques.* ACM Press/Addison-Wesley Publishing Co., 217–224.

Yixuan Qiu. 2018. spectra: C++ Library For Large Scale Eigenvalue Problems. https://github.com/yixuan/spectra/.

John W Ruge and Klaus Stüben. 1987. Algebraic multigrid. In *Multigrid methods.* SIAM, 73–130.

Raif M Rustamov, Maks Ovsjanikov, Omri Azencot, Mirela Ben-Chen, Frédéric Chazal, and Leonidas Guibas. 2013. Map-based exploration of intrinsic shape differences and variability. *ACM Transactions on Graphics (TOG)* 32, 4 (2013), 72.

Peter Schroder. 1996. Wavelets in computer graphics. *Proc. IEEE* 84, 4 (1996), 615–625.

William J Schroeder, Jonathan A Zarge, and William E Lorensen. 1992. Decimation of triangle meshes. In *ACM siggraph computer graphics*, Vol. 26. ACM, 65–70.

Andrei Sharf, Thomas Lewiner, Gil Shklarski, Sivan Toledo, and Daniel Cohen-Or. 2007. Interactive topology-aware surface reconstruction. *ACM Trans. on Graph.* (2007).

Anja Struyf, Mia Hubert, and Peter Rousseeuw. 1997. Clustering in an Object-Oriented Environment. *Journal of Statistical Software* (1997).

Klaus Stuben. 2000. Algebraic multigrid (AMG): an introduction with applications. *Multigrid* (2000).

Yifan Sun and Lieven Vandenberghe. 2015. Decomposition methods for sparse matrix nearness problems. *SIAM J. Matrix Anal. Appl.* 36, 4 (2015), 1691–1717.

Rasmus Tamstorf, Toby Jones, and Stephen F McCormick. 2015. Smoothed aggregation multigrid for cloth simulation. *ACM Trans. on Graph.* (2015).

Lieven Vandenberghe and Martin S. Andersen. 2015. Chordal Graphs and Semidefinite Optimization. *Found. Trends Optim.* 1, 4 (2015).

Matthias Vestner, Zorah Lähner, Amit Boyarski, Or Litany, Ron Slossberg, Tal Remez, Emanuele Rodolà, Alexander M. Bronstein, Michael M. Bronstein, Ron Kimmel, and Daniel Cremers. 2017. Efficient Deformable Shape Correspondence via Kernel Matching. In *3DV*.

Han Xiao, Kashif Rasul, and Roland Vollgraf. 2017. Fashion-MNIST: a Novel Image Dataset for Benchmarking Machine Learning Algorithms. arXiv:cs.LG/cs.LG/1708.07747

Jinchao Xu and Ludmil Zikatanov. 2017. Algebraic multigrid methods. *Acta Numerica* 26 (2017), 591–721.

Yang Zheng, Giovanni Fantuzzi, Antonis Papachristodoulou, Paul Goulart, and Andrew Wynn. 2017. Fast ADMM for semidefinite programs with chordal sparsity. In *American Control Conference.*

Qingnan Zhou and Alec Jacobson. 2016. Thingi10K: A Dataset of 10,000 3D-Printing Models. *arXiv preprint arXiv:1605.04797* (2016).

A DERIVATIVE WITH RESPECT TO SPARSE G

To use a gradient-based solver in Algorithm 2 to solve the optimization problem in Eq. 7 we need derivatives with respect to the non-zeros in G (the sparsity of S_G). We start with the dense gradient:

$$\frac{\partial E_k}{\partial G} = \frac{\partial}{\partial G}\frac{1}{2}\|PM^{-1}L\Phi - \widetilde{M}^{-1}G^\top LGP\Phi\|^2_{\widetilde{M}}.$$

We start the derivation by introducing two constant variables A, B to simplify the expression

$$\begin{aligned}\frac{\partial E_k}{\partial G} &= \frac{\partial}{\partial G}\frac{1}{2}\|A - \widetilde{M}^{-1}G^\top LGB\|^2_{\widetilde{M}} \\ A &= PM^{-1}L\Phi,\ B = P\Phi.\end{aligned}$$

Using the fact that L, M, M_c are symmetric matrices and the rules in matrix trace derivative, we expand the equation as follows.

