Published in final edited form as:
ACM Trans Graph. 2009 December 1; 27(5): 165. doi:10.1145/1409060.1409118.

Optimizing Cubature for Efficient Integration of Subspace Deformations

Steven S. An, **Theodore Kim**, and **Doug L. James**
Cornell University

Abstract

We propose an efficient scheme for evaluating nonlinear subspace forces (and Jacobians) associated with subspace deformations. The core problem we address is efficient integration of the subspace force density over the 3D spatial domain. Similar to Gaussian quadrature schemes that efficiently integrate functions that lie in particular polynomial subspaces, we propose cubature schemes (multi-dimensional quadrature) optimized for efficient integration of force densities associated with particular subspace deformations, particular materials, and particular geometric domains. We support generic subspace deformation kinematics, and nonlinear hyperelastic materials. For an r-dimensional deformation subspace with $O(r)$ cubature points, our method is able to evaluate subspace forces at $O(r^2)$ cost. We also describe composite cubature rules for runtime error estimation. Results are provided for various subspace deformation models, several hyperelastic materials (St.Venant-Kirchhoff, Mooney-Rivlin, Arruda-Boyce), and multimodal (graphics, haptics, sound) applications. We show dramatically better efficiency than traditional Monte Carlo integration.

CR Categories: I.6.8 [Simulation and Modeling]: Types of Simulation—Animation, I.3.5 [Computer Graphics]: Computational Geometry and Object Modeling—Physically based modeling G.1.4 [Mathematics of Computing]: Numerical Analysis—Quadrature and Numerical Differentiation

Keywords

Dimensional model reduction; reduced-order modeling; subspace integration; quadrature; subspace dynamics; dynamic deformations; nonlinear solid mechanics; real-time simulation

1 Introduction

Recently, dimensional model reduction has gained attention due to the difficulty of simulating detailed physical models at high rates for multimodal (graphics, haptics, sound) applications. Such reduced-order methods construct a small, r-dimensional subspace that captures the salient features of a much larger, N-dimensional model. If $r \ll N$, simulating these reduced-order models entirely in the r-dimensional subspace holds the promise of superior runtime performance provided costs independent of N are achieved.

Unfortunately, efficient evaluation of internal forces for subspace deformation models has proven difficult for arbitrary geometry and nonlinear materials. In particular, the inability of many models to support more complex materials, such as biological materials for surgical simulation, is unfortunate. Current force evaluation methods either have costs dependent on N [Krysl et al. 2001] or that scale poorly ($O(r^4)$) or are essentially restricted to particular materials (St. Venant-Kirchhoff) in practice [Barbič and James 2005], or are inaccurate.

In this paper, we present a general-purpose force evaluation method that applies to more general materials, subspace kinematics, and geometry, while delivering fast *N*-independent force evaluations at $O(r^2)$ cost (without exploiting sparsity). We achieve this scalability by performing a cubature optimization preprocess that enables fast runtime evaluation. Additionally, we provide evidence that our cubature schemes are computationally accurate and efficient, are resistant to over-fitting, and provide clear improvements over traditional Monte Carlo integration [Baraff and Witkin 1992].

Subspace Internal Forces

Our method can be applied to general subspace deformation models, but for concreteness of exposition we will focus on the case of dimensional model reduction for detailed finite element meshes [Krysl et al. 2001; Barbič and James 2005]. In a full FEM simulation with *N* degrees of freedom, the displacement vector would be of length *N*. However, in dimensional model reduction, the equations of motion have been projected into a linear, *r*-dimensional subspace of deformations. The reduced-order equations of motion describing a mesh deforming in subspace coordinates can be written as

$$\mathbf{M}\,\ddot{\mathbf{q}}+\mathbf{f}(\mathbf{q})=\mathbf{f}_{\mathbf{ext}}, \tag{1}$$

where, $\mathbf{M}\in\mathbb{R}^{\mathbf{r}\times\mathbf{r}}$ is the (often constant) mass matrix, $\mathbf{q}\in\mathbb{R}^r$ is the generalized displacement vector of reduced coordinates, $\mathbf{f}(\mathbf{q})\in\mathbb{R}^{\mathbf{r}}$ is the subspace internal restoring force, $\mathbf{f}_{\mathbf{ext}}\in\mathbb{R}^{\mathbf{r}}$ are external forces, and the overdot denotes differentiation.

Unfortunately, the subspace internal force term $\mathbf{f}(\mathbf{q})$ in (1) is responsible for the poor $O(rN)$ and $O(r^4)$ scalings of previous methods [Krysl et al. 2001;Barbič and James 2005], so its efficient evaluation is the focus of this paper. The subspace force can be formulated in terms of a potential energy function, $E(\mathbf{q}):\mathbb{R}^r\rightarrow\mathbb{R}$, given by the domain integral,

$$E(\mathbf{q})=\int_{\Omega}\Psi(X;\mathbf{q})\,d\Omega_X, \tag{2}$$

where $\boldsymbol{\psi}(X;\mathbf{q})$ is the nonnegative strain energy density at material point *X* of the undeformed material domain $\boldsymbol{\Omega}$ [Bonet and Wood 2008]. The subspace internal force is then the gradient of this energy, and is given by the vector integral

$$\mathbf{f}(\mathbf{q})=-\nabla_{\mathbf{q}}E(\mathbf{q})=-\int_{\Omega}\nabla_{\mathbf{q}}\Psi(X;\mathbf{q})\;d\Omega_X=\int_{\Omega}\mathbf{g}(X;\mathbf{q})\;d\Omega_X, \tag{3}$$

where we denote the "reduced-force density" integrand by

$$\mathbf{g}=\mathbf{g}(X;\mathbf{q})=-\nabla_{\mathbf{q}}\Psi(X;\mathbf{q})\in\mathbb{R}^r. \tag{4}$$

Our approach is to approximate $\mathbf{f}(\mathbf{q})$ using an *n*-point cubature (multi-dimensional quadrature) scheme,

$$\mathbf{f}(\mathbf{q})=\int_{\Omega}\mathbf{g}(X;\mathbf{q})\;d\Omega_X\approx\sum_{i=1}^{n}w_i\,\mathbf{g}(X_i;\mathbf{q}). \tag{5}$$

We precompute estimates of the *n* positive cubature weights (w_i), and *n* cubature points (X_i) by minimizing $\mathbf{f}(\mathbf{q})$ integration error over a training set of ($\mathbf{q}$, $\mathbf{f}(\mathbf{q})$) pairs. Our proposed

preprocess is a greedy algorithm that incrementally selects cubature sample points, and estimates their nonnegative cubature weights. In our discrete implementation, each cubature point X_i corresponds to a linear tetrahedral element, since the force density $\boldsymbol{\psi}$ is constant over each element. Runtime evaluation of subspace forces consists of evaluating only n deformed tetrahedra, and accummulating their $\mathbf{f}(\mathbf{q})$ contribution. An overview of our preprocess and runtime pipeline is shown in Figure 1.

Although no formal theory exists for cubature over nontrivial 3D domains, our empirical evidence indicates that cubature schemes can be optimized for efficient subspace force evaluation for (1) particular geometric domains, (2) particular materials, (3) particular deformation subspace kinematics and/or motion examples, and (4) greatly accelerated subspace force evaluation. See figure 2 for a preview of our results.

2 Other Related Work

For more than two decades, following the pioneering work of Terzopoulos, Barr, Witkin, and others, the mathematical foundations of Lagrangian dynamics have been employed in computer graphics to build dynamic physically based models of parametrized deformable shapes [Terzopoulos et al. 1987; Terzopoulos and Witkin 1988; Witkin and Welch 1990]. Monte Carlo methods were widely used to evaluate subspace force integrals (3); for example, Baraff and Witkin [1992] mention that the gradient of the potential energy integral could be easily computed for relatively simple examples (such as a quadratically deforming block) using Monte Carlo integration: “For second-order polynomial deformations, a small number of sample points (on the order of fifty) yields adequate results.” Unfortunately, we observe (Figure 8) that Monte Carlo is inefficient for more complex geometry, deformations, and materials.

Our approach is inspired by Gaussian quadrature and related schemes from classical 1-D numerical integration [Hildebrand 1956; Press et al. 1992], e.g., an n-point Gaussian quadrature scheme for a proper integral is

$$\int_{-1}^{1} g(X)\, dX = \sum_{i=1}^{n} w_i\, g(X_i), \tag{6}$$

where w_i are n (positive) weights, and X_i are n abscissae chosen as roots of a suitable orthogonal polynomial. Surprisingly, with only n quadrature samples, Gaussian quadrature can evaluate integrals of polynomials of degree $2n-1$ *exactly*, so that each function sample effectively kills off a polynomial subspace of dimension two. If the function is very well approximated by a degree $2n-1$ polynomial, then the integral is also very well approximated. While Gaussian quadrature and related quadrature schemes (Gauss-Radau, Gauss-Lobatto, Radau, etc.) are widely used, variants for integrating higher-dimensional functions are restricted to tensor product domains and other simple parameterizations [Hildebrand 1956; Press et al. 1992]. In finite element analysis, integrals such as (3) are computed using related quadrature schemes but only for simple element shapes and low-order basis functions [Bathe 1996]. Exotically shaped elements have essentially avoided evaluation of (3) by using corotated linear models [Kaufmann et al. 2008]. Sadly, no generalizations of Gaussian quadrature exist to nontrivial multidimensional domains (where it is called cubature), or to nonpolynomial function spaces relevant to subspace deformation forces.

