Trading Spaces: Adaptive Subspace Time Integration for Contacting Elastodynamics

TY TRUSTY, University of Toronto, Canada and Adobe, USA
YUN (RAYMOND) FEI, Adobe, USA
DAVID I.W. LEVIN, University of Toronto, Canada and NVIDIA, Canada
DANNY M. KAUFMAN, Adobe, USA

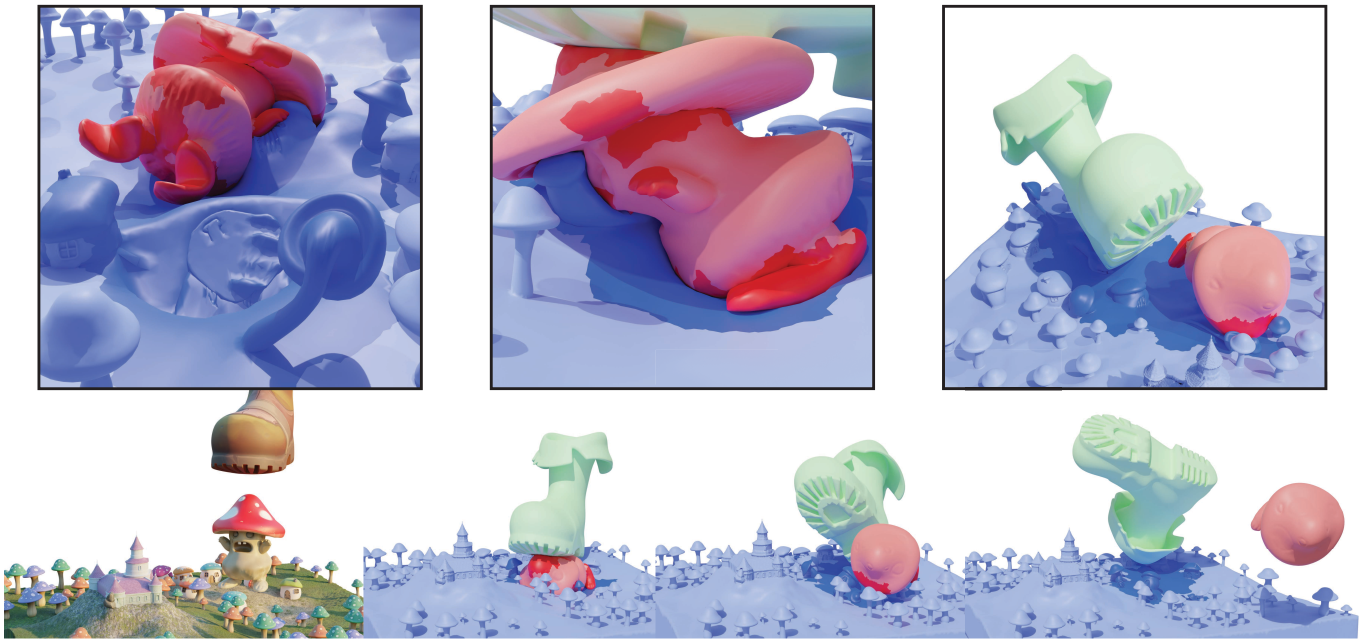

Fig. 1. **Mushroom Madness:** Adaptive subspace simulation captures the challenging detailed deformations and complex heterogeneous material interactions as a rubber boot descends to stomp on the mushroom kingdom. In this large-scale simulation with 2.5M tets adaptive subspace simulation locally updates both a subspace modal basis and local nodal refinements (dark-shaded) to carefully capture necessary local deformations. Here, as the shoe drops on the mushroom guy, adaptivity enables it to squash the ground with sharp detail (note the boot pattern captured by enrichment), as boot and mushroom guy then tumble across the kingdom, our method's oracle progressively updates and downdates both nodes and modes to track the local deformations of the kingdom's homes, mushrooms and landscape, resulting in an order-of-magnitude speed-up over an equivalent full-space simulation with comparable visual quality.

We construct a subspace simulator that adaptively balances solution improvement against system size. The core components of our simulator are an adaptive subspace oracle, model, and parallel time-step solver algorithm. Our in-time-step adaptivity oracle continually assesses subspace solution quality and candidate update proposals while accounting for temporal variations in deformation and spatial variations in material. In turn our adaptivity model is subspace agnostic. It allows application across subspace representations and expresses unrestricted deformations independent of subspace choice. We couple our oracle and model with a custom-constructed parallel time-step solver for our enriched systems that exposes a pair of user tolerances which provide controllable simulation quality. As tolerances are tightened our model converges to full-space solutions (with expected cost increases). On the other hand, as tolerances are relaxed we obtain output-bound simulation costs. We demonstrate the efficacy of our approach across a wide range of challenging nonlinear materials models, material stiffnesses, heterogeneities, dynamic behaviors, and frictionally contacting conditions, obtaining scalable and efficient simulations of complex elastodynamic scenarios.

CCS Concepts: • **Computing methodologies → Physical simulation**.

Additional Key Words and Phrases: Adaptive Subspace Simulation, Nonlinear Elastodynamics

Authors' addresses: Ty Trusty, trusty@cs.toronto.edu, University of Toronto, Canada and Adobe, USA; Yun (Raymond) Fei, yfei@adobe.com, Adobe, USA; David I.W. Levin, diwlevin@cs.toronto.edu, University of Toronto, Canada and NVIDIA, Canada; Danny M. Kaufman, dannykaufman@gmail.com, Adobe, USA.

ACM 0730-0301/2024/12-ART
https://doi.org/10.1145/3687946

ACM Reference Format:
Ty Trusty, Yun (Raymond) Fei, David I.W. Levin, and Danny M. Kaufman. 2024. Trading Spaces: Adaptive Subspace Time Integration for Contacting Elastodynamics. *ACM Trans. Graph.* 43, 6 (December 2024), 16 pages. https://doi.org/10.1145/3687946

1 INTRODUCTION

We focus on the efficient and scalable simulation of large-deformation contacting elastodynamic Finite Element (FE) models via subspace reduction. Reduced subspace models have long been applied to enable efficient computation of deformable body motion via low degree-of-freedom (DOF) kinematics. In the extreme, rigid and affine models reduce kinematics to a single transform per-body while, on the other end of the spectrum, a base FE model's nodes provide a DOF upper bound. The development of subspace models that find a happy balance between these two extremes remains a highly active area of investigation.

Recently developed subspace models targeting fast simulation and subspace construction are rapidly advancing in efficiency and expressiveness. But the fundamental questions of how much of a subspace to include and which subspace model to use remain, and must be answered anew for each application. Specifically, subspace simulations require expert knowledge and hand-tuning to carefully select a suitable model and subspace truncation that neither under-represents deformation nor unnecessarily increases compute cost with too many DOFs.

In part to address these questions, adaptive subspace construction methods have been proposed (see Section 2) to adaptively update subspace bases to better capture deformation as needed. However, these methods remain challenged by dynamics with high-speed transients, localized deformations (especially, given the often global support of reduced model modes) – such as those driven by contact and friction, and domains with material and geometric heterogeneities. Essentially, subspace adaptivity poses a "chicken and egg" problem: we generally cannot know what a reasonable subspace approximation is until we have a baseline full-space model solution to analyze and, by that point, we've already done the full-space computational work we wish to avoid in the first place. Or, put more simply, how can we measure the suitability of a specific subspace choice for a simulation step yet to be run?

In this paper, we propose an adaptive subspace model, oracle, and time-integration algorithm for large-deformation contacting elastodynamics. Our adaptive model is subspace agnostic: it can be used across subspace representations and allows unrestricted deformations, independent of subspace choice. Starting with a base, piecewise-linear volumetric FE model and user-selected subspace(s), we propose an in-time-step adaptivity oracle that measures the physical solution quality of a currently employed subspace basis, and evaluates the potential improvement offered by both available subspace updates *and* localized nodal enrichments.

We integrate our oracle into an adaptive subspace time-integrator that provides a pair of exposed tolerances that enables users to balance cost against accuracy. As these tolerances are tightened, we show that our model converges to a full-space solution. As this tolerance is relaxed, we obtain output-bound simulation costs primarily tied to the complexity of the simulated deformable body dynamics, rather than mesh resolution or simulated material type. In our evaluation, we show these properties hold across a wide range of challenging nonlinear materials models, material stiffnesses, heterogeneities, dynamic behaviors, and frictionally contacting conditions.

To support efficient and scalable simulation, we then build a nonlinear time-step solve algorithm customized for our model and oracle that supports fully parallel Newton-type time-step solves with a parallel iterative linear solver and efficient, accurate subspace integration via localized BFGS [Broyden 1970; Fletcher 1970; Goldfarb 1970; Shanno 1970] Hessian approximations.

For evaluation, we implement our adaptive subspace model, oracle and time-stepper with a high-fidelity IPC-based [Li et al. 2020] simulation model. We demonstrate our method's ability to preserve underlying model invariants (intersection and inversion-free trajectories), and to model challenging nonlinear elastodynamics with wide-ranging material variations and complex large-deformation dynamics, with a maximum speed-up of over an order-of-magnitude when producing simulations of equivalent visual plausibility to full-space IPC simulations.

2 RELATED WORK

Implicit time integrators [Baraff and Witkin 1998] are a fundamental tool in elastodynamic simulation with robust and stable output, even at large time steps. Many implicit integrators apply a variation of a Newton-type solver which requires solving sequences of linear systems constructed with Hessians of system energies [Hairer and Lubich 2014; Martin et al. 2011]. Implicit time-step solvers often distinguish themselves based on whether they utilize the exact-computed Hessian [Gast et al. 2015] or proxies thereof [Liu et al. 2017], and based on whether they solve the resulting linear system exactly (oftentimes via direct solvers) or iteratively using (for instance) Krylov subspace approaches [Smith et al. 2018], or multigrid methods [Liu et al. 2016]. Alternately, the implicit time-step solve itself can instead be resolved by other nonlinear methods including Gauss-Seidel iterations [Macklin et al. 2016; Müller et al. 2007], local-global methods [Bouaziz et al. 2014; Overby et al. 2017] and quasi-Newton strategies [Li et al. 2019; Liu et al. 2017]. These design choices, along with decisions on how to measure convergence and correspondingly how many inner and outer iterations to take, affect integrator performance, solution quality, and domain of applicability [Li et al. 2019]. While a full Newton-based method has the benefit of generality and convergence across wide material and resolution variations [Gast et al. 2015; Li et al. 2020], many methods often trade performance for stability and convergence in regimes that feature large stiffness ratios, stiff contacts and/or higher-resolution discretizations.

2.1 Subspace Simulation

Subspace simulation is one of the earliest strategies for accelerating elastodynamic simulation in graphics [Pentland and Williams 1989]. Subspace methods arise in two ways. The first constructs a reduced space that spans a low-energy set of deformations (either defined via elasticity or a data distribution) [Barbič and James 2005; Benchekroun et al. 2023; Chang et al. 2023; Sharp et al. 2023; Shen

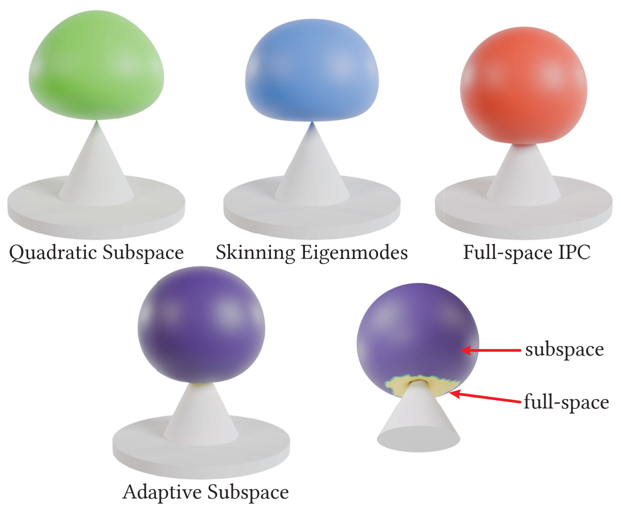

Fig. 2. Top left: available subspace models are generally well-suited for large global deformation but unable to resolve local deformations. Bottom: with our adaptive method a small amount of combined nodal and subspace enrichment closely captures the full-space solution's deformation (top right).

et al. 2021]. The second defines a small set of high-order DOF and associated interpolation functions [Faure et al. 2011; Jacobson et al. 2011; Wang et al. 2015]. Simulation performance then comes from operating on a smaller set of total variables due to a "small-enough" subspace size, but sacrifices fidelity as lower-dimensional subspaces tend to capture global modes and often just larger and smoother deformations. With active research focusing on new subspace models, we instead focus here on the complementary problem of adaptively utilizing available linear subspace models to their best advantage.