$$\begin{aligned}\frac{\partial E_k}{\partial G} &= \frac{\partial}{\partial G}\frac{1}{2}\|A - \widetilde{M}^{-1}G^\top LGB\|^2_{\widetilde{M}} \\ &= \frac{\partial}{\partial G}\frac{1}{2}\operatorname{tr}\left((A^\top - B^\top G^\top LG\widetilde{M}^{-1})\widetilde{M}(A - \widetilde{M}^{-1}G^\top LGB)\right) \\ &= -\frac{\partial}{\partial G}\left(\operatorname{tr}\left(B^\top G^\top LGA\right) + \frac{1}{2}\operatorname{tr}\left(B^\top G^\top LG\widetilde{M}^{-1}G^\top LGB\right)\right) \\ &= -\left(LGAB^\top + LGBA^\top\right) \\ &\quad + \left(LG\widetilde{M}^{-1}G^\top LGBB^\top + LGBB^\top G^\top LG\widetilde{M}^{-1}\right) \\ &= LG(-AB^\top - BA^\top + \widetilde{M}^{-1}G^\top LGBB^\top + BB^\top G^\top LG\widetilde{M}^{-1})\end{aligned}$$

Computing the $\partial E_k/\partial G$ subject to the sparsity S_G can be naively achieved by first computing the dense gradient $\partial E_k/\partial G$ and then project to the sparsity constraint through an element-wise product with the sparsity S_G. However, the naive computation would waste a large amount of computational resources on computing gradient values that do not satisfy the sparsity. We incorporate the sparsity and compute gradients only for the non-zero entries as

$$\begin{aligned}&\left(\frac{\partial E_k}{\partial G}\right)_{ij} = Y_{i*}Z_{*j}, \quad i,j \in S_G \\ &Y = LG \\ &Z = -AB^\top - BA^\top + \widetilde{M}^{-1}G^\top LGBB^\top + BB^\top G^\top LG\widetilde{M}^{-1},\end{aligned}$$

where Y_{i*}, Z_{*j} denote the ith row of Y and the jth column of Z.

B SPARSE ORTHOGONAL PROJECTION

Let $\mathbf{g} \in \mathbb{R}^z$ be the vector of non-zeros in G, so that $\operatorname{vec}(G) = Z\mathbf{g}$, where $Z \in \{0,1\}^{mn\times z}$ scatter matrix.

Given some G_1 that does not satisfy our constraints, we would like to find its closest projection onto the matrices that do satisfy the constraints. In other words, we aim at solving:

$$\begin{aligned}&\underset{G \doteq S_G}{\text{minimize}}\ \|G - G_1\| \\ &\text{subject to } GP\Phi_0 - \Phi_0 = 0.\end{aligned}$$

Using properties of the vectorization and the Kronecker product, we can now write this in terms of vectorization:

$$\begin{aligned}&\underset{\mathbf{g}}{\text{minimize}}\ \|\mathbf{g} - \mathbf{g}_1\| \\ &\text{subject to } ((P\Phi_0)^\top \otimes \mathrm{id}_m)Z\mathbf{g} - \operatorname{vec}(\Phi_0) = 0.\end{aligned}$$

whose solution is given as:

$$\begin{aligned}\mathbf{g} &= \mathbf{g}_1 - \mathsf{A}^\top(\mathsf{A}\mathsf{A}^\top)^{-1}\mathbf{b}\\ \mathsf{A} &= ((\mathsf{P}\Phi_0)^\top \otimes \mathrm{id}_m)\mathsf{Z}\\ \mathbf{b} &= ((\mathsf{P}\Phi_0)^\top \otimes \mathrm{id}_m)\mathsf{Z}\mathbf{g}_1 - \mathrm{vec}(\Phi_0).\end{aligned}$$

This can be simplified to an element-wise division when Φ_0 is a single vector.