Dimensional model reduction techniques use Galerkin projection onto a relatively low-dimensional linear subspace (spanned by the columns of the dense basis matrix, $\mathbf{U}$) to obtain a smaller reduced set of equations of motion, the unreduced internal forces, $\mathbf{F} = \mathbf{F}(\mathbf{Uq}) \in \mathbb{R}^N$ (evaluated in shape $\mathbf{Uq}$) are projected to yield reduced forces, $\mathbf{f} = \mathbf{U}^T\mathbf{F}(\mathbf{Uq}) \in \mathbb{R}^r$, equivalent to (3). Linear eigenmode coordinates are often used for subspace dynamics to resolve weak

material nonlinearities in small-strain configurations [Bathe 1996] or mode-mode coupling [Vakakis 2001]. For more nonlinear problems, Krysl et al. [2001] formalized subspace integration for finite element models by employing a posteriori dimensional model reduction with "empirical eigenvectors" (or proper orthogonal decomposition (POD); principle component analysis) subspaces. The principle benefits are fewer ODEs to integrate, and smaller linear systems to solve (during implicit Newmark integration and Newton iterations). Unfortunately the speedup is fundamentally limited by reduced force (and Jacobian) evaluation since they are based on "brute force" evaluation of $O(N)$ unreduced nodal force values ($\mathbf{F}$) in order to evaluate $\mathbf{f}$ via subspace projections ($\mathbf{U}^T\mathbf{F}$)–an $O(rN)$ cost. Related issues arise when computing forces and gradients for general-purpose multi-scale basis formulations, e.g., the basis-refinement formulation of CHARMS [Grinspun et al. 2002] (see also [Capell et al. 2002]) formalizes the (multi-resolution) scatter/gather integration steps of Galerkin subspace projection, but again invokes fine-scale evaluation of reduced force (and Jacobian) components for accurate evaluation of nonlinear force response. Other schemes rely on coarsened discrete approximations for speed [Debunne et al. 2001], albeit at the cost of geometric and/or material resolution.

Recently Barbič and James [2005] observed that for the special case of St.Venant-Kirchhoff materials (large deformations, but linear stress-strain response), the reduced internal force (3) was in fact a vector of cubic polynomials in the reduced coordinates, $\mathbf{q} \in \mathbb{R}^r$. Subspace integrators can thus be generated for large-deformation reduced StVK models using arbitrary linear subspace bases, such as from PCA of training data, or linear and derivative modes using mass-PCA [Barbič and James 2005]. While extremely fast for small r, and suitable for real-time haptics [Barbič and James 2007], the cost complexity of the reduced force evaluation scales as $O(r^4)$ so that only models smaller than, e.g., $r = 30$, offer significant speedups [Barbič 2007]. More general reduced kinematics would also be useful, but the model is limited to the linear basis superposition typical of POD methods. In contrast, our proposed approximation allows higher rank models due to its $O(r^2)$ force calculations for $O(r)$ cubature points.

For linear quasistatics, condensation and other precomputations can enable output-sensitive (subspace) evaluation of contact force/displacement responses [Cotin et al. 1999; James and Pai 1999]. For linear elastodynamics, linear modal analysis allows efficient subspace force and dynamics models to be precomputed and diagonalized, thereby enabling $O(r)$ mode integration using IIR filters [Pentland and Williams 1989; James and Pai 2002]. Linear modes have been warped to approximate large deformation kinematics [Choi and Ko 2005].

Closely related to our approach, Key-Point Subspace Acceleration (KPSA) and caching have been proposed to accelerate posing of deformable characters [Meyer and Anderson 2007]. The selection and use of key points is analogous to our selection of cubature points (or key elements). Unfortunately, KPSA does not solve the problem of estimating subspace forces and Jacobians associated with subspace deformations (although it was never intended to). For example, KPSA applied to subspace forces results in nonconservative force models with nonsymmetric Jacobians due to KPSA's use of least-squares estimation. Instead, it is more natural here to formulate reduced forces in terms of subspace derivatives of strain energy using the solid foundations of numerical integration.

Articulated subspace deformation models are commonplace in character animation, and can be used with cubature optimization. Pose space deformation interpolates shape correction coefficients as a function of pose [Lewis et al. 2000]. For physics-based models [Kry et al. 2002], cubature could assist with estimating pose-specific subspace corrections potentially avoiding high-dimensional interpolation and training difficulties.

An alternative to runtime integration of **g** over **Ω** is to precompute a fast model of **f**(**q**) using various system modeling techniques, such as data-driven interpolation using radial basis functions [Nelles 2000] or neural networks [Grzeszczuk et al. 1998]. Other approaches tabulate forces indirectly using compressed motion libraries for runtime playback, but simulation is done in an unreduced setting [James and Fatahalian 2003]. In such approaches, challenges include guaranteeing adequate data and model training, supporting high subspace dimensionality, avoiding over-fitting, and ensuring energy conservation, passivity, and stability. To a large extent, cubature avoids problems associated with approximating **f**(*q*) (or dynamics) by estimating and storing just 2*n* cubature values (*n* sample indices, and *n* nonnegative weights), thereby exploiting the redundant spatial structure of subspace deformation, and the energy integrand's functional structure.

3 Subspace Deformation Model

Subspace Kinematics

Given the time-dependent parameters of an *r*-dimensional subspace deformation, $\mathbf{q} = \mathbf{q}(t) \in \mathbb{R}^r$, the subspace deformation is specified pointwise by the deformation operator, $\varphi = \varphi(X;\mathbf{q})$,

$$x = \varphi(X;\mathbf{q}) \in \mathbb{R}^3, \tag{7}$$

where $x \in \mathbb{R}^3$ is the deformed image of the undeformed material point, $X \in \mathbb{R}^3$. Computationally, we assume that the cost of evaluating a deformed point using (7) is $O(r)$ flops. The partial derivatives of φ are important kinematic quantities: (1) the *deformation gradient*,

$$F = F(X,\mathbf{q}) = \frac{\partial x}{\partial X} \in \mathbb{R}^{3\times 3} \tag{8}$$

and (2) the *displacement sensitivity matrix*,

$$U = U(X,\mathbf{q}) = \frac{\partial x}{\partial \mathbf{q}} \in \mathbb{R}^{3\times r}. \tag{9}$$

Materials are defined via the strain energy density, $\psi(X;\mathbf{q})$, which are used to evaluate the deformation potential energy and subspace force integrals (2-3). Materials used in this paper are given in Appendix A.

Discrete Setting

To support reduced-order model construction, we will employ an underlying discrete model. Without loss of generality, our implementation uses tetrahedral finite element models with linear shape functions. We will refer to nodal positions as a vector of position quantities: given N_v nodal vertices, let the unde-formed material positions be $\mathbf{X} = (\mathbf{X}_1, \mathbf{X}_2, \ldots, \mathbf{X}_{N_v})^{\mathrm{T}} \in \mathbb{R}^{3N_v}$, and the deformed positions be $\mathbf{x} = (\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_{N_v})^T \in \mathbb{R}^{3N_v}$, such that $\mathbf{x}_i = \varphi(\mathbf{X}_i;\mathbf{q})$. An important displacement-like matrix quantity is the gradient of the position with respect to **q**,

$$\frac{\partial \mathbf{x}}{\partial \mathbf{q}} \equiv \mathbf{U} = \mathbf{U}(\mathbf{q}) \in \mathbb{R}^{3N_v\times r}. \tag{10}$$

For the important case of linear shape models, **U** is a constant matrix of displacement modes, which has the important consequence of making the mass matrix **M** constant.

4 Discrete Cubature

We use optimization to estimate an n-point cubature scheme (5) that approximates the reduced force $\mathbf{f}(\mathbf{q})$ integral in (3). It follows that our approximation of the gradient of the reduced force vector, or the stiffness matrix, is

$$\mathbf{K}(\mathbf{q}) \approx \sum_{i=1}^{n} w_i \frac{\partial \mathbf{g}(X_i;\mathbf{q})}{\partial \mathbf{q}^T}. \tag{11}$$

Observe that by choosing nonnegative weights, $w_i \geq 0$, we are guaranteed that **K** inherits the same semi-definiteness properties of the integrand.

Element-based Cubature Schemes

Cubature optimization would suggest considering continuous positions, $X \in \mathbf{\Omega}$, and thus a continuous-valued, constrained optimization problem. However, in reduced-order modeling we may only have access to discrete representations of the integrand. Therefore, we consider discrete optimization schemes, wherein the candidate cubature "points" (and hence **g**) are chosen from a finite set.