2.2 Adaptive Subspace Methods for Dynamics

The above-covered challenges to subspace simulation are especially critical when large deformations and localized contacts can not be expressed. To improve subspace simulation in these scenarios a number of adaptive methods for subspace integration have been developed. Kim and James [2009] focus on swapping time-step solves between a full-DOF solve and a subspace solve of dynamically updating orthogonalized snapshots. Here their oracle focuses on disagreements between the running basis and the simulated full-space model, with an emphasis on skipping as many full-DOF solves as possible. Alternately, other methods maintain a fixed subspace model and focus solely on its local enrichment with small regions of nodal DOF [Teng et al. 2015] or localized displacement fields [Harmon and Zorin 2013; Romero et al. 2022], at detected contacts [Harmon and Zorin 2013; Teng et al. 2015] or regions of anticipated deformation [Mercier-Aubin et al. 2022].

These methods obtain impressive speedups [Teng et al. 2015]. However, extending them to full contacting elastodynamics has remained challenging. Oracles either focus solely on deformation energies [Kim and James 2009], without simulating contact at all, or else apply refinement at all detected contact regions [Harmon and Zorin 2013; Teng et al. 2015], without analyzing internal energies (leading to overrefinement) nor handling critical-to-model self-contact. At the same time swapping between kinematic models per time step, as done in many of these methods, can generate popping and other discontinuity artifacts in simulated trajectories Kim and James [2009]. In this work we propose an oracle for combined subspace adaptivity and nodal enrichment that considers the coupled physics of contact and deformation forces, rather than geometric contact proximity. We combine this with a backing model that ensures consistent representation of full-space deformation with adapted subspace displacements, and so without popping, that can be computed efficiently online during the solve process, rather than in-between time steps.

2.3 Full-Space Adaptive methods

Instead of enriching subspaces or basis functions, an alternate and complementary strategy is adaptive meshing (see e.g., Manteaux et al. [2017] for a comprehensive review). While we do not perform explicit remeshing, related to this work is the question of how to pick a criteria for adaptivity. Prior works rely on the geometric discretization of the rest configuration [Bargteil et al. 2007], the deformed configuration [Dunyach et al. 2013], or stress/strain of a geometric primitive [Debunne et al. 2001; Ferguson et al. 2023; Narain et al. 2013, 2012; Simnett et al. 2009; Spillmann and Teschner 2008; Wicke et al. 2010]. These works, in particular contact-centric methods [Narain et al. 2013; Spillmann and Teschner 2008; Wicke et al. 2010] and especially in-time-step adaptivity [Ferguson et al. 2023], inspire our work. However, while these full-space methods focus on improving re-meshing, our adaptivity focuses on measuring and improving the expressivity of currently adopted subspaces for solving time steps on-the-fly.

2.4 IPC Simulation

Elastodynamics with large deformations, high stiffness ratios, and frictional contact are common and challenging scenarios for simulation. These features stress-test a solvers' robustness, convergence and speed. While our adaptive simulation framework is applicable to general FE models, we evaluate it using the recently developed Incremental Potential Contact (IPC) [Li et al. 2020] model. IPC allows us to challenge our oracle, model and solver to generate simulations with tightly coupled resolution of contact, friction and deformation. These are potentially expensive time-step solves with high-stiffness (from both barrier functions and material properties) and and strong nonlinearity that require many iterations to converge. This, with the added overhead of continuous collision detection (CCD) [Li et al. 2021] highlights the necessity of efficient use of subspace DOFs.

2.5 Fast Solvers For Contacting Elastodynamics

A wide range of methods target fast performance for elastodynamic simulation with contact (e.g., Bouaziz et al. [2014]; Macklin et al. [2016]; Müller et al. [2007]; Overby et al. [2017]). Motivated by IPC's guarantees and solution quality, many recent such methods develop algorithms specifically customized for accelerating IPC-type simulations [Chen et al. 2024; Huang et al. 2024; Lan et al. 2023, 2022; Li et al. 2023; Shen et al. 2024]. Similar in manner to accelerating

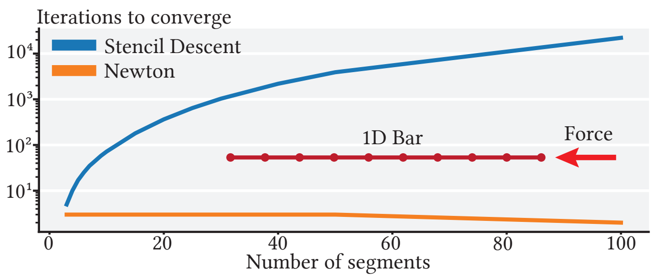

Fig. 3. Comparing the convergence behavior of Stencil Descent [Lan et al. 2023] and Newton's methods' on a simple example highlights the challenges with Stencil Descent's convergence under mesh refinement. We solve a single time step of a simple 1D bar with a nonlinear elastic material discretized using linear elements. As with many coordinate descent methods, Stencil Descent's convergence rate worsens (i.e., an increase in the number of iterations needed to converge) as the bar's resolution increases. For comparison, we plot a Newton solver's convergence for the same time step.

standard implicit time integration, these methods apply various combinations of quasi-Newton approximations and iterative-solver strategies. These methods demonstrate significant and impressive performance gains, especially when compared to the direct linear solvers and Newton methods used in standard IPC implementations.

Of course, trade-offs for these speed-ups are unavoidable. As well-covered in the literature, fast numerical solvers developed may often only apply to materials with a restricted upper bound of Young's modulus [Shen et al. 2024], induce strong resolution dependent convergence, may not model standard hyperelastic materials or friction [Lan et al. 2022], or discard IPC non-penetration guarantees [Chen et al. 2024]. As Young's modulus increases to several giga-pascal ranges (consistent with stiff real-world materials), and/or time-step sizes grow, convergence can likewise break down altogether. As a concrete example, in Figure 3 we highlight the strong mesh-dependent convergence degradation behavior of coordinate-descent-type solvers. Here we implement a simple 1D elastic bar simulation with Stencil Descent [Lan et al. 2023] as a representative method, and see that even for this simple domain, iteration counts required for comparable quality solutions blow up as we increase resolution.

Ideally, a simulator's performance should be proportional to the complexity of the dynamics rather than mesh resolution, time step or material properties. Unlike prior work on accelerating high-fidelity frictionally contacting elastodynamic simulations in specific regimes, our method focuses on producing output that is user-controllable, convergent, efficient, scalable, and general across a wide range of materials, time-step sizes, mesh resolutions and subspace models. Of course this generality comes with a tradeoff: we currently do not provide these additional, specialized solver speed-ups that take advantage of restricted simulation parameter ranges. However, when such restrictions are reasonable, future work could certainly look to combine our subspace adaptivity with these further accelerations.

3 METHOD

3.1 Background

We simulate large-deformation elastodynamics with frictional contact on simplicial meshes $\mathcal{T}$ (tetrahedra in 3D, triangles in 2D). Applying piecewise-linear discretization we store discrete fields for position and velocity in vectors $x, v \in \mathbb{R}^{dn}$ at the n vertices of the mesh in d-dimensional (d respectively 3 or 2) space. Each time step update solve can then be cast in optimization form as

$$x^{t+1} = \operatorname*{argmin}_x E(x), \tag{1}$$

with the incremental potential [Kane et al. 2000; Li et al. 2020],

$$\begin{aligned} E(x) &= K(x) + \alpha h^2\big(\Psi(x) + B(x) + D(x)\big), \\ K(x) &= \frac{1}{2}\|x - \tilde{x}^t\|_M^2, \end{aligned} \tag{2}$$

formed by the weighted sum of deformation (Ψ), contact barrier (B), and friction (D) potential energies, and an inertial weighting energy (K). Choice of predictor position, $\tilde{x}^t$ (an explicit function of prior position and velocity), scaling term $\alpha \in \mathbb{R}^+$, and explicit update equation for velocity from optimal solution x^{t+1}, then jointly define the specific choice of numerical time integration method. As a concrete example, consider implicit Euler with

$$\tilde{x}^t = x^t + hv^t, v^{t+1} = \frac{1}{h}\left(x^{t+1} - x^t\right), \text{ and } \alpha = 1. \tag{3}$$

Each piecewise-linear discrete energy in our incremental potential is then expressed as a weighted sum of energy functions over mesh element stencils, s (tetrahedral, triangle, edge, point or pairings thereof depending on energy type and dimension) in $\mathcal{T}$,

$$\sum_{s \in \mathcal{T}} w_s W_s(x), \tag{4}$$

with $w_s > 0$ the volume, area or length-weighted scaling of a stencil's rest-shape elements s, and W_s is the respective energy density function of each potential restricted to this element's stencil.

3.2 Adaptive Subspace Model

We begin our adaptive subspace model by considering subspaces with bases $U = (u_1, \cdots, u_r) \in \mathbb{R}^{dn \times r}$ and corresponding reduced coordinates $q \in \mathbb{R}^r$. We do not, however, make the usual assumption of a skinny subspace, with a "small enough" $r \ll dn$ a priori assumed sufficient to express full-space deformations. Instead, we adapt our subspace basis on-the-fly, inside each time-step solve, to construct locally well-suited subsets U_s of the larger full-span basis U.

We then break another standard subspace modeling convention: we *do not* use subspace DOF, q, to define our kinematics (e.g., via the linear map Uq). Instead, we always keep our full DOF model $x \in R^{dn}$ as our primary backing representation for each time-step t's final deformed position x^t. We then use our adapting subspace basis and additional nodal enrichment DOF as a running reduced-model scratch pad to efficiently compute displacements for each time-step solve from t to $t+1$. This provides lower-DOF computation within each time-step solve, continuity for our deformable body positions, and so ensures that each new update and downdate of our reduced model does not wipe out pre-existing deformations that the subspace currently cannot (and does not need to) express. In turn

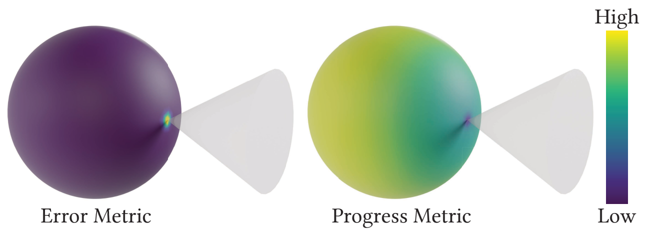

Fig. 4. Visualizing error and progress metrics: low magnitudes appear in dark purple, high magnitudes in bright yellow. Left: during an initial collision of soft ball with a fixed spike contact forces at the spike tip produce high measured error in our metric. Right: as visualized by our progress metric, localized forces are poorly represented in the current subspace, demonstrating the need for adaptive enrichment.

this allows us to choose local subspaces solely on their usefulness to improve our current time-step's solution.

We begin our simulations with a (small) initial active mode basis U_a (this can be as simple as just a single translational mode) and then apply updates and downdates to this model with both new candidate basis vectors from U, *and* via localized deformations enabled by adding and removing a subset, $\mathcal{X}_a \subseteq [1, n]$, of individual nodal enrichment DOF, $x_i \in \mathbb{R}^d, i \in \mathcal{X}_a$, to the reduced model. These latter local nodal enrichments complement the generally global support of subspace models. We can consider them as a special-case subspace with each DOF x_i matched to a corresponding "mode" given by the node i's selection matrix.

Given our currently active subspace basis $U_a \subset U$ and enriching nodal DOF $\mathcal{X}_a \subseteq [1, n]$, we evaluate a prospective mode's potential utility to improve our current solution estimate, x. To do so we balance the current solution estimate's error expressed in this candidate mode, against the currently active basis's ability to reduce this same error *without inclusion of the candidate mode*, as measured by a mode's corresponding progress metric.

In the following sections we first define our error and progress measures, demonstrate their application to both modal and nodal queries for updates of our running subspace model, and then construct our corresponding in-time-step adaptive subspace method and a custom solver algorithm for its efficient parallel computation of time-step solves.

3.3 Measuring Error

We start by first constructing our measure of model error for each individual node in the FE mesh. Within each time-step solve the IP's (Eq. (2)) gradient directly defines error with a measure of momentum imbalance per-node. However, the relative scaling of these gradients can widely vary spatially and by scene, with a strong dependence on materials (e.g., stiffnesses), mesh structure, scene dimensions, and temporal resolution.

Towards an error metric independent of these factors we construct a dimension-normalizing scaling factor for each node i,

$$\gamma_i = h^2 a_i \bar{W}_i. \tag{5}$$

that gives us a re-scaled, per-node gradient norm measure of *error*: $\|\nabla_{x_i} E(x)\|_2 / \gamma_i$. Here, $\bar{W}_i$ is the L2 norm of the elastic energy density Hessians computed at rest [1] ($\|\nabla^2 W_j(I)\|$) averaged across the elements node i contributes to, and a_i is the surface area (length in 2D) of the 1-ring around node i containing these elements. The first and second terms in γ_i above account for time-step dependence and mesh variation respectively, while the last term accounts for the (spatially varying) scaling of the system's elastic energies, which largely dominate the variations in scaling for large-deformation elastodynamics.