C EIGENVALUE PRESERVATION

Minimizing the commutative energy Eq. 4 implies

$$\mathsf{P}\mathsf{M}^{-1}\mathsf{L}\Phi_i = \widetilde{\mathsf{M}}^{-1}\widetilde{\mathsf{L}}\mathsf{P}\Phi_i. \tag{9}$$

As Φ_i comes from solving the generalized eigenvalue problem, for every Φ_i we must have: $\mathsf{L}\Phi_i = \lambda_i\mathsf{M}\Phi_i$. Therefore, Eq. 9 implies $\lambda_i\mathsf{P}\mathsf{M}^{-1}\Phi_i = \widetilde{\mathsf{M}}^{-1}\widetilde{\mathsf{L}}\mathsf{P}\Phi_i$, which means that $\mathsf{P}\Phi_i$ must be an eigenvector of $\widetilde{\mathsf{M}}^{-1}\widetilde{\mathsf{L}}$ corresponding to *the same eigenvalue* λ_i.

D MODIFIED SHAPE DIFFERENCE

Rustamov et al. [2013] capture geometric distortions by tracking the inner products of real-valued functions induced by transporting these functions from one shape to another one via a functional map [Ovsjanikov et al. 2012]. This formulation allows us to compare shapes with different triangulations and encode the shape difference between two shapes using a single matrix. Given a functional map C between a reference shape $\mathcal{R}$ and a deformed shape $\mathcal{D}$, the area-based shape difference operator A can be written as ([Rustamov et al. 2013] option 2)

$$\mathsf{A}_{\mathcal{R},\mathcal{D}} = \mathsf{C}^\top\mathsf{C},$$

where C is the functional map from functions on $\mathcal{R}$ to functions on $\mathcal{D}$. The operator $\mathsf{A}_{\mathcal{R},\mathcal{D}}$ encodes the difference between $\mathcal{R}$ and $\mathcal{D}$. Its eigenvectors corresponding to eigenvalues larger than one encode area-expanding regions; its eigenvectors corresponding to eigenvalues smaller than one encode area-shrinking regions.

Motivated by the goal of producing denser samples in the parts undergoing shape changes, no matter the area is expanding or shrinking, our modified shape difference operator $\tilde{\mathsf{A}}$ has the form

$$\tilde{\mathsf{A}}_{\mathcal{R},\mathcal{D}} = (\mathrm{id} - \mathsf{A}_{\mathcal{R},\mathcal{D}})^\top(\mathrm{id} - \mathsf{A}_{\mathcal{R},\mathcal{D}}).$$

This formulation treats area expanding and shrinking equally, the eigenvectors of eigenvalues larger than zero capture where the shapes differ.

Note that this $\tilde{\mathsf{A}}$ is a size k-by-k matrix where k is the number of bases in use. We map $\tilde{\mathsf{A}}$ back to the original domain by

$$\tilde{\mathsf{A}}^{\mathrm{orig}}_{\mathcal{R},\mathcal{D}} = \Phi_{\mathcal{R}}\tilde{\mathsf{A}}_{\mathcal{R},\mathcal{D}}\Phi^\top_{\mathcal{R}}.$$

Although the $\tilde{\mathsf{A}}^{\mathrm{orig}}_{\mathcal{R},\mathcal{D}}$ is dense and lots of components do not correspond to any edge in the triangle mesh, the non-zero components corresponding to actual edges contain information induced by the operator. Therefore by extracting the inverse of the off-diagonal components of $\tilde{\mathsf{A}}^{\mathrm{orig}}_{\mathcal{R},\mathcal{D}}$ that correspond to actual edges as the $-\mathsf{L}_{ij}$, we can obtain a coarsening result induced by shape differences.

E EFFICIENT SHAPE CORRESPONDENCE

We obtain dense functional maps from coarse ones by solving

$$\mathsf{C}_{\mathcal{M},\widetilde{\mathcal{M}}}\mathsf{C}_{\mathcal{N},\mathcal{M}} = \mathsf{C}_{\widetilde{\mathcal{N}},\widetilde{\mathcal{M}}}\mathsf{C}_{\mathcal{N},\widetilde{\mathcal{N}}}, \tag{10}$$