In our implementation, we use linear tetrahedral finite elements with constant deformation gradients. The tetrahedrons are features which can be seen as generalized cubature "points." The reduced-force integrand $\mathbf{g}(X_i;\mathbf{q})$ of any point $X_i \in \mathbf{\Omega}$ is equal to the response of the containing element.

Complexity of Cubature Evaluation

The cost of evaluating internal forces (5) using an n-point cubature scheme is $O(rn)$, since each cubature point's contribution can be accumulated in $O(r)$ operations (assuming dense matrices and global deformation support). For example, a tetrahedron used in a cubature scheme can be deformed in $O(r)$ operations using (7), the resulting 12-vector of vertex forces can be computed at $O(1)$ cost, and these forces can be projected into subspace force contributions using a 12-by-r matrix-vector multiply at $O(r)$ cost. Evaluating the stiffness matrix approximation (11) involves $O(r^2n)$ cost (assuming dense matrices). For example, a 12-by-12 tetrahedral stiffness matrix, $\mathbf{K}_e$, can be computed using (7) at $O(r)$ cost, however computing its contribution to the r-by-r subspace stiffness matrix $\mathbf{K}(q)$ involves a subspace projection of the form, $\mathbf{U}_e^T \mathbf{K}_e \mathbf{U}_e$, which incurs an $O(r^2)$ cost.

Assuming $n \propto r$ (as we observe in practice (§7)), we therefore obtain $O(r^2)$ cost for force evaluation and $O(r^3)$ cost for stiffness evaluation. However, we show in §6.2 that a fast $O(r^2)$ stiffness matrix-vector product is possible if the stiffness matrix need not be formed explicitly. Finally, while complexity analysis can show costs independent of N, an important result is that subspace computations are still fast in practice (see Figure 3).

5 Optimizing Cubature

We model the cubature optimization problem as a discrete subset selection problem, where we attempt to select cubature points/elements that, when weights are optimized using nonnegative least squares (NNLS), will tend to minimize fitting error of (3). The optimization estimates cubature quality via the error of the subspace force estimate using T training data samples, $\{(\mathbf{f}^{(t)},\mathbf{q}^{(t)})\}t=1\ldots T$ where we compute $\mathbf{f}^{(t)} = \mathbf{f}(\mathbf{q}^{(t)})$ for the shape $\mathbf{q}^{(t)}$ using standard methods,

e.g., subspace projection of unreduced forces [Krysl et al. 2001]. We now describe the procedures for estimating weights (§5.1), greedy cubature optimization (§5.2), training data generation (§5.3), and cubature validation (§5.5).

5.1 Estimating Nonnegative Cubature Weights

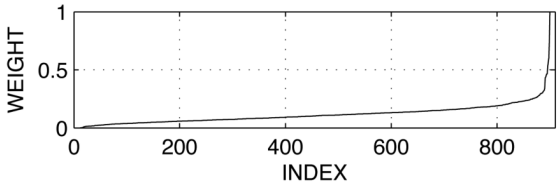
Given a set $\mathscr{S}$ of n cubature points, we estimate cubature weights $\mathbf{w} \in \mathbb{R}^n_+$ (that multiply the integrand samples, $\mathbf{g}_i^{(t)} = \mathbf{g}\left(X_{s_i}; \mathbf{q}^{(t)}\right)$), by minimizing the error in predicting (3) over the training data's T reduced force values, $\mathbf{f}^{(t)}$, $t = 1 \ldots T$. To avoid over-fitting and preserve the spectral properties of stiffness matrices, we estimate nonnegative weights using nonnegative least squares (NNLS) by solving:

$$\begin{bmatrix} \frac{\mathbf{g}_1^{(1)}}{\|\mathbf{f}^{(1)}\|} & \cdots & \frac{\mathbf{g}_i^{(1)}}{\|\mathbf{f}^{(1)}\|} & \cdots & \frac{\mathbf{g}_n^{(1)}}{\|\mathbf{f}^{(1)}\|} \\ \vdots & & \vdots & & \vdots \\ \frac{\mathbf{g}_1^{(t)}}{\|\mathbf{f}^{(t)}\|} & \cdots & \frac{\mathbf{g}_i^{(t)}}{\|\mathbf{f}^{(t)}\|} & \cdots & \frac{\mathbf{g}_n^{(t)}}{\|\mathbf{f}^{(t)}\|} \\ \vdots & & \vdots & & \vdots \\ \frac{\mathbf{g}_1^{(T)}}{\|\mathbf{f}^{(T)}\|} & \cdots & \frac{\mathbf{g}_i^{(T)}}{\|\mathbf{f}^{(T)}\|} & \cdots & \frac{\mathbf{g}_n^{(T)}}{\|\mathbf{f}^{(T)}\|} \end{bmatrix} \begin{bmatrix} w_1 \\ \vdots \\ w_i \\ \vdots \\ w_n \end{bmatrix} = \begin{bmatrix} \frac{\mathbf{f}^{(1)}}{\|\mathbf{f}^{(1)}\|} \\ \vdots \\ \frac{\mathbf{f}^{(t)}}{\|\mathbf{f}^{(t)}\|} \\ \vdots \\ \frac{\mathbf{f}^{(T)}}{\|\mathbf{f}^{(T)}\|} \end{bmatrix} \iff \mathbf{A}\mathbf{w} = \mathbf{b}, \tag{12}$$

subject to nonnegativity constraints, $\mathbf{w} \geq \mathbf{0}$. Here $\mathbf{A}$ is a dense rT-by$n$ matrix, and $\mathbf{b}$ is an rT-vector. Each r-vector in row t of $\mathbf{A}$ and $\mathbf{b}$ is scaled by $\|\mathbf{f}^{(t)}\|^{-1}$ in order to *minimize relative error instead of absolute error*. Larger absolute errors will be tolerated for larger training forces, so a small number of very large samples will not distort the fitting process.

Such problems can be solved efficiently using available NNLS implementations [Lawson and Hanson 1974]. In practice, we can estimate nonnegative cubature weights with one NNLS call in less than a minute for a relatively complex model with $T = 1000$ poses, r=100 dimensions, and n=200 samples. However, for challenging examples, such as the "Menger shell" (Figure 2), NNLS calls can be expensive: 36 minutes for $T = 1000$, r=200, and n=800. Given that solve times scale roughly as $O(rTn^2)$, we will address the superlinear scaling of NNLS with n and r later in §5.4.

Example sorted nonnegative weights (for "Menger shell") are:



Error estimator, ε—In subsequent optimizations, we choose to minimize RMS relative L2-norm error over all samples, as described by the following error metric:

$$\varepsilon = \sqrt{\frac{1}{T}\sum_{t=1}^{T} \frac{\|\mathbf{f}^{(1)} - \mathbf{f}^{(t)}\|^2}{\|\mathbf{f}^{(t)}\|^2}}. \tag{13}$$

All following training and validation convergence plots were done using this metric. Given the scaling by $\|\mathbf{f}^{(t)}\|^{-1}$, this is equivalent to the relative residual error $\|\mathbf{r}\|/\|\mathbf{b}\|$ optimized by the NNLS problem and the greedy algorithm that follows.

5.2 Greedy Estimation of Cubature Points

We propose a simple, iterative, greedy, subset-selection algorithm[1]. At every iteration, we add a key element e such that $\mathbf{g}_e$ is the most positively parallel to the current NNLS residual. This will reduce the size of the residual the most. Then, we update the residual and iterate again. The algorithm is as follows:

GREEDYCUBATURE($\mathbf{A}$, $\mathbf{b}$, TOL)

1. $\mathcal{S} \leftarrow \varnothing$
2. $\mathbf{r} \leftarrow \mathbf{b}$
3. **while** $\|\mathbf{r}\|/\|\mathbf{b}\| > TOL$
4. $\mathcal{C}$ SELECTCANDIDATEPOINTS($\mathcal{S}$)
5. $e \leftarrow \arg\max_{e \in \mathcal{C}} \frac{\mathbf{g}_e^T \mathbf{r}}{\| \mathbf{g}_e \| \| \mathbf{r} \|}$
6. $\mathcal{S} \leftarrow \mathcal{S} \cup \{e\}$
7. $\mathbf{w} \leftarrow \text{NNLS}(\mathbf{A}_{\mathcal{S}}, \mathbf{b})$
8. $\mathbf{r} \leftarrow \mathbf{b} - \mathbf{A}_{\mathcal{S}}\mathbf{w}$
9. **return** $(\mathcal{S}, \mathbf{w})$

Here $\mathcal{S}$ is the set of key elements, $\mathbf{r}$ is the current NNLS residual, $\mathbf{w}$ is the vector of cubature weights, and $\mathbf{A}_{\mathcal{S}}$ is the $\mathbf{A}$ matrix with columns corresponding to elements in $\mathcal{S}$.