Following the analysis of Zhu et al. [2018] we obtain the scaling terms in γ_i by considering the gradient of our discrete elastic energy in Eq. (2) at each node i,

$$h^2 \nabla_{x_i} \Psi(x) = h^2 \sum_{j \in \mathcal{T}_i} w_j G_{ij}^T \nabla W_j(G_j x), \tag{6}$$

where $\mathcal{T}_i$ are node i's participating elements, G_j is the gradient operator for the j-th element (G_{ij} its restriction to node i), and W_j is again the elastic energy density function for each element j's material, and w_j are element volumes. We capture time-step scaling in the gradient directly with h^2 and account for material variation via the local average of participating W_j norms. We then choose scaling by a_j from the observation that that $\|G_{ij}\|_2 = \frac{a_{ij}}{2w_j}$, where a_{ij} is the area of the face (length of edge in 2D) opposed to node i in element j [Zhu et al. 2018].

With a dimensionless argument to W its gradients share the same dimensions, making the Hessian norm a suitable scaling. When combined with our scaling by area and time-step, our rescaled gradient is then unitless, providing a per-node error metric largely agnostic to time step, material, and mesh variations[2].

3.4 Measuring Progress

Regions with large measured error provide useful candidates for enriching our subspace model. However, we also want to be frugal with our enrichment – each new mode added incurs additional cost. Before adding any promising candidate modes, we want to check whether the currently active subspace model can resolve the error identified in its associated region. For example, in Figure 5, we see large portions of the deformed sliding block are highlighted with large gradient error; nevertheless, a single (already present) translational mode sliding the block downhill is sufficient to solve the time-step and no enrichment is required. Similarly, as stiffness increases, contact-induced deformations become increasingly global, correspondingly making a small number of subspace modes increasingly more effective, so less enrichment of regions with high-gradient necessary.

[1] **Rest Hessian for normalization:** we prefer a dimension-normalized error metric reflecting the magnitude of deformation. While the magnitude of the current Hessian is an appropriate candidate for normalization, it increases faster than the magnitude of the gradient during deformation, defying our purpose. Hence, we instead use the rest Hessian, which still works well for dimensional normalization but keeps the metric proportional to the magnitude of deformation.

[2] **Error metric sanity check in 3D:** the elastic energy density W is defined on a *unitless* deformation argument (refer to Zhu et al. [2018] for details), and thus its gradient and Hessian both have units of $\mathrm{ML^{-1}T^{-2}}$. Per Eq. (2), we have $\|\nabla_{x_i} E(x)\|_2 \sim h^2 \|\nabla_{x_i} \Psi(x)\|$, which (see Eq. (6)) has units of $\mathrm{T^2 \cdot L^3 \cdot L^{-1} \cdot ML^{-1}T^{-2} = ML}$. Similarly, the scaling factor γ_i has units of $\mathrm{T^2 \cdot L^2 \cdot ML^{-1}T^{-2} = ML}$, which cancels out numerator units $\|\nabla_{x_i} E(x)\|_2$ in the error metric, making the measure dimensionless.

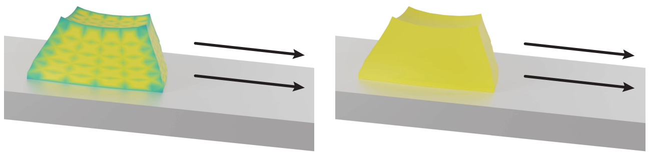

Fig. 5. A soft cube slides on a frictionless ramp. Elastic forces are balanced, so the resulting motion is pure rigid sliding. Left: large deformation results in high gradient error. Right: but the energy decrease, as measured by our progress metric, is also large since the rigid motion can be exactly represented in the current subspace.

We thus measure the *progress* for each node i, by the change in energy that would be generated nearby, if we were just to use the currently active subspace. Specifically, we first compute the line-search-scaled descent direction, d_a, generated by a Newton step using solely the currently active subspace, U_a. We then consider the stencil-size weighted sum of IP energies in the one-ring around each node i, $E_i(x)$, and compute the potential change generated by the subspace, per node, as

$$\Delta E_i = E_i(x) - E_i(x + d_a). \tag{7}$$

Following our above analysis of the IP gradient, we then consistently re-scale our per-node energy change with the scaling factor

$$\eta_i = h^2 v_i \bar{W}_i, \tag{8}$$

to get our dimensionless [3] per-node *progress* measure: $\Delta E_i/\eta_i$. Here $\bar{W}_i$ is the same averaged energy-density Hessian measure derived above, and v_i is a per-node volume (computed by apportioning element volume fractions equally to each node). For larger positive ΔE_i values, energy decrease is significant around node i, and so we use this measure as a local estimate for our active subspace's effectiveness to resolve errors in regions with large gradients.

3.5 Subspace and Node Candidates

With our above measures in place we can now define corresponding error and progress measures for nodal and modal update candidates.

Nodal Candidates. For an individual candidate node $x_i \in \mathbb{R}^3, i \in [1, n]$ we compute the gradient,

$$g^i = \sum_{s \in \mathcal{T}_i} w_s \nabla W_s(x) \in \mathbb{R}^{3n}, \tag{9}$$

restricted to the node's FE basis[4]. This amounts to a projection of the full gradient, $\nabla E(x)$, into the node's local basis, giving us a smoother measure of the error at the node compared to $\nabla_{x_i} E(x)$. We then define its local error with

$$\mathcal{G}_i(x) = \frac{\|g^i\|_2}{\gamma_i}, \tag{10}$$

[3]**Progress measure sanity check in 3D:** Following Eq. (2), the difference of $E(x)$ has the unit of $\mathrm{T}^2 \cdot \mathrm{L}^3 \cdot \mathrm{ML}^{-1}\mathrm{T}^{-2} = \mathrm{ML}^2$. The scaling factor η_i has the unit of $\mathrm{T}^2 \cdot \mathrm{L}^3 \cdot \mathrm{ML}^{-1}\mathrm{T}^{-2} = \mathrm{ML}^2$, which cancels the unit of the numerator and makes the progress measure dimensionless.
[4]Recall that $\mathcal{T}_i$ are node i's participating elements, w_s are element stencil weights (e.g., volume for tetrahedral elements), and W_s associated energy densities.

and progress by

$$\mathcal{E}_i(x) = \frac{\Delta E_i}{\eta_i}. \tag{11}$$

Modal Candidates. For an individual candidate mode $u_s \in \mathbb{R}^{dn}$ we compute its corresponding gradient and energy change by projection, respectively as

$$g^s = u_s \frac{u_s^T \nabla E(x)}{u_s^T u_s} \in \mathbb{R}^{3n}, \tag{12}$$

and

$$c^s = \bar{u}_i \frac{\bar{u}_s^T \Delta E}{\bar{u}_s^T \bar{u}_s} \in \mathbb{R}^n, \tag{13}$$

where $\Delta E = (\Delta E_1, \cdots, \Delta E_n) \in \mathbb{R}^n$, and the scalar mode reduction $\bar{u}_s \in \mathbb{R}^n$ is formed by the per-DOF average of u_s over each successive set of d coordinates. The reduction, $\bar{u}_s$, enables distribution of per-vertex scalar quantities (in our case energy change, ΔE_n) to per-vertex vector-valued modal bases, $u_s \in \mathbb{R}^{dn}$. Our choice, informed by equal basis values per-dimension in most bases (e.g. eigenmodes), is to average over the d coordinates to obtain the per-vertex scalar basis vector.

We then define each mode's error with a maximum (L_∞ measure),

$$\mathcal{G}_s(x) = \max_{j \in [1,n]} \frac{\|g_j^s\|_2}{\gamma_j}, \tag{14}$$

and progress by total (cumulative) energy change

$$\mathcal{E}_s(x) = \sum_{j \in [1,n]} \frac{c_j^s}{\eta_j}, \tag{15}$$

across the mode's (generally global) support in the mesh.

Modal Block Candidates. For some subspace models (e.g., skinning modes) it is natural to consider blocks of subspace modes $U_s = (U_{s,1}, \cdots, U_{s,p}) \in \mathbb{R}^{dn \times p}$ as a single unit for an individual error and progress query. To do so, our modal projections in Eq. (12) and Eq. (13) extend naturally as

$$g^s = U_s \Big(\frac{u_{s,1}}{{u_{s,1}}^T u_{s,1}}, \cdots, \frac{u_{s,p}}{{u_{s,p}}^T u_{s,p}} \Big)^T \nabla E(x) \in \mathbb{R}^{3n}, \tag{16}$$

and,

$$c^s = \bar{U}_s \Big(\frac{\bar{u}_{s,1}}{\bar{u}_{s,1}^T \bar{u}_{s,1}}, \cdots, \frac{\bar{u}_{s,p}}{\bar{u}_{s,p}^T \bar{u}_{s,p}} \Big)^T \Delta E \in \mathbb{R}^n, \tag{17}$$

respectively. Error and progress measures for the modes in U_s then follow unchanged with Eq.s (14) and (15).

3.6 In-Time-Step Adaptivity

We begin each time-step's Newton solve with our current state $x^t, v^t \in \mathbb{R}^{dn}$, our active subspace basis $U_a \in \mathbb{R}^{dn \times a}, a \leq r$, our subset $\mathcal{X}_a \subseteq [1, n], |\mathcal{X}_a| = m \leq n$, of enriching nodes, and a corresponding sparse selection matrix $S_a \in \mathbb{R}^{dn \times dm}$ that maps our enriching nodal DOF to their full-space entries.

To reduce our subspace model's kernel and to enable efficient time-step solves (see Section 3.8 and Figure 6 below) we choose a non-overlapping decomposition of our simulation domain between our nodal enrichment DOF and active modal DOF. To do so, in computing displacements, we often employ in the following a *masked* active modal basis $B_a = S'_a U_a$ where $S'_a \in R^{dn \times dn}$ masks out all our enriched DOF coordinates from the active modes' displacements to enforce the non-overlapping decomposition.

Within each time-step solve, at the start of each Newton iterate k, we first incrementally update U_a and $\mathcal{X}_a$ using our current error and progress measures, and then solve for new displacements $d = B_a d_q + S_a d_x \in \mathbb{R}^{dn}$ to update our improving solution $x^k \leftarrow x^{k-1} + d$. For modal updates, we visit modal block or individual candidates in parallel. Similarly, for nodal updates we visit all nodes in parallel. If a node is included, all nodes in its region within a fixed decomposition [5] are added. This approach reduces mesh-resolution dependency in nodal enrichment by adding uniformly sized groups of nodes (see Fig. 7). The fixed decomposition is also used for subspace Hessian updates, as discussed in Sec. 3.9. To summarize, we use two decompositions: the first is adaptive for separating nodal and modal domains, and changes within each time-step solve, the second is fixed and used exclusively for nodal enrichment and Hessian updates. At the end of the time-step solve we then use an a posteriori analysis based on our solution for downdating U_a and $\mathcal{X}_a$. We visit each of these steps in greater detail in the following sections below.

3.7 Adaptive Update Oracle

At start of each Newton iterate k we first compute an *unmasked* subspace descent direction for the active modal coordinates, q_a:

$$d_a = -H_q^{-1} \nabla_q E(x^{k-1} + U_a q_a),$$
$$H_a = \left(\nabla_q^2 E(x^{k-1} + U_a q_a)\right), \tag{18}$$

with a rescaling, $d_a \leftarrow \alpha d_a$, given by line-search on the initial d_a.

As covered above in Sections 3.4 and 3.5, we then apply d_a and x^{k-1} to compute current errors, $\mathcal{G}_s(x)$, and $\mathcal{G}_i(x)$, and subspace progressions, $\mathcal{E}_s(x)$ and $\mathcal{E}_i(x)$, for queried modes and nodal enrichments. Our adaptivity oracle then activates all candidate queries with both errors greater than threshold, $\mathcal{G}(\boldsymbol{x}) > \epsilon_{\mathcal{G}}$, and progress less than target, $\mathcal{E}(\boldsymbol{x}) < \epsilon_{\mathcal{E}}$. Thus, when local error is high, and the predicted improvement by the current active model is low for a queried mode or node, the displacement subspace model is enriched with the new DOF.

3.8 Newton-Iterate System

With our updated displacement model, $(U_a, \mathcal{X}_a)$, we next build the full system to compute our next descent direction. Expanding the corresponding IP energy evaluation, $E(x^{k-1} + B_a q_a + S_a x_a)$, w.r.t. variations in modal and enriched coordinates, we solve the linear system

$$\begin{pmatrix} H_q & J \\ J^T & H_x \end{pmatrix} \begin{pmatrix} d_q \\ d_x \end{pmatrix} = - \begin{pmatrix} g_q \\ g_x \end{pmatrix}. \tag{19}$$

[5] We precompute our subdomain decomposition with METIS [Karypis and Kumar 1997], which yields nodal partitions that are geometry-aware, uniformly shaped, and roughly equal volume. Example partitions are seen in Fig. 7, and the number of partitions for each scene are reported in Table 2.