where $\mathsf{C}_{\mathcal{N},\mathcal{M}}, \mathsf{C}_{\mathcal{M},\widetilde{\mathcal{M}}}, \mathsf{C}_{\widetilde{\mathcal{N}},\widetilde{\mathcal{M}}}$ are functional maps represented in the Laplace basis. $\mathsf{C}_{\widetilde{\mathcal{N}},\widetilde{\mathcal{M}}}$ is the functional map of functions stored in the hat basis. To distinguish $\mathsf{C}_{\widetilde{\mathcal{N}},\widetilde{\mathcal{M}}}$ from others, we use $\mathsf{T}_{\widetilde{\mathcal{N}},\widetilde{\mathcal{M}}}$ to represent the map in the hat basis. Eq. 10 can be re-written as

$$\Phi_{\widetilde{\mathcal{M}}}\mathsf{C}_{\mathcal{M},\widetilde{\mathcal{M}}}\mathsf{C}_{\mathcal{N},\mathcal{M}} = \mathsf{T}_{\widetilde{\mathcal{N}},\widetilde{\mathcal{M}}}\Phi_{\widetilde{\mathcal{N}}}\mathsf{C}_{\mathcal{N},\widetilde{\mathcal{N}}},$$

where Φ are eigenvectors of the Laplace-Beltrami operator. Then we can solve the dense map by, for example, the MATLAB backslash.

$$\mathsf{C}_{\mathcal{N},\mathcal{M}} = (\Phi_{\widetilde{\mathcal{M}}}\mathsf{C}_{\mathcal{M},\widetilde{\mathcal{M}}}) \setminus (\mathsf{T}_{\widetilde{\mathcal{N}},\widetilde{\mathcal{M}}}\Phi_{\widetilde{\mathcal{N}}}\mathsf{C}_{\mathcal{N},\widetilde{\mathcal{N}}}),$$

Due to limited computational power, we often use truncated eigenvectors and functional maps. To avoid the truncation error destroys the map inference, we use rectangular wide functional maps for both $\mathsf{C}_{\mathcal{M},\widetilde{\mathcal{M}}}, \mathsf{C}_{\mathcal{N},\widetilde{\mathcal{N}}}$ to obtain a smaller squared $\mathsf{C}_{\mathcal{N},\mathcal{M}}$. For instance, the experiments in Fig. 27 use size 120-by-200 for both $\mathsf{C}_{\mathcal{M},\widetilde{\mathcal{M}}}, \mathsf{C}_{\mathcal{N},\widetilde{\mathcal{N}}}$, and we only take the top left 120-by-120 block of $\mathsf{C}_{\mathcal{N},\mathcal{M}}$ in use.

To compute $\mathsf{C}_{\widetilde{\mathcal{N}},\widetilde{\mathcal{M}}}$ (or $\mathsf{T}_{\widetilde{\mathcal{N}},\widetilde{\mathcal{M}}}$), we normalize the shape to have surface area 2,500 for numerical reasons, coarsen the shapes down to 500 vertices, and use the kernel matching [Vestner et al. 2017] for finding bijective correspondences. We use $\alpha = 0.01$ (the same notation as [Vestner et al. 2017]) to weight the pointwise descriptor, specifically the wave kernel signature [Aubry et al. 2011]; we use time parameter $0.01 \times$ surface area for the heat kernel pairwise descriptors; we use 7 landmarks for shape pairs in the SHREC dataset.

F GRAPH POOLING IMPLEMENTATION DETAIL

We use the LeNet-5 network architecture, the same as the one used in [Defferrard et al. 2016], to test our graph pooling on the mesh EMNIST [Cohen et al. 2017] and mesh fashion-MNIST [Xiao et al. 2017] datasets. Specifically, the network has 32, 64 feature maps at the two convolutional layers respectively, and a fully connected layer attached after the second convolutional layer with size 512. We use dropout probability 0.5, regularization weight 5e-4, initial learning rate 0.05, learning rate decay 0.95, batch size 100, and train for 150 epochs using SGD optimizer with momentum 0.9. The graph filters have the support of 25, and each average pooling reduces the mesh size to roughly 1/8 of the size before pooling.

Our mesh is generated by the Poisson disk sampling followed by the Delaunay triangulation and a planar flipping optimization implemented in MeshLab [Cignoni et al. 2008]. We also perform local midpoint upsampling to construct meshes with non-uniform discretizations. Then EMNIST letters are "pasted" to the triangle mesh using linear interpolation.