Lazy Evaluation—The sub-function SELECTCANDIDATEPOINTS must choose the remaining elements/points most likely to reduce the residual error. The most thorough implementation is to return all remaining elements not already in $\mathcal{S}$. However, for high-resolution meshes, storing or calculating the $\mathbf{g}_e$ vectors for all elements is impractical. Instead, we pick a random subset of the remaining elements and calculate the $\mathbf{g}_e$ vectors needed for the argmax step. The full $\mathbf{A}$ matrix is never actually computed or stored, so the algorithm has modest memory requirements, even for meshes with hundreds of thousands of elements. These column computations are also trivial to parallelize.

One adjustable parameter of the algorithm is the number of random candidates, $|\mathcal{C}|$, to consider at each iteration. We found that for our examples, increasing the value of $|\mathcal{C}|$ beyond 1000 elements does not significantly improve the quality of the cubature after 10% error is reached. Figure 4 demonstrates that considering only 100 candidates per iteration is about the same as considering 10000, suggesting that optimal selection at any greedy step is not critical.

5.3 Training Data Generation

Various methods may be used to collect cubature training samples:

- A simple approach is to use pre-simulated motion for reanalysis [Krysl et al. 2001] of non-interactive simulation conditions, possibly with forcing variations, e.g., in gravity, wind, or inflation conditions of a balloon. The $\mathbf{q}$ values produced during the simulation can then be recorded to disk and sampled for training and validation.

[1]Our greedy algorithm is analogous to one proposed for multipole source placement [James et al. 2006], although here the data is real-valued, and NNLS solves are used at each iteration.

- Manual interaction (or user sketches) with a reduced simulation using fully projected forces can be used, but can be slow for large meshes. To work around this, manual input can be recorded and then simulated off-line [Barbič and James 2005].
- For a linear modal analysis basis used for small-strain subspace integration [Bathe 1996], training data can be generated automatically by randomly sampling a Gaussian distribution for each mode (component of **q**), with standard deviations proportional to the inverse of the mode's frequency.

Once the $\mathbf{q}^{(t)}$ values are known for the poses, we can lazily compute the columns of **A** for training, as needed.

5.4 Optimization Complexity Analysis

Scaling with *N*—The algorithm cost and memory requirements are linear in the size of discrete model, which is mandatory for reduced-order modeling. This *N*-dependence arises from the computation of *T* training force values, $\mathbf{f}^{(t)}$. The "Menger shell" model illustrates that our method can support large tetrahedral models.

Scaling with *r*—For higher rank models, one may need hundreds of cubature points to achieve low error. For such problems, the optimization process can become impractical. Informally, NNLS exhibits $O(rTn^2)$ complexity, so that high *r* and *n* values can be slow. To limit this bottleneck, we use *subset training*, wherein NNLS regression is done on a small subset of T_s training samples. Before the NNLS call, we randomly choose this subset, perform NNLS only on the subset to update the residual, and then on the next iteration we greedily select the point most fit with respect to the residual. Although the residual is inaccurate, each iteration is much faster. However, one can occasionally do NNLS on the full training set to check the actual error and output a final cubature rule. In our experiments, we used $T_s = 10$ and do a full regression every $O(r)$ iterations to obtain cubature schemes that appear to achieve training errors within 5% of comprehensively trained (T_s=T) rules. For high-rank models, such as $r = 200$ for the detailed Menger shell, optimization times became hours instead of days.

Complexity Summary—The optimization bottleneck is repeated calls to the NNLS routine. Characterizing the cost of NNLS as $O(rTn^2)$, then since every iteration calls NNLS with one more cubature sample the total complexity is $O(rTn^3)$. If only a subset of the training set is used each iteration the cost is $O(rT_sn^3)$.

5.5 Cubature Validation

In some ways, the requirement of training data is a weakness of cubature optimization. However, a strength of cubature optimization, perhaps due to the fact that the method only learns cubature points and nonnegative weights, is that it resists over-fitting. To assess the quality of a given cubature rule, we evaluate it on a validation set, $\{(\mathbf{f}^{(v)},\mathbf{q}^{(v)})\}_{v=1\ldots V}$, obtained in ways similar to training set generation (§5.3). To estimate validation error, we use the training error metric (13) on the validation set. Our experiments show that cubature optimization is surprisingly resistant to over fitting: for all examples, the training and validation convergence plots are nearly identical. The two representative plots are shown for the rope bridge example in Figure 5.

6 Implementation Details

6.1 Fast Integrand Evaluation

For a constant displacement sensitivity matrix **U**, equation (5) can be accelerated using level-2 BLAS matrix-vector multiplies. Re-shaping the deformation gradient for element *e* as a 9-vector, it can be calculated as $\mathbf{F_e} = \mathbf{E_e}\mathbf{q}+\mathbf{I}$, where $\mathbf{E_e} \in \mathbb{R}^{9\times r}$ can be derived and pre-computed

using **U** (c.f. [Barbič 2007]). Furthermore, all deformation gradients for cubature elements can be calculated in one matrix-vector multiply by stacking all $\mathbf{E_e}$ matrices into a single matrix $\mathbf{E} \in \mathbb{R}^{9n\times r}$. All energy density gradients with respect to the deformation gradient, $\tilde{\mathbf{g}}_i$, can then be evaluated for each sample element (invoking the constitutive model) in an $O(n)$ loop. Finally, the subspace projection and summation can be evaluated as another matrix-vector multiply, $\mathbf{f} = \mathbf{H}^T [\tilde{\mathbf{g}}_1 \ldots \tilde{\mathbf{g}}_n]$, where the matrix $\mathbf{H} \in \mathbb{R}^{9n\times r}$ can be pre-computed using **U** and the cubature weights w_i. Similar optimizations exist for Jacobian matrix evaluation. In our implementation, this level-2 BLAS optimization provided nearly a two-fold speedup in force evaluation.

6.2 $O(r^2)$ Dense Stiffness Matrix-Vector Products Fast

Fast matrix-vector product evaluation with the stiffness matrix, $\mathbf{K} \in \mathbb{R}^{r\times r}$, is often required for stiffness proportional Rayleigh damping, or implicit integrators. Unfortunately, for dense **U**, the stiffness matrix is dense, and therefore evaluating each cubature sample's contribution to all entries of **K** involves $O(r^2)$ work, so that the total cost of forming the stiffness matrix is $O(r^2 n)$. Fortunately, many iterative Krylov-Newton solvers only require evaluation of matrix-vector products. Consequently we can exploit the fact that each cubature sample only contributes a low constant-rank update to **K**, so that matrix-vector products, **Kv**, can be constructed in $O(rn)$ (or $O(r^2)$ for $n=O(r)$). The **Kv** matrix-vector product can be written

$$\mathbf{Kv} = \sum_{e\in\mathcal{S}} w_e \frac{\partial \mathbf{g}(e;\mathbf{q})}{\partial \mathbf{q}^T}\mathbf{v} = \sum_{e\in\mathcal{S}} w_e \, \mathbf{U}_e^T \, \mathbf{K}_e \, \mathbf{U}_e \mathbf{v} \tag{14}$$

where $\mathbf{U}_e \in \mathbb{R}^{12\times r}$ is the displacement matrix for tetrahedral element e, and $\mathbf{K}_e \in \mathbb{R}^{12\times 12}$ is the element's stiffness matrix. The product $\mathbf{K}_e\mathbf{U}_e$ takes $O(r)$ flops to compute, and can be precomputed and cached. For the **Kv** product, we first compute $\mathbf{t}=(\mathbf{K}_e\mathbf{U}_e)\mathbf{v}$, and then $\mathbf{U}_e^T\mathbf{t}$, both of which take $O(r)$ flops per cubature sample.

7 Results

We now provide numerical results and analysis; please see our accompanying video for animation results. Model statistics and algorithm timings are provided in Table 1. We note that graphical renderings were done using unoptimized implementations of ambient occlusion [Hoberock and Jia 2008], and offline renderings.

Comparison to Gaussian quadrature

Although we do not target 1-dimensional or tensor-product integration applications, out of curiosity, we compared our Greedy-NNLS cubatures to Gaussian quadrature on the [−1,1] interval. To do so, we randomly generated suitably normalized degree-n polynomials using Chebyshev basis functions, then trained our cubature using $N = 1000$ uniformly distributed candidate abscissae. We observe that while an n-point Gaussian quadrature rule can exactly integrate a degree $2n - 1$ polynomial, on average our n-point Greedy-NNLS scheme can integrate only an n-degree polynomial for 0.5% training error. Results are shown in Figure 6.

Scaling with rank for given error tolerances

Analogous to the 1-dimensional comparison, we also investigated how n scales with the rank of the reduced model. For a given reduced model of rank r, how many cubature samples n are necessary to achieve a given error tolerance? Figure 7 shows that the greedy algorithm can produce cubature rules that satisfy the error tolerance with $n=O(r)$ sample elements. The constant factor varies depending on the error tolerance and the geometry of the mesh. As expected, more cubature samples are needed to meet lower error tolerances.