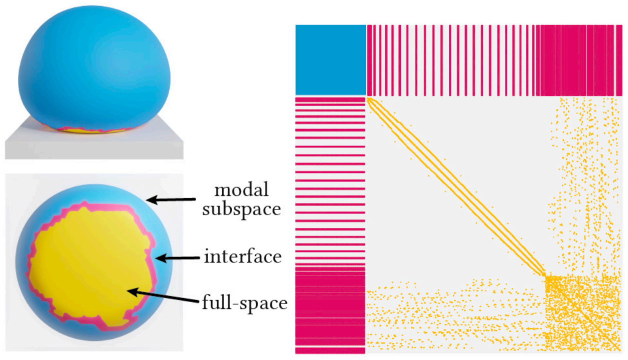

Fig. 6. Left: our *adaptive* subspace decomposition. Right: visualization of the corresponding matrix structure for our method's linear system solve.

for subspace and nodal enrichment descent directions d_q and d_x. Here $H_q \in \mathbb{R}^{a \times a}$ and $H_x \in \mathbb{R}^{dm \times dm}$ are our active modal and node Hessians respectively, $J \in \mathbb{R}^{a \times dm}$ is our coupling Jacobian capturing interaction between enriching nodes and modes, and $g_q = B_a^T \nabla E$, and $g_x = S_a^T \nabla E$ are the corresponding IP gradients for modal and nodal DOF.

Computation of the nodal Hessian $H_x = \nabla_{x_a}^2 E = S_a^T \nabla_x^2 E \, S_a$ sandwiches our full-space Hessian, $\nabla_x^2 E$, with our sparse selection matrix. However, in practice, as the selection stencils pull out just our m enriched DOF, we efficiently compute the sparse H_x matrix with just the corresponding enriched DOF stencil entries.

The coupling Jacobian $J = \frac{\partial^2 E}{\partial q_a \partial x_a} = B_a^T \nabla_x^2 E \, S_a$ then covers the interaction between enriched and modal DOF. Because of our design choice of a non-overlapping decomposition (see Section 3.6 above) for enriched nodes and modal subspaces, computing the Jacobian is also efficient with Hessian contributions to the sparse matrix J only sourced from energy stencils that include both enriched *and* subspace DOF (see Figure 6). This decomposition reduces J assembly cost, but also the resulting sparsity (often more than 10x fewer non-zeros than an overlapping decomposition), greatly reduces the cost of the matrix-vector products in the linear solve (Sec. 3.10).

On the other hand, because of the generally large subspace support, our subspace Hessian, $H_q = \nabla_{q_a}^2 E = B_a^T \nabla_x^2 E \, B_a$ is the most computational challenging term in each linear system. Unmasked subspaces, spanning an entire deformable body's domain, contribute large dense Hessian matrix entries to H_q for each energy stencil contribution, while evaluation of their entries could require computation of all full DOF internal energy Hessians in the domain. Direct computation of H_q is thus impractical. Nevertheless, we require accurate approximations of these entries for efficient and stable simulation solutions.

3.9 Subspace Hessian Construction

To efficiently compute our subspace Hessian, within each time-step solve we maintain a running BFGS [Broyden 1970; Fletcher 1970; Goldfarb 1970; Shanno 1970] approximation for the deformation energy (Ψ) contributions to our unmasked subspace Hessian $H_a = U_a^T \nabla_x^2 E U_a$. To have a more accurate approximation, and to be free

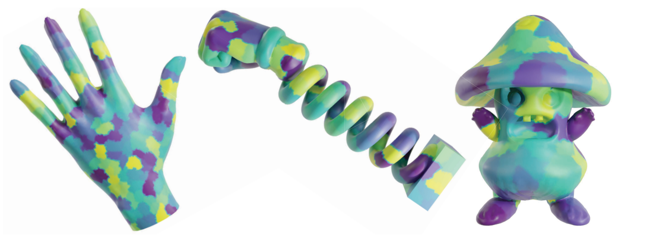
Fig. 7. We apply a precomputed METIS-constructed [Karypis and Kumar 1997] subdomain partition to both locally update our nodal DOFs in constructing our subspace Hessian (Sec. 3.9) *and* to form candidates groups for nodal enrichment queries (Sec. 3.5).

from parameter tuning, we do not use L-BFGS[Liu and Nocedal 1989] or other quasi-Newton strategies but rather direct BFGS-type secant updates to efficiently approximate the subspace integration of elastic energy Hessians [6]. We then adopt an exact evaluation of inertial (K) and contact (B and D) energy contributions to H_a. We choose this balance due to the following observations:

(1) elasticity costs dominate the computation of H_a,
(2) overhead for the constant inertial and sparse surface contact Hessians are small in comparison, and
(3) most critically, the large gradient variations introduced by contact energies significantly break down BFGS approximation accuracy over iterations.

We thus balance efficiency and accuracy by applying BFGS approximations for deformable energy stencil contributions to the subspace Hessian but exact contact and inertial energy Hessian evaluations.

While it is then tempting to simply and directly apply a BFGS approximation for all subspace elasticity contributions across our entire integration domain, there is a subtle but critical obstacle that requires care. Considering Eq. (19), notice that subspace energy contributions enter our solved linear system from both H_q and off-diagonal J entries. Using different approximations for these two blocks would significantly harm our approximation accuracy (and so significantly degrade convergence) and can break the positive definiteness of our system, blocking solver progress altogether.

To guarantee system consistency in Eq. (19), we instead *exactly* integrate elastic energy contributions to the subspace Hessians for all for all stencils that become coupled by including both subspace and enriched DOF. We do this because a BFGS approximation becomes inconsistent once even a single node within it is enriched, and we must regularly account for this as nodal updates can be applied in each new Newton iteration.

To allow for an efficient, consistent, and reusable BFGS approximation, we precompute an element-wise decomposition of our full simulation mesh $\mathcal{T}$ into L subspace-integration subdomains, reusing the fixed decomposition introduced in Sec. 3.6. For each subdomain $\ell \in [1, L]$, across iterations of each time-step solve, we maintain our running elastic energy BFGS approximation (re-initialized at the start of each time step). Then, to ensure system consistency, once any node in a subdomain ℓ is enriched and added to $\mathcal{X}_a$, we invalidate that subdomain and swap it over from BFGS approximation to exact evaluation. This allows us to apply exact evaluations only where and when needed and to continue our efficient BFGS approximations everywhere else in the domain away from nodal enrichments. These per-subdomain BFGS Hessians are guaranteed to remain positive semi-definite [Nocedal and Wright 2006], ensuring that the final assembled subspace Hessian is both consistent and positive definite. Finally, it's important to emphasize that this subdomain decomposition is fixed and soley for efficient subspace integration updates. It is maintained separately from our adaptively updating non-overlapping decomposition between subspace and nodal enrichment domains (Sec. 3.6).

We apply our BFGS update, at Newton iteration k, for each valid subdomain $\ell \in [1, L]$, to compute its subspace elastic Hessian contribution, H_a^ℓ, by solving the local subspace secant system,

$$H_a^\ell(q_a^k - q_a^{k-1}) = U_a^T S_\ell\big(\nabla\Psi(x^k) - \nabla\Psi(x^{k-1})\big), \tag{20}$$

where S_ℓ is a selection matrix that extract nodes in subdomain ℓ, and averages (evenly distributed) contributions from interface nodes shared by multiple subdomains. We then assemble our full elastic subspace Hessian by summing the contributions from all per-subdomain BFGS approximations and Hessians.

3.10 Adaptive Time-Step Solver

At the start of each new Newton iteration k, we compute our unmasked subspace Hessian H_a, use it to compute tentative subspace direction d_a, and apply it, as detailed above, to update our modal and nodal enrichments. If the subspace basis, U_a, is updated, we rebuild H_a from scratch, and reinitialize BFGS using the updated subspace Hessians. Then, to assemble our linear system in Eq. (19) we first directly mask H_a to compute our non-overlapping subspace Hessian, H_q. Then, taking advantage of our decomposed system structure, we construct a Schur-complement-based iterative method to solve the system.

Leveraging the subspace Hessian's small dense structure, we first (Cholesky) factorize H_q. With

$$d_q = -H_q^{-1}(Jd_x + g_q) \tag{21}$$

we obtain

$$(H_x - J^T H_q^{-1} J)d_x = H_q^{-1} g_q - g_x. \tag{22}$$

We solve this latter system with diagonal (d-blocked) preconditioned conjugate gradient (PCG) [7] via efficient matrix-vector products. Here $H_x d_x$ is inexpensive with sparse H_x applying a small fraction of the full-space Hessian's non-zeros. Likewise, we apply the sandwiched subspace inverse $J^T H_q^{-1} J d_x$ in three efficient substeps. First $y_1 = Jd_x \in \mathbb{R}^a$ is efficiently applied due to our choice of non-overlapping subspaces. Next $y_2 = H_q^{-1} y_1$ is extremely fast with our pre-factorization and subspaces with $a \ll n$. Then, $y_3 = J^T y_2$ follows as per y_1 with a final add of $-y_3$ to $H_x d_x$ to complete a matrix-multiply. Once we complete a CG solve we generate the new iterate's nodal descent direction d_x. Substituted into Eq. (21), we then get d_q, and so our corresponding full-space descent direction for iteration k: $d^k = B_a d_q + S_a d_x$.

[6] We apply BFGS as it significantly improves accuracy over L-BFGS. Since we are already working with a small, dense subspace Hessian that is factorized at a low cost, we do not need an L-BFGS approximate Hessian inverse.

[7] We use 10^{-5} relative tolerance.

We then perform backtracking line-search with α on $E(x^{k-1} + \alpha d^k)$ to get our new updated full-DOF state $x^k \leftarrow x^{k-1} + \alpha d^k$. Convergence and solution accuracy after each Newton iteration are measured in full space using a Newton-solve termination criteria with $\|d\|/h \leq \epsilon$ following Li et al. [2021]. After the solve, we update our active subspace basis and enriched nodal set with downdating (Sec. 3.11). We summarize our full algorithm below in Algorithm 1.

Algorithm 1: One step of our adaptive time-step solver

```
Function SimulationStep($x^t, U_a, \mathcal{X}_a$):
    $x \leftarrow x^t$;
    $q \leftarrow 0$;
    while not converged do
        $H_a \leftarrow$ SubspaceHessian$(x)$ ;                         // Sec. 3.9
        $U_a, \mathcal{X}_a, H_a \leftarrow$ UpdateBasis$(x, U_a, \mathcal{X}_a, H_a)$ ;  // Sec. 3.6
        // Compute descent direction (Sec. 3.10)
        $d_q, d_x \leftarrow$ LinearSolve$(x, U_a, \mathcal{X}_a, H_a)$;
        $\alpha \leftarrow$ LineSearch$(x, d_q, d_x)$;
        $x \leftarrow x + \alpha(B_a d_q + S_a d_x)$;
        $q \leftarrow q + \alpha d_q$;
    $U_a, \mathcal{X}_a \leftarrow$ DowndateBasis$(x, q, U_a, \mathcal{X}_a)$ ;           // Sec. 3.11
    return $x, U_a, \mathcal{X}_a$;
```

3.11 Downdating

We end each adaptive time-step solve with an optimal full-DOF solution $x^{t+1} \in \mathbb{R}^{dn}$, with a corresponding subspace displacement, $d_q^{t+1} \in \mathbb{R}^a$. We then apply these optimal end-of-step solutions for an a posteriori analysis to downdate the active subspace $U_a \in \mathbb{R}^{dn \times a}$ and the active nodal set $\mathcal{X}_a$.

First, for modal downdating, we consider the amount of utility, in the form of actual displacement, that each active subspace mode contributed during the just-completed time-step solve. To do so, for each active mode $u_i \in U_a$ with corresponding displacement $d_{q_i}^{t+1}$, we downdate mode i from the active subspace basis U_a if the full-space displacement applied by the mode was small, i.e., $\|u_i d_{q_i}^{t+1}\|_\infty/(lh) < \epsilon_d$. Here we scale by characteristic object length, l (given by bounding-box diagonal), and time-step, h, to correspondingly make our downdating criterion robust to scale and time-step size variation.

Correspondingly, for nodal downdating, we measure the utility (value added) of the nodal enrichment to the current subspace model. We do this by comparing the difference between the displacement that would have been effected at each node $i \in \mathcal{X}_a$ using the subspace displacement at end of step solve $d_{x_i}^a = S_i U_a d_q^{t+1} \in \mathbb{R}^d$ (where S_i is node i's selection matrix) against the actual displacement applied at the enriched nodal DOF, $d_{x_i} = x_i^{t+1} - x_i^t \in \mathbb{R}^d$. We downdate all enriched DOF i from $\mathcal{X}_a$ for which the currently active subspace model already well-captures their displacement, $\|d_{x_i} - d_{x_i}^a\|_2/(lh) < \epsilon_d$.