Cubature error analysis and comparisons

Figure 8 provides error convergence plots as a function of the number of cubature samples/elements used. The results indicate that our Greedy-NNLS cubature schemes tend to perform well in practice, especially for low error (ε) and high rank (r) situations, and is dramatically efficient than Monte Carlo integration.

Error Estimation

Classical quadrature schemes often provide error estimators that can be used with little additional computation during evaluation. For example, a Gauss-Kronrod pair consists of an n-point Gaussian rule and a $(2n+1)$-point Kronrod rule that reuses all of the points from the Gaussian rule [Heath 2002]. The Kronrod rule is used as the integral approximation, while the difference between the two rule values is an error estimate.

Analogously, given $2n$ cubature elements, we run NNLS on the first n elements to compute the nonnegative weights of a coarser cubature rule C. The relative error estimate is calculated as $\|\mathbf{f}_C - \mathbf{f}\|/\|\mathbf{f}\|$ where $\mathbf{f}_C$ is the force estimated by the coarser rule C. Like a Gauss-Kronrod pair, this requires no additional evaluations of the integrand. Figure 9 illustrates estimator performance.

Comparison to reduced St.Venant-Kirchhoff

In Figure 10 we compare a cubature-based reduced-force evaluation to an optimized reduced St.Venant-Kirchhoff (StVK) model where reduced forces are exactly represented by an r-vector of polynomials cubic in the components of $\mathbf{q}$ [Barbič and James 2005]. Unlike our $O(r^2)$ approximation, 4 exact evaluation of StVK reduced forces requires $O(r^4)$ operations, and can be prohibitive for larger r values. For a fair timing comparison, we use the optimized level-3 BLAS implementation of the authors of [Barbič and James 2005], and the level-2 BLAS implementation of our reduced-force evaluation (§6.2).

Rope bridge

To illustrate that cubature can be optimized for structures with complicated topologies, we considered a jungle-like rope bridge (see Figure 11). This polygon soup model was discretized into 29,800 tetrahedra using a voxel embedding approach [Barbič and James 2005], and then approximated using an StVK model with tuned parameters. Because their dynamics looked reasonable, we used a linear modal basis shape model, and relied on the StVK nonlinearity to avoid large-deformation distortion.

Reanalysis

Another method of generating a basis is to perform proper orthogonal decomposition (POD), or principle component analysis (PCA), on full simulation data [Krysl et al. 2001]. For our example, we simulate a balloon being inflated by constant air pressure forces (see Figure 12). Our reduced model can then recreate the dynamics of inflation at interactive rates, allowing the user to control the amount of air pressure[2] interactively.

Haptic force-feedback rendering

can exploit reduced-order models for simulation speed, and output-sensitive collision processing [Barbič and James 2007]. Optimized cubature enables complex nonlinear material models for real-time haptic rendering applications. As a proof of concept, we simulated a

[2]Pressure forces were modeled as a linear function of the form, $\mathbf{Pq} + \mathbf{f}_0$, where the constant matrix, $\mathbf{P} \in \mathbb{R}^{r \times r}$, and offset f_0 model the subspace force resulting from a unit pressure applied to an internal tetrahedron's face whose deformed normal is approximated by transforming the material-frame normal by the deformation gradient (a quantity linear in $\mathbf{q}$).

hollow rubber-like structure using both StVK and Mooney-Rivlin material models; without loss of generality, linear modal analysis (LMA) was used to generate an r=10 shape basis. The Mooney-Rivlin material gives clearly different behavior than the StVK model (see Figure 14).

Hyperelastic material test

To investigate the ability of optimized cubatures to approximate nonlinear material response, we performed virtual compression tests on cubes (see Figure 15) made using three hyperelastic constitutive models: St. Venant-Kirchhoff, Mooney-Rivlin and Arruda-Boyce (see Appendix A). In each case we precompute a compression test to estimate a PCA basis and reduced model for optimization and simulation[3]. Results of the compression tests are in Figure 13, and illustrate that optimized cubature can accurately reproduce nonlinear force responses when given a suitable subspace deformation model. For the StVK sample, our parameters were λ = 1000 and μ = 5000. For the Arruda-Boyce, we used μ = 5000, N=5, and K=1×10^5. For the Mooney-Rivlin sample, we used μ_{10}=1×10^5, μ_{01}=10, and K=1×10^5.

Nonlinear modal sound synthesis:

Linear modal analysis is widely used for vibration modeling of effectively rigid objects since the runtime space and time complexity of integrating r modes with an IIR filter is only $O(r)$ flops [James and Pai 2002], after which the modal coefficients $\mathbf{q}$ can be used to evaluate sound radiation [James et al. 2006]. However, for objects such as thin shells, even small deformations can induce nonlinear dynamics, and effectively "coupling" linear modes. To avoid the N-dependent costs of a fully nonlinear vibration analysis, such as in [O'Brien et al. 2001], we apply cubature optimization to nonlinear StVK forces based on sampling the space of linear-mode shapes. Optimized cubature provides an efficient $O(r^2)$ evaluation of subspace forces, suitable for long-time explicit Newmark subspace integration [Krysl et al. 2001]. Subspace integration provides easy access to modal coordinates $\mathbf{q}(t)$ used in sound radiation models (here we use the farfield, low-frequency, monopole radiation model (see (15) in [James et al. 2006])). Results are shown in Figure 2, and the accompanying video. We compared our cubature approximation to a brute-force subspace simulation [Krysl et al. 2001] and obtained nearly identical sounds. However, computing $\mathbf{f}(\mathbf{q})$ by evaluating $O(N)$ unreduced forces, followed by subspace projection (multiplication by $\mathbf{U}^T$), at each explicit timestep was approximately 110× more expensive; 16-core parallelization was used to compute the comparison.

8 Conclusion and Discussion

In summary, optimized cubature is a simple and mathematically sound way to build reduced-order force models for subspace integration. It is not restricted to any particular shape model, and it supports various hyperelastic material models. Cubature can be trained for geometrically complex examples, and fast subspace integration performance can enable interactive graphics, large multibody simulations, or simulations requiring high temporal rates such as haptics and sound synthesis. Although cubature schemes depend on training data, they also are surprisingly robust to over-fitting.

[3]We compress a cube of the material by constraining the nodes of the top-face and lowering them by a small amount in a quasi-static simulation. When the material response reaches equilibrium, we record the total upward forces on the top-face nodes and record the displacement state. We then perform PCA on these recorded states to produce a linear basis, use these states to train a cubature rule, and observe the upward forces reproduced by the cubature rule. Because the top-face nodes are constrained in the original simulation, we need to artificially include them in the PCA basis for training cubature and reduced simulation. If $\mathbf{U}'$ is the basis produced by PCA of the simulation data, then the basis we use for the reduced simulation is $\mathbf{U} = [\mathbf{U}', \mathbf{u}_{top}]$, where $\mathbf{u}_{top}$ is a normalized $3N$ basis vector with positive vertical displacement for the top-face nodes. This allows us to measure the total upward force on the top-face nodes reproduced by the cubature force model by unprojecting the reduced force using $\mathbf{U}$.

Discussion and Limitations

We have provided some examples to provide evidence in support of cubature optimization, however there are many other ways to realize its benefits, and also other limitations to overcome.

Exploiting sparsity is important for fast evaluation and training of high-rank r models, whereas our current analysis is limited to models with dense $\mathbf{U}$ bases. Generating sparse bases and cubature schemes can lead to linear-time $O(r)$ schemes for reduced force and sparse stiffness matrix evaluation. The existence of an $O(r)$ algorithm for reduced force evaluation using dense rank-r displacement bases remains an open problem.

Although cubature schemes support general subspace deformations, we have mostly considered linear (constant $\mathbf{U}$) shape models here, however many other successful models exist, e.g., modal derivatives [Barbič and James 2005]. We have also constructed cubature schemes for articulated and skinned mesh models [James and Twigg 2005], however such models require more efficient implicit Newmark subspace integrators with efficient sparse matrix solves and/or preconditioning for similar performance. Physically based character animation, especially detailed skin deformation and facial animation, are areas likely to benefit from fast cubature schemes. Some shape models may be more susceptible than others to element inversion, and non-element-based approaches for evaluating $\mathbf{g}$ may help. For example, we implemented subspace deformations based on modal warping [Choi and Ko 2005], however we found element-based cubature schemes susceptible to element inversion, presumably due to highly extrapolated element deformations.

Future work includes applying subspace-based cubature optimization to the simulation of physical phenomena other than volu-metric deformable objects, e.g., shells, MEMS, etc. Efficient error estimators allows the possibility of adaptive simulation. For general kinematics, the mass matrix can be time dependent and potentially expensive to evaluate. Optimized cubature might also be used for fast estimates of the mass matrix, $\int_{\Omega} U(\mathbf{x})^T U(\mathbf{x}) \rho(\mathbf{x}) d\mathbf{\Omega}$.

Finally, we have proposed a greedy algorithm for cubature optimization, however, a stronger theoretical footing is desirable for automatic cubature generation. For example, it is tempting to think of cubature as some sort of critical points of some class of functions, similar to the definition of Gaussian quadratures as the zeros of Legendre polynomials and other functions. It appears that similar accuracy may be possible using half as many cubature points if Gaussian-quality cubatures could somehow be learned, thus leading to a two-fold speedup in reduced force evaluation.

Acknowledgments

The authors wish to acknowledge funding and support from the National Science Foundation (CAREER-0430528, EMT-CompBio-0621999), National Institutes of Health (NIBIB/NIH R01EB006615), the Alfred P. Sloan Foundation, Intel, Pixar, The Boeing Company, Autodesk, and NVIDIA. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of the National Science Foundation.

A Material Strain Energy Densities, ψ

This appendix reports strain energy densities, $\boldsymbol{\psi}$, used in this paper [Bonet and Wood 2008]. The St. Venant-Kirchhoff model is

$$\Psi(\mathbf{E}) = \frac{1}{2}\lambda tr(\mathbf{E})^2 + \mu \mathbf{E}:\mathbf{E}, \tag{15}$$

where λ and μ are the Lamé constants, and μ corresponds to the shear modulus. The Arruda-Boyce model uses

$$\Psi(\mathbf{E}) = \mu \sum_{i=1}^{5} \frac{C_i}{N^{i-1}} \left(I_C^i - 3^i \right), \tag{16}$$

where I_C is the first deviatoric strain invariant, μ is the initial shear modulus, and N is the number of rigid links for the model [Liu et al. 2004]. The Mooney-Rivlin constitutive model uses

$$\Psi(\mathbf{E}) = \mu_{10}(I_C - 3) + \frac{1}{2}\mu_{01}\left(I_C^2 - II_C - 6 \right), \tag{17}$$

where II_C is the second deviatoric strain invariant, and μ_{10} and μ_{01} are material constants. The last two models do not enforce incompressibility, so we add a penalty term $K\log(J^2)$ to approximately conserve volume, where K is the bulk modulus, and $J = det(F)$.

References

Baraff, D.; Witkin, A. Dynamic simulation of non-penetrating flexible bodies; Computer Graphics (Proceedings of SIGGRAPH 92); 1992. p. 303-308.

Barbič J, James DL. Real-Time Subspace Integration for St. Venant-Kirchhoff Deformable Models. ACM Trans. on Graphics Aug. 3;2005 24:982–990.

Barbič, J.; James, DL. Time-critical distributed contact for 6-dof haptic rendering of adaptively sampled reduced deformable models; Proceedings of ACM SIGGRAPH Symposium on Computer Animation (SCA 2007); San Diego, CA. 2007.

Barbič, J. Real-time Reduced Large-Deformation Models and Distributed Contact for Computer Graphics and Haptics. Carnegie Mellon University; 2007. PhD thesis

Bathe, K-J. Finite Element Procedures. Vol. second ed.. Prentice Hall; 1996.

Bonet, J.; Wood, RD. Nonlinear Continuum Mechanics for Finite Element Analysis. Vol. second ed.. Cambridge University Press; New York: 2008.

Capell, S.; Green, S.; Curless, B.; Duchamp, T.; Popović, Z. A Multiresolution Framework for Dynamic Deformations; ACM SIGGRAPH Symposium on Computer Animation; 2002. p. 41-48.

Choi MG, Ko H-S. Modal Warping: Real-Time Simulation of Large Rotational Deformation and Manipulation. IEEE Transactions on Visualization and Computer Graphics Jan./Feb. 1;2005 11:91–101. [PubMed: 15631132]

Cotin S, Delingette H, Ayache N. Real-time elastic deformations of soft tissues for surgery simulation. IEEE Transactions on Visualization and Computer Graphics Jan./Mar. 1;1999 5:62–73.

Debunne, G.; Desbrun, M.; Cani, M-P.; Barr, AH. Dynamic Real-Time Deformations Using Space & Time Adaptive Sampling; Proc. of ACM SIGGRAPH 2001; 2001. p. 31-36.

Grinspun E, Krysl P, Schröder P. CHARMS: A Simple Framework for Adaptive Simulation. ACM Trans. on Graphics July 3;2002 21:281–290.

Grzeszczuk, R.; Terzopoulos, D.; Hinton, G. NeuroAnimator: Fast Neural Network Emulation and Control of Physics-Based Models. Proc. of SIGGRAPH 98; Computer Graphics Proceedings, Annual Conference Series; 1998. p. 9-20.

Heath, MT. Scientific Computing: An Introductory Survey. McGraw-Hill Higher Education; New York: 2002.

Hildebrand, FB. Introduction to Numerical Analysis. McGraw-Hill; New York: 1956.

Hoberock, J.; Jia, Y. Chapter 12: High-Quality Ambient Occlusion. In: Nguyen, H., editor. GPU Gems. Vol. 3. Addison-Wesley; 2008. p. 257-274.

James DL, Fatahalian K. Precomputing interactive dynamic deformable scenes. ACM Transactions on Graphics July 3;2003 22:879–887.

James, DL.; Pai, DK. ArtDefo: Accurate real time deformable objects. Proceedings of SIGGRAPH 99; Computer Graphics Proceedings, Annual Conference Series; 1999. p. 65-72.

James DL, Pai DK. DyRT: Dynamic response textures for real time deformation simulation with graphics hardware. ACM Transactions on Graphics July 3;2002 21:582–585.

James DL, Twigg CD. Skinning mesh animations. ACM Transactions on Graphics Aug. 3;2005 24:399–407.

James DL, Barbič J, Pai DK. Precomputed Acoustic Transfer: Output-sensitive, accurate sound generation for geometrically complex vibration sources. ACM Transactions on Graphics July 3;2006 25:987–995.

Kaufmann, P.; Martin, S.; Botsch, M.; Gross, M. Flexible Simulation of Deformable Models Using Discontinuous Galerkin FEM; ACM SIGGRAPH/Eurographics Symposium on Computer Animation; 2008. p. 105-115.

Kry, PG.; James, DL.; Pai, DK. EigenSkin: Real Time Large Deformation Character Skinning in Hardware; ACM SIGGRAPH Symposium on Computer Animation; 2002. p. 153-160.

Krysl P, Lall S, Marsden JE. Dimensional model reduction in non-linear finite element dynamics of solids and structures. International Journal for Numerical Methods in Engineering 2001;51:479–504.

Lawson, CL.; Hanson, RJ. Solving Least Square Problems. Prentice Hall; Englewood Cliffs, NJ: 1974.

Lewis, JP.; Cordner, M.; Fong, N. Pose Space Deformations: A Unified Approach to Shape Interpolation and Skeleton-Driven Deformation. Proceedings of ACM SIGGRAPH 2000; Computer Graphics Proceedings, Annual Conference Series; 2000. p. 165-172.

Liu, Y.; Kerdok, AE.; Howe, RD. A nonlinear finite element model of soft tissue indentation. In: Cotin, S.; Metaxas, D., editors. Proc. Intl. Symp. Medical Simulation (ISMS 2004), Lecture Notes in Computer Science; Springer-Verlag; 2004. p. 67-76.

Meyer M, Anderson J. Key Point Subspace Acceleration and Soft Caching. ACM Transactions on Graphics July 3;2007 26:74:1–74:8.

Nelles, O. Nonlinear System Identification: From Classical Approaches to Neural Networks and Fuzzy Models. Springer Verlag; Dec. 2000

O'Brien, JF.; Cook, PR.; Essl, G. Synthesizing sounds from physically based motion. Proceedings of ACM SIGGRAPH 2001; Computer Graphics Proceedings, Annual Conference Series; 2001. p. 529-536.

Pentland, A.; Williams, J. Good vibrations: Modal dynamics for graphics and animation; Computer Graphics (Proceedings of SIGGRAPH 89); 1989. p. 215-222.

Press, WH.; Teukolsky, SA.; Vetterling, WT.; Flannery, BP. Numerical Recipes in C: The Art of Scientific Computing. Cambridge University Press; New York, NY, USA: 1992. Gaussian quadrature is discussed in section 4.5

Terzopoulos D, Witkin A. Physically Based Models with Rigid and Deformable Components. IEEE Computer Graphics & Applications Nov. 6;1988 8:41–51.

Terzopoulos, D.; Platt, J.; Barr, A.; Fleischer, K. Elastically Deformable Models; Computer Graphics (Proceedings of SIGGRAPH 87); 1987. p. 205-214.

Vakakis, A. Normal modes and localization in nonlinear systems. Kluwer Academic Publishers; 2001.

Witkin, A.; Welch, W. Fast Animation and Control of Nonrigid Structures; Computer Graphics (Proceedings of SIGGRAPH 90); 1990. p. 243-252.