4 RESULTS

Our code is implemented in C++ with Eigen [Guennebaud et al. 2010] for linear algebra and parallelized with Intel TBB [Pheatt 2008] and OpenMP [Dagum and Menon 1998]. All timings are reported on a machine with a 32-core AMD Ryzen Threadripper PRO 3975WX CPU and 512 GB of RAM. For our Schur-complement's iterative solver phase, we use Eigen's Conjugate Gradient (CG) solver. The same CG solver is adopted when compared with full-DOF IPC using CG in its Newton solver. On the other hand, when comparing with full-DOF IPC using a direct solver, we use Intel Pardiso [Schenk et al. 2001] for LLT factorization[8].

For all examples, we use non-inverting, Neo-Hookean elasticity for our material model and BDF2 for our base time integration method [9]. We use a progress metric threshold of $\epsilon_{\mathcal{E}} = $ 1e-4 throughout this paper for our oracle. We evaluate most examples with default error metric threshold settings of $\epsilon_{\mathcal{G}} = 0.5$ for both nodal and modal adaptivity. In some examples, we vary these settings to explore the balance between the aggressiveness of each enrichment mode. Table 2 summarizes these and other parameters for all examples.

The full-time-step cost of each large scene can be found in Table 1. For each scene, we report the mesh statistics and the amount of nodal and modal enrichment for a representative time step with a significant iteration count and large number of contacts. For the same time step, we compare the cost of computing a solution with our method versus IPC using Pardiso LLT with both TBB and OpenMP parallelization – the fastest variant of IPC we test. We also compare with IPC using Eigen's CG solver with a 3×3 block-Jacobi preconditioner, against which we see speedups over of up to 70x (see Sec. 4.5). For all examples, we choose tolerance settings to achieve comparable high-fidelity outputs to IPC (please see our supplemental video) while observing significantly faster times. We observe speedups of 6-40x for timestep solves, depending on scene scale and amount of adaptivity. Unless expressly specified, all examples in this table and elsewhere (e.g., for comparing subspace models) use subspaces constructed from Skinning Eigenmodes [Benchekroun et al. 2023].

4.1 Oracle Evaluation

In Fig. 2, we highlight the inability of subspace bases (here Quadratic and Skinning Eigenmodes) to resolve localized deformations due to contact. However, with our (small) enriched nodal basis plus the same number of Skinning Eigenmodes (10 modes each), our method produces a simulation state qualitatively close to full-space IPC with a fraction of total DOFs.

Fig. 4 we visualize our error and progress metrics for a collision between a soft sphere a fixed cone. At the point of contact, our metrics measure high error and low energy progress, resulting in our adaptive model's downstream enrichment of nodal DOFs near this point of contact. In contrast, in Fig. 8, we see that full-space Newton decrement measure for the same time step, a residual that

[8]We also test with Cholmod Supernodal LLT [Chen et al. 2008], Eigen Simplicial LLT [Guennebaud et al. 2010], AND AMGCL [Demidov 2020]; we find that Pardiso is consistently fastest.

[9]For BDF2 (second-order backward differentiation) time-integration we use

$$\alpha = 4/9,\ \tilde{x}^t = \frac{1}{3}\left(4x^t - x^{t-1}\right) + \frac{2h}{9}\left(4v^t - v^{t-1}\right),$$

$$a^{t+1} = \frac{4h^2}{9}\left(x^{t+1} - \tilde{x}^t\right),\ \text{and}\ v^{t+1} = \frac{1}{3}\left(4v^t - v^{t-1}\right) + \frac{2h}{3}a^{t+1}.$$

in our IP energy (see Eq. (2)).

Table 1. We report statistics for a single time-step solve (choosing a step with large deformation and contact) for both our method and IPC. $|V|$, $|F|$, $|T|$ are the number of vertices, faces, and tetrahedra, respectively, in the scene. $|\mathcal{E}_a|/|V|$ is the ratio of enriched vertices to total vertices. $|U_a|/|U|$ is the ratio of enriched bases to the total number of available bases. These values are measured at the end of the time-step. Timings are reported in $minute : seconds$ format for our algorithm as well the time taken for a full-space IPC to solve the same time-step.

Scene	$\|V\|$	$\|F\|$	$\|T\|$	$\|\mathcal{E}_a\|/\|V\|$	$\|U_a\|/\|U\|$	Ours $(m:s)$	IPC $(m:s)$
Iceberg (Fig. 12)	18k	26k	64k	0.16	72/204	0:02	0:03
Roller Terrain (Fig. 11)	80k	72k	357k	0.049	72/240	0:18	7:29
Jello Soft (Fig. 14)	149k	39k	815k	0.77	120/216	0:49	4:19
Jello Rubber (Fig. 14)	149k	39k	815k	0.11	84/216	0:12	4:38
Jello Stiff (Fig. 14)	149k	39k	815k	0.0	84/216	0:07	33:17
Spiky Ball Eigenmodes (Fig. 18)	116k	207k	329k	0.27	84/204	2:21	3:49
Spring Fist (5e-3) (Fig. 15)	245k	67k	1314k	0.037	312/384	0:24	3:20
City (Fig. 13)	465k	634k	1719k	0.028	204/204	7:11	27:45
Mushroom Kingdom (Fig. 1)	550k	472k	2516k	0.051	264/576	14:36	153:36

Table 2. Scene parameters: E is Young's Modulus, ν is Poisson's Ratio, Δt is the time-step size, L is the number of subdomains (same for both nodal enrichment and BFGS), $\epsilon^s_{\mathcal{G}}$ and $\epsilon^e_{\mathcal{G}}$ are the error thresholds for the subspace DOF and nodal DOF, respectively. ϵ_d is the downdating threshold and $|U_{\text{init}}|$ is the size of the initial basis.

Scene	E (Pa)	ν	Δt	tol	L	$\epsilon^e_{\mathcal{G}}$	$\epsilon^s_{\mathcal{G}}$	ϵ_d	$\|U_{\text{init}}\|$
Iceberg (Fig. 12)	1e6	0.45	0.01	1e-3	200	0.1	0.1	0.1	60
Roller Terrain (Fig. 11)	1e5-1e7	0.4	0.01	5e-4	500	0.5	0.5	0.05	72
Jello Soft (Fig. 14)	1e5-1e9	0.4	0.005	5e-3	700	0.5	0.5	0.1	72
Jello Rubber (Fig. 14)	1e6-1e9	0.4	0.005	5e-3	700	0.5	0.5	0.1	72
Jello Rubber (Fig. 14)	1e9	0.4	0.005	5e-3	700	0.5	0.5	0.1	72
Spiky Ball Eigenmodes (Fig. 18)	5e5	0.4	0.01	1e-3	800	0.2	0.1	0.1	36
Spring Fist (5e-3) (Fig. 15)	1e5-1e8	0.4	0.005	1e-2	300	0.5	0.5	0.2	312
City (Fig. 13)	1e5-1e7	0.4	0.01	1e-4	1000	0.5	0.5	0.1	204
Mushroom Kingdom (Fig. 1)	1e5-1e9	0.4	0.005	1e-3	800	0.5	0.5	0.1	132

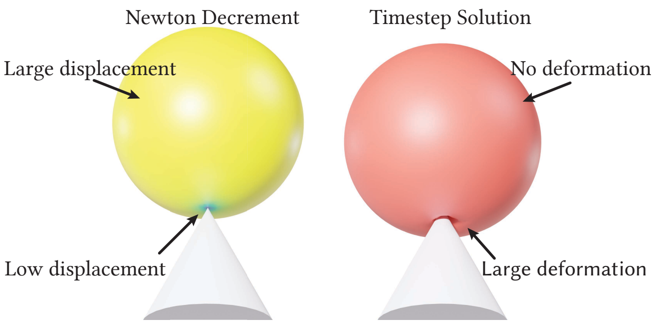

Fig. 8. The magnitude of an expensive full-space Newton decrement is not a useful measure for evaluating an individual vertex's contribution to the dynamics. For a sphere falling on spike example, deformation occurs near the point of contact, but points at the contact are *not* displaced (so the Newton decrement is small here); instead, the surrounding vertices translate vertically. Please compare with our adaptive method's inexpensive and effective oracle evaluation for the same example in Fig. 4.

we otherwise might presume to be a better-scaled error measure (albeit also prohibitively expensive), is not useful for determining the utility of a DOF's contribution to the dynamics.

Fig. 5 demonstrates the application of our progress metric. In this simulation, we slide a soft E=5e3 Pa cube down a frictionless

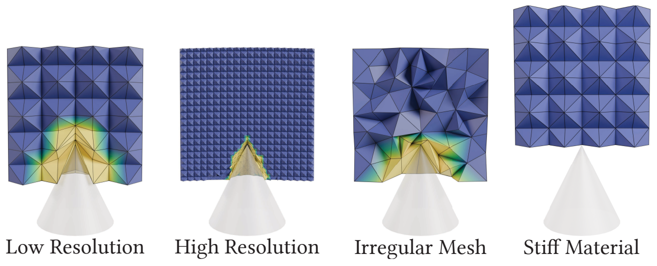

Fig. 9. Visualizing enrichment for a cube falling on a spike. In all simulations, we use the same parameters, demonstrating the oracle's robustness to configuration changes. In the first three simulations, we vary the mesh topology, but see similar enrichments. In the last simulation, we increase the stiffness (Young's Modulus 1e5 Pa to 1e8 Pa), and as expected, no enrichment occurs despite similar contact forces.

ramp. Once elastic forces balance and oscillations dampen, the cube remains compressed and only slides rigidly while deformed. In contrast to our example in Fig. 4, our error measure is significant due to the high elastic gradient. However, our progress measure is *also* correctly large, indicating that a simple modal basis that includes rigid motion, already applied, can capture the necessary error reduction via a rigid motion so that no additional unnecessary enrichment is applied.

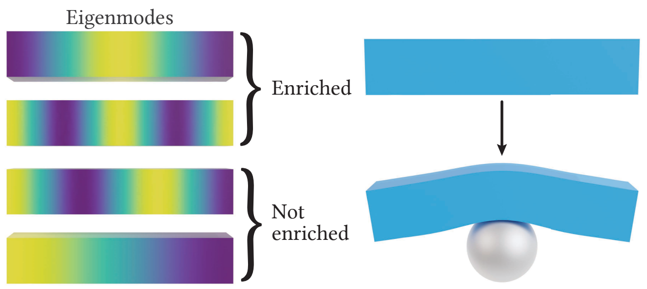

Fig. 10. To visualize modal enrichment, we drop a beam on a fixed sphere using a Skinning Eigenmodes subspace. Starting with just a constant affine basis, we query the addition of 4 additional Eigenmodes (left). The oracle enriches with necessary bending modes upon contact, resulting in the desired global bending deformation at the end of the time step (right).

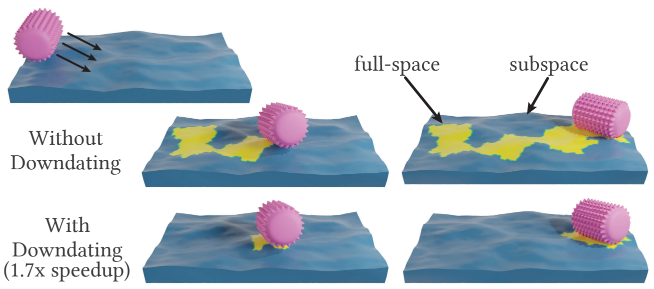

Fig. 11. Downdating reduces DOF, leading to significant speedups. We drop a spiky roller on a soft terrain and compare the simulation without (top) and with (bottom) downdating enabled. With downdating, we see fewer active DOF (in yellow) and a 1.7x speedup in the end-to-end simulation time with no loss in visual fidelity.

In Fig. 9 we drop a cube on a fixed spike to evaluate the mesh-independence and material-awareness of our adaptivity oracle. Here, we run the same scene up to the same time step with four variations: a low-resolution mesh, a high-resolution mesh, a nonuniform mesh, and a stiff material. In the first three examples (left to right), all scene parameters are held the same (including material stiffness) so we expect the same effective compliance and see that our oracle's applied adaptivity helps achieve this. Likewise, we see that in all three examples, the enrichment and, as a result, the total deformation is qualitatively close, demonstrating the oracle's consistency across variations in mesh resolution and topology. In the fourth (rightmost) example, we highlight the oracle's ability to account for material variation. Here, the cube's stiffness is increased from 1e5 Pa to 1e8 Pa, so that no deformation is expected. In response, we see that our oracle correctly does not enrich (with nodal nor subspace DOF), and the dynamics remain fully and correctly governed by the current simple subspace basis.

Next, in Fig. 10, we demonstrate the effectiveness of our modal adaptation. Here, we drop a beam on a fixed sphere with a Skinning Eigenmodes subspace. Our starting subspace basis includes just the constant affine basis. At each Newton iteration, we query the addition of 4 additional subspace Eigenmodes for modal adaptivity. In this scenario, the expected behavior is to bend after contact with the sphere. During the beam's initial contact with the sphere, the oracle appropriately enriches with additional necessary bending modes, generating the desired global bending deformation at the end of the time step.

4.2 Downdating

In Fig. 11, we evaluate the effectiveness of our downdating. In this example, we drop a stiffer spiky roller on a soft terrain. Keeping parameters fixed, we simulate this scene with (bottom) and without (top) our downdating enabled. Our posterior analysis ensures that when downdating is enabled we only remove DOFs that contribute little utility to the current displacement. In turn, we see that our simulation with downdating produces a simulation with qualitatively identical dynamics at a fraction of enriched DOFs. This results in a 1.7x speedup in the end-to-end simulation over our method without downdating applied.

4.3 Error Analysis

We expose our two oracle error (nodal and modal) thresholds to allow users to vary output fidelity against cost. With these thresholds, users can change solution quality in their simulations, ranging all the way from very little enrichment to full-space solutions of the underlying full-DOF IP in Eq. (2).

In Fig. 12, we demonstrate this control. We simulate a soft toy boat dropped on a fixed iceberg geometry and, varying our error thresholds, reporting our method's corresponding generated enrichment (nodal and modal) and the full-space residuals of the solutions averaged over the five most challenging (and expensive) time steps mid-drop.

We hold our subspace error threshold fixed (at $\epsilon^s_{\mathcal{G}} = 0.5$) for the two left plots and vary our nodal error threshold. We report the ratio of enriched nodes at each threshold and the full-space residual. The full-space residual is measured using the full IPC Newton Decrement norm (∞-norm), giving us the exact error of each of our adaptive subspace simulations w.r.t. their accuracy in the full-space IPC simulation solution. We observe that at small thresholds (i.e., $\epsilon^e_{\mathcal{G}} = 1\text{e-}2$) all or almost all nodes are enriched, giving a residual that tightly satisfies the IPC tolerance. As we increase our threshold, the ratio of enriched nodes decreases, and the residual monotonically increases, giving simulation results that span the entire spectrum from accurate IPC solutions to fast approximations.

In the next two plots, we perform the same threshold analysis on our modal subspace, varying our subspace adaptivity error threshold while holding the nodal error threshold fixed at $\epsilon^e_{\mathcal{G}} = 0.5$. As in the nodal experiment above, as we decrease the threshold, our number of added modes increases, and our subspace solution's full-DOF residual decreases. However, as expected, with the limited ability of modes to fully resolve local deformations, solely varying subspace thresholds has less impact on residuals. In summary, we see that decreasing the error threshold improves accuracy and expressiveness while increasing the DOF count for both nodal and modal components.

In Fig. 13, we drop a spiky mace on a large squishy city scene. Taking a simulation snapshot from the middle of our adaptive subspace simulation, we solve this same corresponding middle step

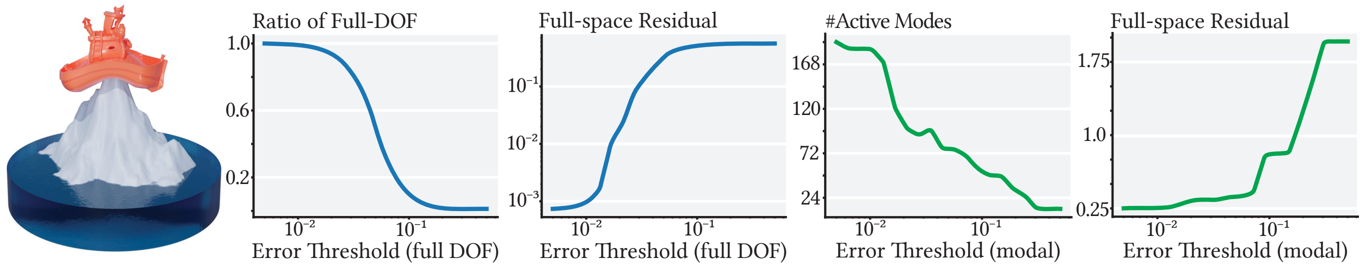

Fig. 12. We drop a soft boat on an iceberg and perform a threshold analysis for the full-space error metric (left two plots) and the subspace error metric threshold (right two plots). As the error threshold decreases for full DOF enrichment, our solver's enrichment increases and monotonically reduces the residual (measured using the full-space IPC residual) all the way to the full IPC solution. Similarly, for the modal error, reducing the threshold improves the residual and increases the number of active modes.

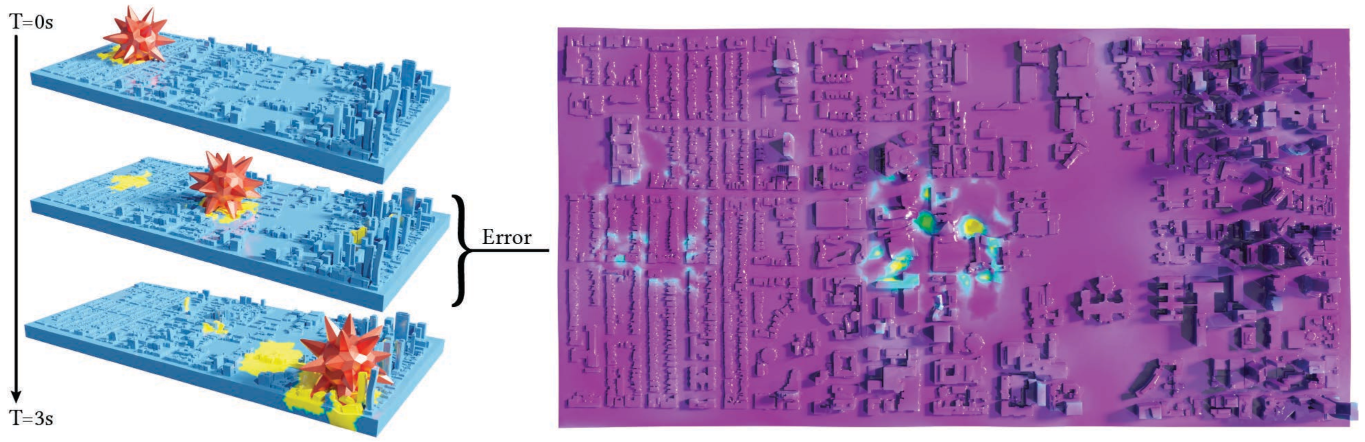

Fig. 13. (Left) Three sequential frames from the City simulated with adaptive subspaces. (Right) We compare the difference (displacement distance) between our adaptive subspace solution and the full-space IPC solution at the middle time step. The highest error is concentrated at the deepest-compressed contact points, where the largest deformation occurs. Small displacement differences are also measured (right) but not visible in comparisons at some of the interfaces between modal and full DOF spaces, and likewise elsewhere in the scene due to the modal deformations imperfectly resolving the full-space deformations below the currently applied thresholds.

with full-space IPC and measure the magnitude of the differences between our method and the IPC solution. As expected, the largest solution difference between the two is at the deepest-compressed points of contact. Additionally, we also measure much finer, not visible, displacement differences at the interface between the subspace and full-space representations (on the far left of the domain). Interspersed throughout the scene are also other small magnitudes differences due to the modal deformations imperfectly resolving the full-space deformation solution. Of course, as covered above, decreasing tolerance thresholds then further decreases these differences with increasing cost.

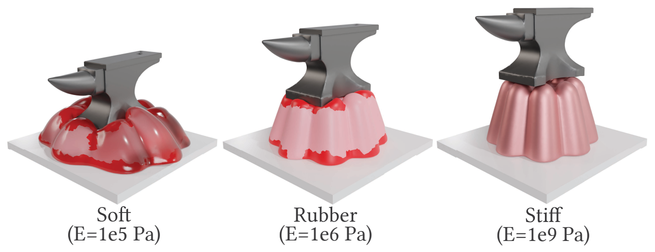

Fig. 14. Our oracle performs well across a broad spectrum of material stiffnesses. We drop a stiff anvil (E = 1e9 Pa) onto a "jello" block with varying stiffness (1e5 Pa, 1e6 Pa, and 1e9 Pa) and observe that the amount of nodal enrichment (shown in dark red) directly varies with the stiffness and so on the amount of localized deformation.

4.4 Varying Materials and Time Steps

Subspace adaptivity enables efficient simulation across a wide range of variations in scene parameters. In Fig. 14, we drop a stiff (E=1e9) anvil on a "jello" block that we progressively stiffen across simulations with Young's moduli ranging from 1e5 to 1e9 Pa. For each simulation we show the frame at effective maximum compression. Light red indicates no enrichment, and dark red indicates enriched nodes.

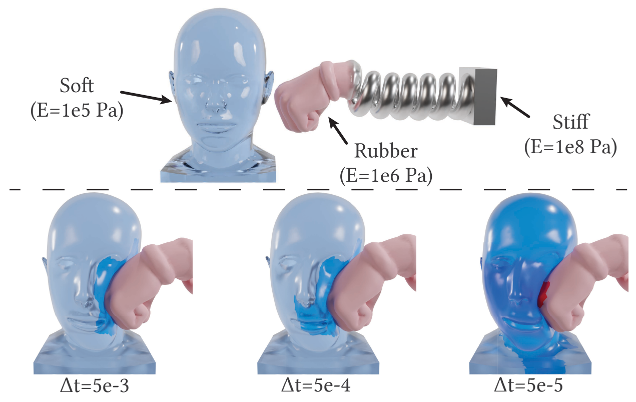

Fig. 15. Our model can incorporate heterogeneous materials with Skinning Eigenmodes. We initialize the spring-fist in a compressed state and release it to punch a gelatin head. Simulations at different timesteps show that as the timesteps decrease, elastic shocks are better resolved, resulting in more nodal-DOF enrichment (in dark blue and dark red).

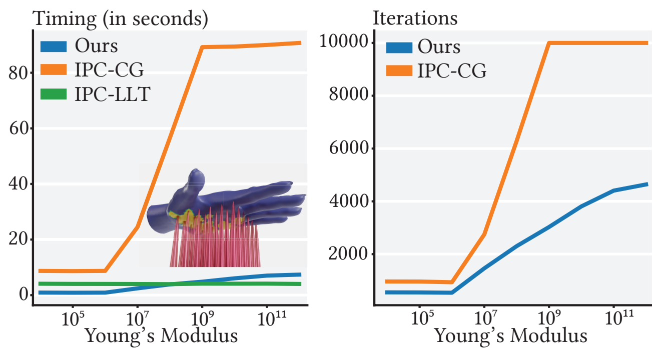

Fig. 16. We drop a rubber hand on a bed of spikes, vary the spike stiffness, and measure resulting linear solve costs. As the stiffness increases, the runtime of IPC-LLT is effectively fixed, but IPC with a Conjugate Gradient iterative solver with block-jacobi preconditioning explodes in cost. We cap the iterative solves at 10,000 iterations and see that IPC-CG reaches this at stiffnesses of 1e8 Pa and above. Our solver also sees an iteration count increase, but is far less sensitive to stiffness variations than IPC-CG.

In the softest case where deformation is expected to be largest, the oracle correctly heavily enriches (77% of full DOF are active) near the points of contact and large deformation. The adapted Skinning Eigenmodes subspace resolves the more global bulging around the perimeter with some nodal corrections. As we increase the stiffness, we see less compression and, correspondingly, less enrichment. In the stiffest setting the oracle then requires no enrichment at all, with dynamics entirely expressed by the subspace.

In Fig. 15, we perform a similar experiment but now both vary time step size and material spatially. Here, smaller time-steps are expected to capture higher-frequency dynamics. In this example, we have a soft gelatin (E=1e5 Pa) head and a heterogeneous spring-fist model with a rubber boxing glove (E=1e6 Pa) and a stiff (E=1e8 Pa) spring.

We initialize the simulation by compressing the spring-fist model using the subspace dynamics with heterogeneous Skinning Eigenmodes. Starting in the compressed state, once the simulation begins, the fist springs towards the head, punching it at high speed.

We simulate this scene with three time-step sizes decreasing in order of magnitude: h = 5e-3s, 5e-4s, 5e-5s. As the time-step decreases, our adaptive subspace simulation increases nodal enrichment and so better resolves the elastic shockwave that propagates across the head after collision. This corresponds with the expected improved resolution of high-frequency deformation dynamics as time-integration accuracy improves with decreasing time step size. Please see our supplemental video.

In most algorithms, scene parameter variations have substantial performance implications. For example, large material stiffnesses significantly impact linear system conditioning, leading to poor convergence of iterative linear solvers, e.g., CG methods. We observe that our oracle and custom iterative solver remain robust to wide variations in parameters and, when coupled with our reduced DOF representation, we see a heavily reduced performance impact on the linear solver as we increase material stiffnesses.