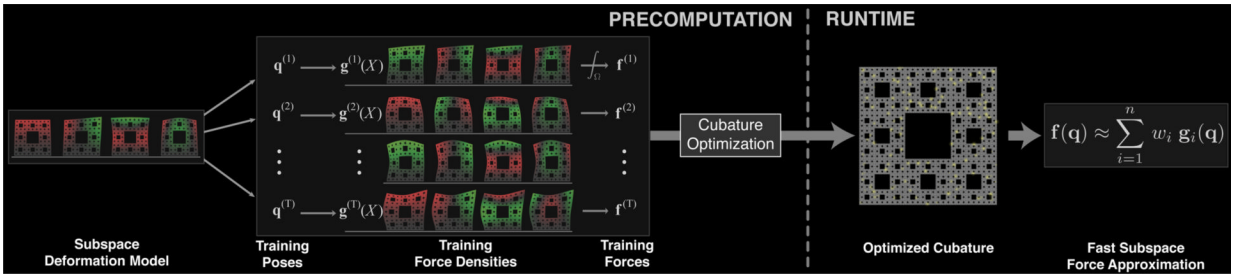


Figure 1. Overview of Cubature Optimization

Given a subspace deformation model with N elements, we generate (sample or simulate) a set of training poses for input to the cubature optimization preprocess. The optimization procedure estimates $n \ll N$ cubature elements/points, and associated nonnegative weights, w_i, such that the the cubature approximation of $\mathbf{f}(\mathbf{q})$ well-approximates the training force/pose data. At runtime, the cubature scheme uses the force response of only the n cubature elements to provide a fast approximation to internal forces, therein accelerating integration of subspace deformation dynamics.

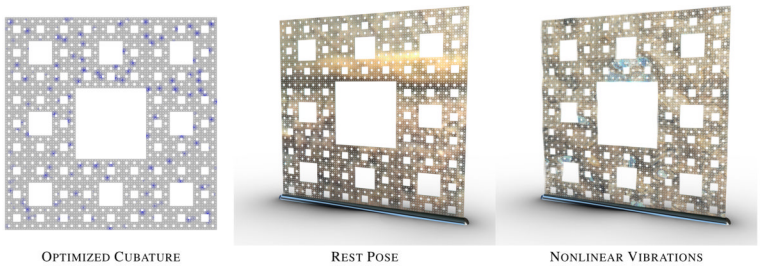


Figure 2. Optimizing cubature for nonlinear modal sound

(Left) A 900-element cubature scheme optimized for integrating the nonlinear 200-mode subspace vibrations of (Middle) a complex Menger-inspired thin shell (one cube thick) modeled with 393216 tetrahedra, and a St.Venant-Kirchhoff material model of aluminum. (Right) Nonlinear shell vibrations (amplified 4× for display). Optimized cubature permits explicit Newmark subspace integration of audio-rate (44.1 kHz) nonlinear sound simulations, with significant and audible nonlinear mode coupling effects, at greatly reduced costs: for a 5.0 second sound clip, nonlinear sound synthesis using optimized cubature and subspace integration ($\Delta t = 1ms/88.2$) required 3.5 single-core hours for subspace vibration (and radiation calculations), whereas sounds integrated using a parallelized implementation of subspace-projected unreduced forces [Krysl et al. 2001] required 4 days on 16 cores. The resulting sounds are virtually indistinguishable.

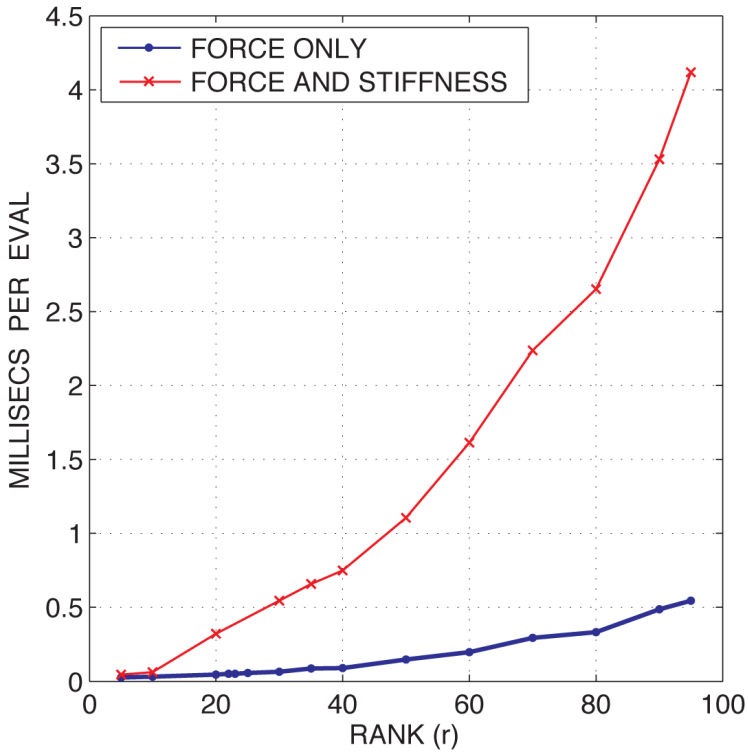


Figure 3. Complexity of force and stiffness evaluation

Assuming $n = O(r)$ cubature samples, internal force evaluation costs $O(r^2)$ flops, whereas dense stiffness matrix evaluation is $O(r^3)$ (*although a fast matrix-vector multiply exists* (§6.2)). *These timings were done using n = r cuba-ture schemes.*

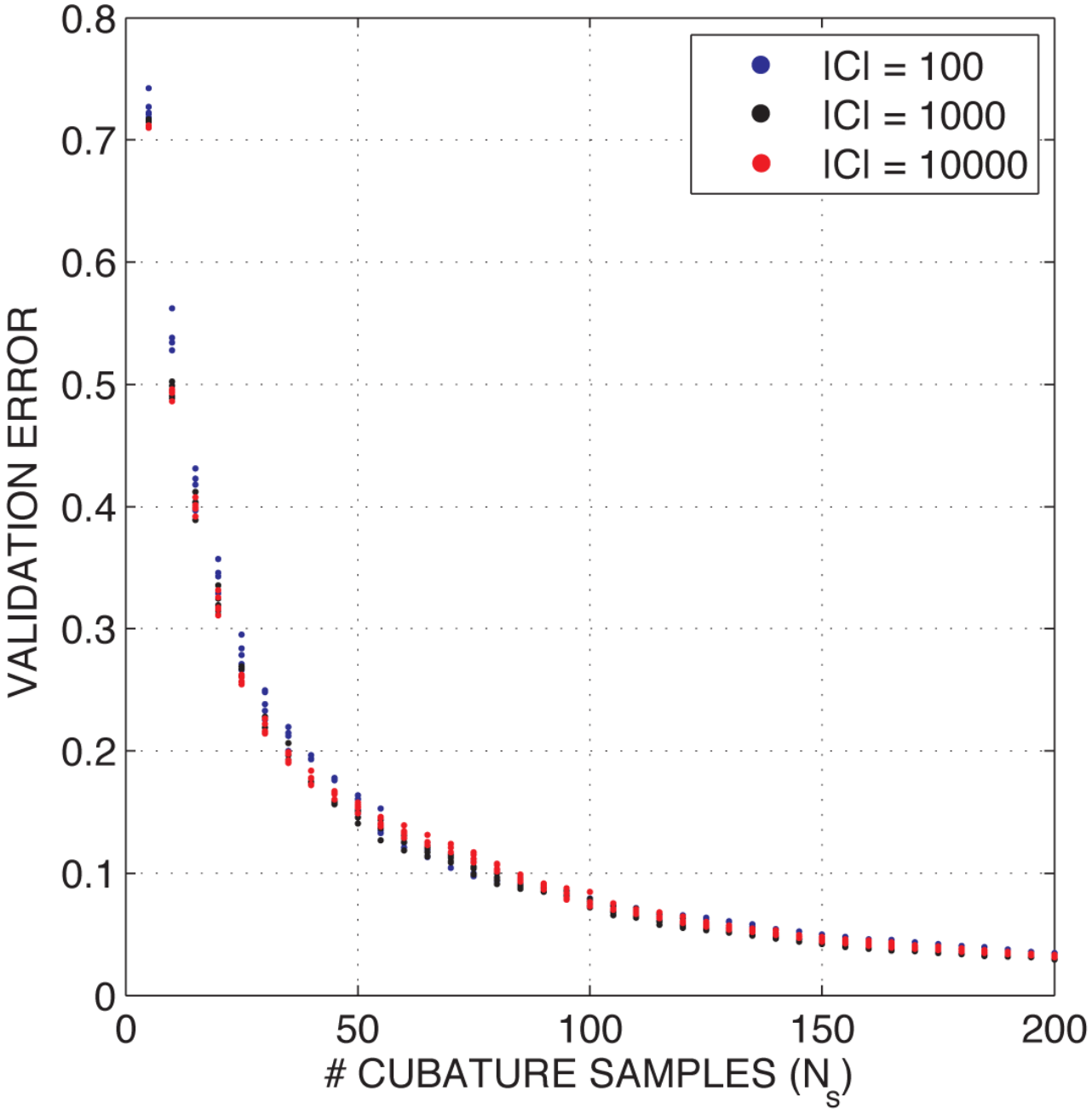


Figure 4. Comparison of different values of |ℂ|

Below 10% error, the quality of the cubature does not degrade significantly when considering less candidates per iteration. Thus, the greedy cuba-ture optimization algorithm is practical for high-resolution meshes. The bridge model of 29,800 elements was used for this plot.