In Fig. 16, we perform a similar experiment to our above "jello-drop" example. Here we drop a rubber (E=1e6 Pa) hand on spikes and take a time-step snapshot in contact. Using this frame for a consistent initialization, we vary the Young's moduli of the spikes from 1e4 to 1e12 Pa. For each stiffness, we report the linear solver time and iteration count for our adaptive subspace linear solver, as well as for corresponding solves for two full-space IPC methods, the first using Pardiso (IPC-LLT) and the second applying CG with a 3×3-block-Jacobi preconditioner (IPC-CG). As expected, the IPC-LLT solver's performance is agnostic to material variations. On the other hand, the IPC-CG solver's performance (and iteration count) quickly explodes in cost as we increase material stiffness: we cap the PCG solver at 10,000 iterations and, after E=1e8 Pa, we observe the PCG solve always saturates to max iterations. Our adaptive subspace solver also increases in cost and iterations with stiffness, but the resulting impact on performance is far "softer", converging in practical counts across simulations. Notably, we find that there is a crossing point at which LLT solves become faster than our solver. However, even here, costs remain of a similar order of magnitude between our iterative solver and LLT.

In Fig. 1 we show frames from our largest "Mushroom Kingdom" simulation. This scene includes material heterogeneties with a stiff castle (E=1e9 Pa), soft terrain with mushrooms and houses (E=1e5 Pa), a soft and evil mushroom guy (E=1e6 Pa), and a rubber stomping shoe (E=1e7 Pa). Here the shoe is initially dropped on the evil mushroom, squashing it and the nearby terrain with large deformation contact (note the boot pattern captured by our oracle's nodal enrichment). As the boot and mushroom guy tumble across the domain, our oracle progressiveley updates and downdates both nodes and modes to track local deformations of the kingdom's homes, mushrooms and landscape. Comparing IPC's cost to ours on an expensive time step, we see that our method is significantly faster, with over a 10x speedup over IPC with Pardiso LLT (see Table 1).

Table 3. We report the linear solver costs of the scenes shown, comparing our Schur complement solver against IPC with Pardiso LLT, and IPC-CG with 3×3-block-jacobi preconditioning. Linear systems are collected from the beginning of a representative time-step (large deformation and contact). We report the number nonzero in our full coupled system, the number of nonzeros for the full-space IPC problem, as well the number of iterations. Both iterative solvers are solved to a relative residual of 1e-5.

Scene	Ours (s)	IPC-LLT (s)	IPC-CG (s)	Ours' nnz	IPC nnz	Ours' iters	IPC-CG iters
Iceberg	0.04	0.25	2.10	7.6e5	1.8e6	59	450
Roller Terrain	0.08	1.71	5.02	8.5e5	9.3e6	268	674
Jello (Soft)	0.53	10.3	7.00	2.5e7	1.9e7	84	782
Jello (Rubber)	0.10	10.2	7.63	4.8e6	1.9e7	68	846
Jello (Stiff)	5e-5	10.4	9.28	7.1e4	1.9e7	0	1035
Spiky Ball (Eigenmodes)	0.91	1.57	3.00	6.4e6	1.1e7	315	348
Spring Fist (5e-3)	0.03	12.4	23.0	2.9e6	3.1e7	50	2232
City	1.30	9.64	14.2	1.5e7	4.9e7	356	689
Mushroom Kingdom	0.52	21.4	36.5	5.5e6	6.4e7	307	1598

4.5 Linear Solver

Our adaptive-subspace-customized iterative solver leverages our model's linear system structure (see Fig. 6). With a small dense subspace block and its correspondingly cheap factorization, we efficiently solve the Schur complement form of the system, leading to an iterative solve of just the enriched nodal DOF. In general the nonzeros (nnz) of this system is dramatically smaller than the full-space Hessian, leading to much faster solutions, as reported in Table 3. We report linear solver costs for our adaptive subspace iterative solver, IPC with Pardiso LLT (IPC-LLT), and IPC with a Conjugate Gradient solver with a 3×3-block-Jacobi preconditioner (IPC-CG) for a single solve at the beginning of a representative time step in each scene (the same costly timesteps from Table 1). Our linear solver is consistently an order of magnitude faster than IPC-LLT, and even faster than IPC-CG, which was consistently the slowest. Additionally, we report the number of nnz of our coupled system and the full-space IPC system, showing that our systems typically have an order of magnitude fewer nonzeros than the IPC system.

4.6 Subspace Suitability

So far we have focused on our method's ability to jointly adapt modes for global deformation and nodal DOF for additional enrichment to make up the difference. As covered in Sec. 2 not all subspaces are created equal. As the provided subspace becomes less suitable for an application and/or fewer modes are made available for adaptivity, the same error tolerance will provide less utility and more nodal refinement and tighter tolerances will be required to make up the difference for a less expressive basis (see Sec. 5 below).

In Fig. 17 we compare three common subspace models: Skinning Eigenmodes [Benchekroun et al. 2023], Biharmonic Coordinates [Weber et al. 2012], and standard Eigenmodes. We apply our beam drop on a sphere experiment (same scene settings for all) with these three different subspaces. For skinning eigenmodes and biharmonic coordinates we use a modal subspace 192 DOF and a smaller corresponding subspace of 48 DOF standard eigenmodes.

Here we see the amount of enrichment depends heavily on the quality and size of the subspace. Skinning Eigenmodes and Biharmonic Coordinates both have the same DOF count (and so can

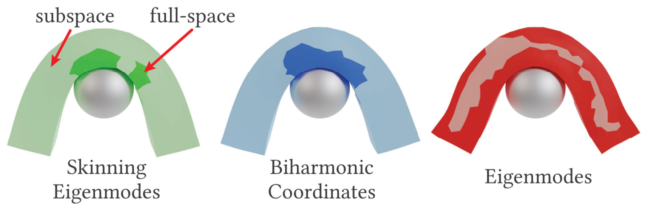

Fig. 17. The quality of the chosen subspace greatly influences the amount of full-space enrichment needed. Using the same oracle tolerances, we simulate a beam dropping on a sphere with Skinning Eigenmodes, Biharmonic Coordinates, and standard Eigenmodes – the first two with 192 DOF and the latter with 48 DOF. Darker colors denote enriched full DOF, and lighter areas are subspace only. Skinning Eigenmodes and Biharmonic Coordinates require relatively little nodal enrichment, whereas standard Eigenmodes, which are not rotationally invariant, require heavy nodal enrichment.

satisfactorily resolve global bending deformation) but we see that skinning modes generally do a better job with overall least amount of enrichment, whereas Biharmonic coordinates introduce a greater amount of enrichment near the point of contact. In contrast, Eigenmodes are not rotation invariant and, with fewer DOF, do not resolve dynamics satisfactorily, leading to heavy full-space enrichment and less reasonable overall deformation for the same tolerances.

5 LIMITATIONS AND FUTURE WORK

As covered in the last section, likely the biggest limitation of our method is a remaining reliance on starting with a sufficiently expressive subspace basis from which to adapt from. As our experiments in Figures 17 and 18 demonstrate, poor subspace model choices can hinder simulation quality while better choices of subspace bases likewise enable both better subspace and nodal refinement. In the extreme (see Figure 18 top) insufficient subspace expressivity can prevent high error norms and so require tighter error tolerances for refinement. This can be ameliorated by lowering our error tolerances or (as we demonstrate in Figure 18 middle) by providing improved and/or larger subspace bases (recall our method automatically updates and downdates the active basis, so there is little cost for providing large initial candidate bases or even multiple subspaces). This intriguingly opens the door to the possibility of

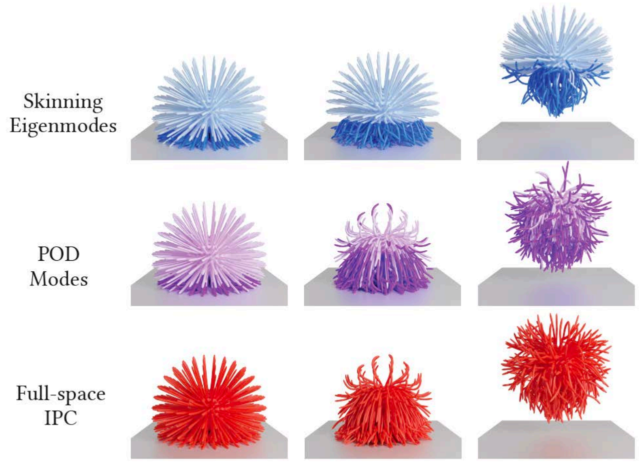

Fig. 18. Adaptive subspaces require sufficiently expressive subspace bases from which to adapt. With unsuitable candidate subspaces, adaptivity can be challenging. Here we simulate the drop of a soft spiky "Koosh" ball with a Skinning Eigenmodes subspace (16 modes) and an empirical Proper Orthogonal Decomposition (POD) [Sirovich and Kirby 1987] basis with 11 modes. Here the Skinning Eigenmode basis is completely unsuited for the small tendril deformation and so is unable to resolve the global deformation of the tentacles. On the other hand the small POD basis built using snapshots from a corresponding full-DOF simulation more closely matches the full simulation, but still requires significant enrichment to capture details.

combining our subspace adaptivity with online updates of our subspace model basis U [Kim and James 2009]. This also speaks to a second related limitation of our method (also demonstrated in Figure 17) that, like all adaptive schemes, our exposed tolerances must be changed with the quality of the basis provided. However, no matter what tolerances are chosen, we retain the guarantees of the underlying simulation model (in the case of IPC – stability, non-inversion and non-interpenetration).

For future work, we intend to explore applying our approach to libraries of disparate subspaces, rather than linear spaces derived from a single procedure. We could also employ continuous neural subspaces [Fulton et al. 2019; Modi et al. 2024] or fracture modes [Sellán et al. 2022], which have shown the ability to handle topology change. In addition, extending our method to shells and rods should be a promising direction that could broaden its utility, while its application for solving static volumetric equilibria should be immediate. Further developments may also focus on refining our subspace construction methods to enhance memory efficiency, which is crucial for managing complex scenarios containing billions of elements. Finally, given our fully parallel pipeline, exploring our model's performance on GPU architectures (e.g., via GIPC [Huang et al. 2024]) is a promising direction for advancing our method's performance.

6 CONCLUSION

We have presented an adaptive subspace time integration method composed of a general-purpose adaptivity oracle for guiding subspace refinement and a corresponding adaptive subspace model and a custom-built fast solver, that operates in the augmented subspace. Our method differs significantly from prior subspace approaches: it applies adapted subspaces solely for efficient computation of displacements during the time-step solution procedure, storing all final states in maximal coordinates, while its oracle works online, during time integration, to predict areas of required refinement regardless of whether they arise from contact or other sources of deformation. Our custom linear solver, built around these pieces, is 6 to 40 times faster than MKL Pardiso and up to 70 times faster than Eigen's Jacobi preconditioned conjugate gradient solver. This leads to speedups of over an order-of-magnitude when producing simulations of equivalent visual plausibility.

REFERENCES

David Baraff and Andrew Witkin. 1998. Large Steps in Cloth Simulation. In *SIGGRAPH'98 Proceedings*. 43–54.

Jernej Barbič and Doug L. James. 2005. Real-Time Subspace Integration for St. Venant-Kirchhoff Deformable Models. *ACM Trans. Graph.* 24, 3 (jul 2005), 982–990. https://doi.org/10.1145/1073204.1073300

Adam W Bargteil, Chris Wojtan, Jessica K Hodgins, and Greg Turk. 2007. A finite element method for animating large viscoplastic flow. *ACM transactions on graphics (TOG)* 26, 3 (2007), 16–es.

Otman Benchekroun, Jiayi Eris Zhang, Siddhartha Chaudhuri, Eitan Grinspun, Yi Zhou, and Alec Jacobson. 2023. Fast Complementary Dynamics via Skinning Eigenmodes. *ACM Transactions on Graphics (TOG)* 42, 4 (2023), 1–21.

S. Bouaziz, S. Martin, T. Liu, L. Kavan, and M. Pauly. 2014. Projective Dynamics: Fusing Constraint Projections for Fast Simulation. *ACM Trans Graph* 33, 4 (2014).

Charles George Broyden. 1970. The convergence of a class of double-rank minimization algorithms 1. general considerations. *IMA Journal of Applied Mathematics* 6, 1 (1970), 76–90.

Yue Chang, Peter Yichen Chen, Zhecheng Wang, Maurizio M Chiaramonte, Kevin Carlberg, and Eitan Grinspun. 2023. LiCROM: Linear-Subspace Continuous Reduced Order Modeling with Neural Fields. In *SIGGRAPH Asia 2023 Conference Papers*. 1–12.

Anka He Chen, Ziheng Liu, Yang Yin, and Cem Yuksel. 2024. Vertex Block Descent. *ACM Transactions on Graphics (TOG)* 116 (2024).

Yanqing Chen, Timothy A Davis, William W Hager, and Sivasankaran Rajamanickam. 2008. Algorithm 887: CHOLMOD, supernodal sparse Cholesky factorization and update/downdate. *ACM Transactions on Mathematical Software (TOMS)* 35, 3 (2008), 1–14.

Leonardo Dagum and Ramesh Menon. 1998. OpenMP: an industry standard API for shared-memory programming. *IEEE computational science and engineering* 5, 1 (1998), 46–55.

Gilles Debunne, Mathieu Desbrun, Marie-Paule Cani, and Alan H Barr. 2001. Dynamic real-time deformations using space & time adaptive sampling. In *Proceedings of the 28th annual conference on Computer graphics and interactive techniques*. 31–36.

Denis Demidov. 2020. AMGCL—A C++ library for efficient solution of large sparse linear systems. *Software Impacts* 6 (2020), 100037.

Marion Dunyach, David Vanderhaeghe, Loïc Barthe, and Mario Botsch. 2013. Adaptive remeshing for real-time mesh deformation. In *Eurographics 2013*. The Eurographics Association.

François Faure, Benjamin Gilles, Guillaume Bousquet, and Dinesh K. Pai. 2011. Sparse Meshless Models of Complex Deformable Solids. *ACM Trans. Graph.* 30, 4, Article 73 (July 2011), 10 pages.

Zachary Ferguson, Teseo Schneider, Danny Kaufman, and Daniele Panozzo. 2023. In-Timestep Remeshing for Contacting Elastodynamics. *ACM Transactions on Graphics (TOG)* 42, 4 (2023), 1–15.

Roger Fletcher. 1970. A new approach to variable metric algorithms. *The computer journal* 13, 3 (1970), 317–322.

Lawson Fulton, Vismay Modi, David Duvenaud, David I. W. Levin, and Alec Jacobson. 2019. Latent-space Dynamics for Reduced Deformable Simulation. *Computer Graphics Forum* 38, 2 (2019), 379–391. https://doi.org/10.1111/cgf.13645 arXiv:https://onlinelibrary.wiley.com/doi/pdf/10.1111/cgf.13645

T. Gast, C. Schroeder, A. Stomakhin, C. Jiang, and J. M Teran. 2015. Optimization integrator for large time steps. *IEEE Trans Vis Comp Graph* 21, 10 (2015).

Donald Goldfarb. 1970. A family of variable-metric methods derived by variational means. *Mathematics of computation* 24, 109 (1970), 23–26.

Gaël Guennebaud et al. 2010. Eigen.

Ernst Hairer and Christian Lubich. 2014. Energy-diminishing integration of gradient systems. *IMA J. Numer. Anal.* 34, 2 (2014), 452–461.

David Harmon and Denis Zorin. 2013. Subspace Integration with Local Deformations. *ACM Trans. Graph.* 32, 4, Article 107 (jul 2013), 10 pages. https://doi.org/10.1145/

2461912.2461922

Kemeng Huang, Floyd M. Chitalu, Huancheng Lin, and Taku Komura. 2024. GIPC: Fast and Stable Gauss-Newton Optimization of IPC Barrier Energy. *ACM Trans. Graph.* 43, 2, Article 23 (mar 2024), 18 pages. https://doi.org/10.1145/3643028

Alec Jacobson, Ilya Baran, Jovan Popović, and Olga Sorkine. 2011. Bounded Biharmonic Weights for Real-Time Deformation. *ACM Trans. Graph.* 30, 4, Article 78 (jul 2011), 8 pages. https://doi.org/10.1145/2010324.1964973

Couro Kane, Jerrold E Marsden, Michael Ortiz, and Matthew West. 2000. Variational integrators and the Newmark algorithm for conservative and dissipative mechanical systems. *International Journal for numerical methods in engineering* 49, 10 (2000), 1295–1325.

George Karypis and Vipin Kumar. 1997. METIS: A software package for partitioning unstructured graphs, partitioning meshes, and computing fill-reducing orderings of sparse matrices. (1997).

Theodore Kim and Doug L. James. 2009. Skipping Steps in Deformable Simulation with Online Model Reduction. *ACM Trans. Graph.* 28, 5 (dec 2009), 1–9. https://doi.org/10.1145/1618452.1618469

Lei Lan, Minchen Li, Chenfanfu Jiang, Huamin Wang, and Yin Yang. 2023. Second-Order Stencil Descent for Interior-Point Hyperelasticity. *ACM Trans. Graph.* 42, 4, Article 108 (jul 2023), 16 pages. https://doi.org/10.1145/3592104

Lei Lan, Guanqun Ma, Yin Yang, Changxi Zheng, Minchen Li, and Chenfanfu Jiang. 2022. Penetration-Free Projective Dynamics on the GPU. *ACM Trans. Graph.* 41, 4, Article 69 (jul 2022), 16 pages. https://doi.org/10.1145/3528223.3530069

Minchen Li, Zachary Ferguson, Teseo Schneider, Timothy Langlois, Denis Zorin, Daniele Panozzo, Chenfanfu Jiang, and Danny M. Kaufman. 2020. Incremental Potential Contact: Intersection- and Inversion-free Large Deformation Dynamics. *ACM Trans. Graph. (SIGGRAPH)* 39, 4, Article 49 (2020).

Minchen Li, Ming Gao, Timothy Langlois, Chenfanfu Jiang, and Danny M. Kaufman. 2019. Decomposed Optimization Time Integrator for Large-Step Elastodynamics. *ACM Trans. on Graph.* 38, 4 (2019).

Minchen Li, Danny M. Kaufman, and Chenfanfu Jiang. 2021. Codimensional Incremental Potential Contact. *ACM Trans. Graph. (SIGGRAPH)* 40, 4, Article 170 (2021).

Xuan Li, Yu Fang, Lei Lan, Huamin Wang, Yin Yang, Minchen Li, and Chenfanfu Jiang. 2023. Subspace-Preconditioned GPU Projective Dynamics with Contact for Cloth Simulation. In *SIGGRAPH Asia 2023 Conference Papers.* 1–12.

Dong C Liu and Jorge Nocedal. 1989. On the limited memory BFGS method for large scale optimization. *Mathematical programming* 45, 1 (1989), 503–528.

H. Liu, N. Mitchell, M. Aanjaneya, and E. Sifakis. 2016. A scalable schur-complement fluids solver for heterogeneous compute platforms. *ACM Trans Graph* 35, 6 (2016).

T. Liu, S. Bouaziz, and L. Kavan. 2017. Quasi-Newton Methods for Real-Time Simulation of Hyperelastic Materials. *ACM Trans Graph* 36, 4 (2017).

Miles Macklin, Matthias Müller, and Nuttapong Chentanez. 2016. XPBD: Position-Based Simulation of Compliant Constrained Dynamics *(MIG '16)*. Association for Computing Machinery, New York, NY, USA, 49–54. https://doi.org/10.1145/2994258.2994272

P-L Manteaux, Christopher Wojtan, Rahul Narain, Stéphane Redon, François Faure, and M-P Cani. 2017. Adaptive physically based models in computer graphics. In *Computer Graphics Forum*, Vol. 36. Wiley Online Library, 312–337.

Sebastian Martin, Bernhard Thomaszewski, Eitan Grinspun, and Markus Gross. 2011. Example-Based Elastic Materials. *SIGGRAPH Comput. Graph.* (2011), 1–8.

Alexandre Mercier-Aubin, Paul G. Kry, Alexandre Winter, and David I. W. Levin. 2022. Adaptive Rigidification of Elastic Solids. *ACM Trans. Graph.* 41, 4, Article 71 (jul 2022), 11 pages. https://doi.org/10.1145/3528223.3530124

Vismay Modi, Nicholas Sharp, Or Perel, Shinjiro Sueda, and David I. W. Levin. 2024. Simplicits: Mesh-Free, Geometry-Agnostic Elastic Simulation. *ACM Trans. Graph.* 43, 4, Article 117 (jul 2024), 11 pages. https://doi.org/10.1145/3658184

Matthias Müller, Bruno Heidelberger, Marcus Hennix, and John Ratcliff. 2007. Position based dynamics. *J Vis Commun Image R* 18, 2 (2007), 109–118.

Rahul Narain, Tobias Pfaff, and James F O'Brien. 2013. Folding and crumpling adaptive sheets. *ACM Transactions on Graphics (TOG)* 32, 4 (2013), 1–8.

Rahul Narain, Armin Samii, and James F O'brien. 2012. Adaptive anisotropic remeshing for cloth simulation. *ACM transactions on graphics (TOG)* 31, 6 (2012), 1–10.

Jorge Nocedal and Stephen Wright. 2006. *Numerical optimization.* Springer Science & Business Media.

M. Overby, G. E Brown, J. Li, and R. Narain. 2017. ADMM $\supseteq$ Projective Dynamics: Fast Simulation of Hyperelastic Models with Dynamic Constraints. *IEEE Trans Vis Comp Graph* 23, 10 (2017).

A. Pentland and J. Williams. 1989. Good Vibrations: Modal Dynamics for Graphics and Animation. *SIGGRAPH Comput. Graph.* 23, 3 (jul 1989), 207–214. https://doi.org/10.1145/74334.74355

Chuck Pheatt. 2008. Intel® threading building blocks. *Journal of Computing Sciences in Colleges* 23, 4 (2008), 298–298.

Cristian Romero, Dan Casas, Maurizio M. Chiaramonte, and Miguel A. Otaduy. 2022. Contact-Centric Deformation Learning. *ACM Trans. Graph.* 41, 4, Article 70 (jul 2022), 11 pages. https://doi.org/10.1145/3528223.3530182

Olaf Schenk, Klaus Gärtner, Wolfgang Fichtner, and Andreas Stricker. 2001. PARDISO: a high-performance serial and parallel sparse linear solver in semiconductor device simulation. *Future Generation Computer Systems* 18, 1 (2001), 69–78.

Silvia Sellán, Jack Luong, Leticia Mattos Da Silva, Aravind Ramakrishnan, Yuchuan Yang, and Alec Jacobson. 2022. Breaking Good: Fracture Modes for Realtime Destruction. arXiv:2111.05249 [cs.GR] https://arxiv.org/abs/2111.05249

David F Shanno. 1970. Conditioning of quasi-Newton methods for function minimization. *Mathematics of computation* 24, 111 (1970), 647–656.

Nicholas Sharp, Cristian Romero, Alec Jacobson, Etienne Vouga, Paul G Kry, David IW Levin, and Justin Solomon. 2023. Data-Free Learning of Reduced-Order Kinematics. (2023).

Siyuan Shen, Yin Yang, Tianjia Shao, He Wang, Chenfanfu Jiang, Lei Lan, and Kun Zhou. 2021. High-Order Differentiable Autoencoder for Nonlinear Model Reduction. *ACM Trans. Graph.* 40, 4, Article 68 (jul 2021), 15 pages. https://doi.org/10.1145/3450626.3459754

Xing Shen, Runyuan Cai, Mengxiao Bi, and Tangjie Lv. 2024. Preconditioned Nonlinear Conjugate Gradient Method for Real-time Interior-point Hyperelasticity. *arXiv preprint arXiv:2405.08001* (2024).

Timothy JR Simnett, Stephen D Laycock, and Andy M Day. 2009. An Edge-based Approach to Adaptively Refining a Mesh for Cloth Deformation.. In *TPCG.* 77–84.

Lawrence Sirovich and Michael Kirby. 1987. Low-dimensional procedure for the characterization of human faces. *Josa a* 4, 3 (1987), 519–524.

Breannan Smith, Fernando De Goes, and Theodore Kim. 2018. Stable neo-hookean flesh simulation. *ACM Transactions on Graphics (TOG)* 37, 2 (2018), 1–15.

Jonas Spillmann and Matthias Teschner. 2008. An adaptive contact model for the robust simulation of knots. In *Computer Graphics Forum*, Vol. 27. Wiley Online Library, 497–506.

Yun Teng, Mark Meyer, Tony DeRose, and Theodore Kim. 2015. Subspace Condensation: Full Space Adaptivity for Subspace Deformations. *ACM Trans. Graph.* 34, 4, Article 76 (jul 2015), 9 pages. https://doi.org/10.1145/2766904

Yu Wang, Alec Jacobson, Jernej Barbič, and Ladislav Kavan. 2015. Linear Subspace Design for Real-Time Shape Deformation. *ACM Trans. Graph.* 34, 4, Article 57 (jul 2015), 11 pages. https://doi.org/10.1145/2766952

Ofir Weber, Roi Poranne, and Craig Gotsman. 2012. Biharmonic coordinates. In *Computer Graphics Forum*, Vol. 31. Wiley Online Library, 2409–2422.

Martin Wicke, Daniel Ritchie, Bryan M Klingner, Sebastian Burke, Jonathan R Shewchuk, and James F O'Brien. 2010. Dynamic local remeshing for elastoplastic simulation. *ACM Transactions on graphics (TOG)* 29, 4 (2010), 1–11.

Yufeng Zhu, Robert Bridson, and Danny M Kaufman. 2018. Blended cured quasi-newton for distortion optimization. *ACM Transactions on Graphics (TOG)* 37, 4 (2018), 1–14.