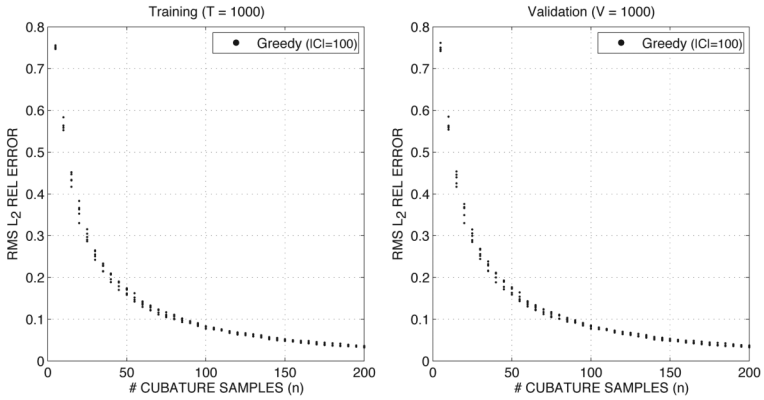


Figure 5. Training and validation convergence plots

for the rope bridge example (*r*=100) illustrate characteristic resistance of cubature optimization to over-fitting. Five cubatures were optimized for each n value.

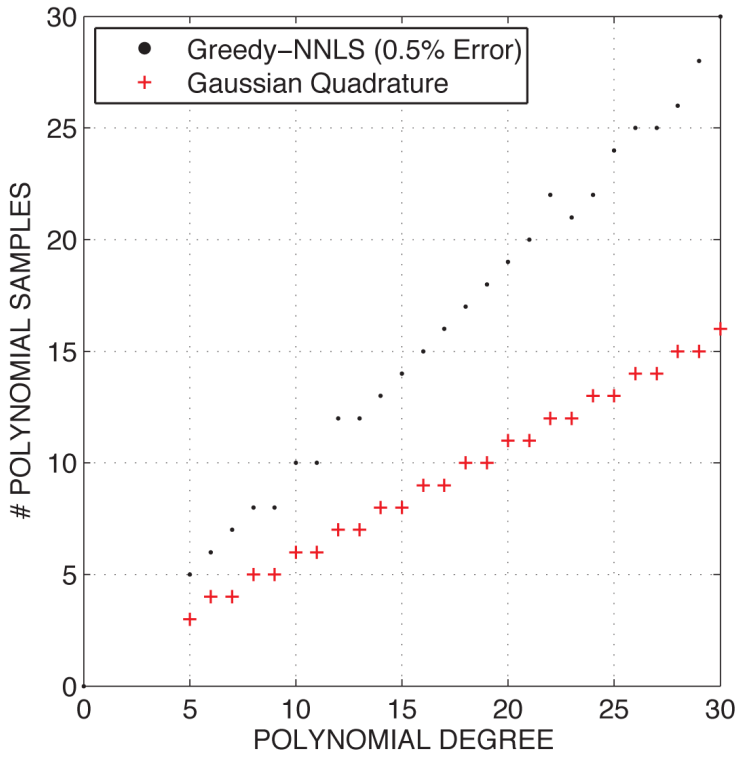


Figure 6. Comparison to Gaussian quadrature

In the 1-D case, our greedily-optimized quadrature rules require about twice as many samples as the corresponding Gaussian quadrature rules to achieve 0.5% error.

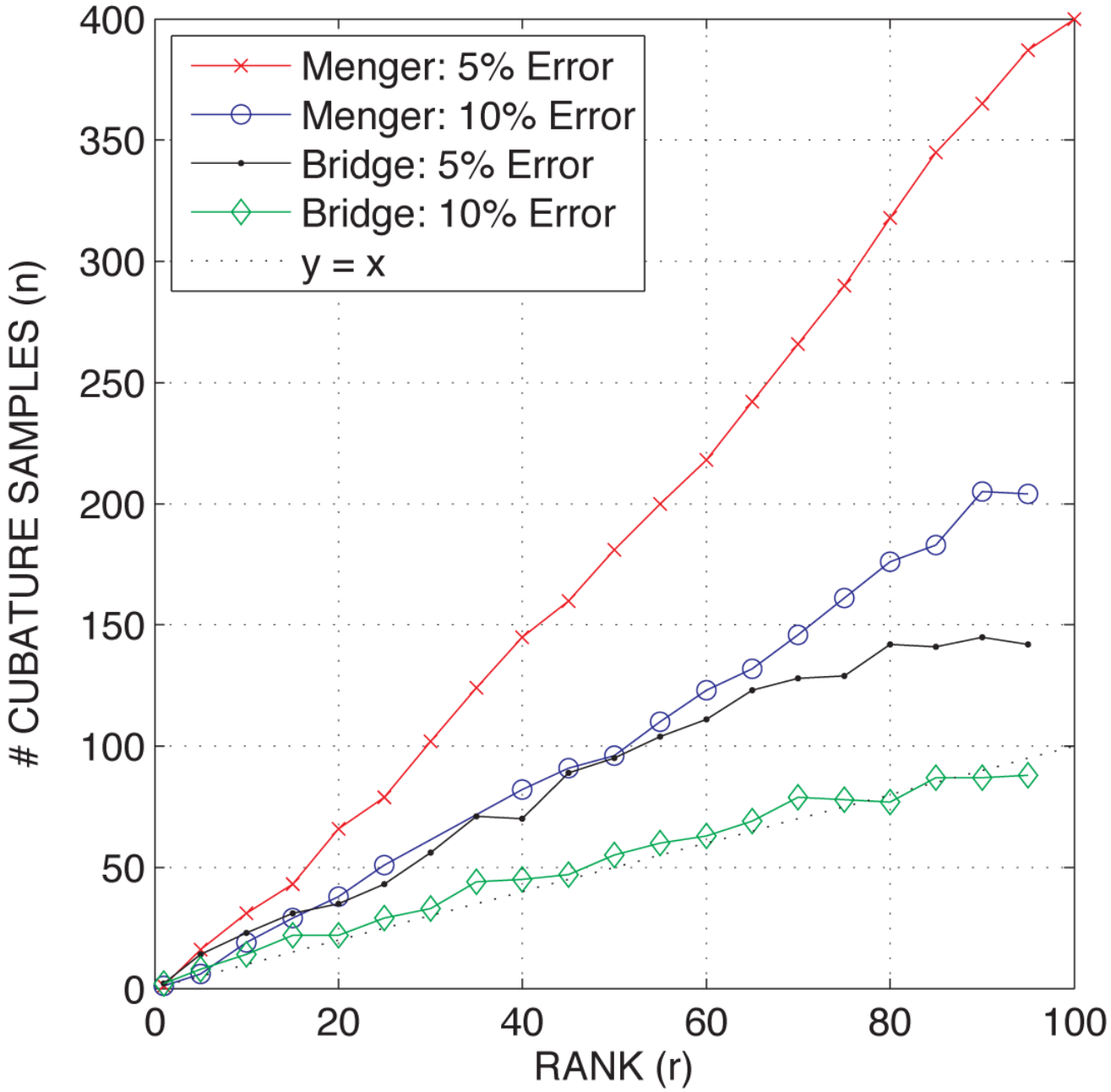


Figure 7. Scaling with rank for given error tolerances

In practice, $n = O(r)$ cubature samples are sufficient to achieve a given error tolerance, but the constant factor depends on the example and the desired error tolerance.

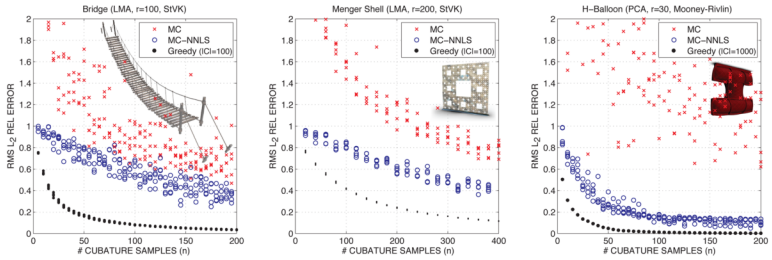


Figure 8. Cubature Convergence Analysis

Plots of reduced-force training error, ε, as a function of the number of key elements, n. Results are shown for the uniformly weighted Monte Carlo scheme (MC), MC-sampled positions but NNLS-estimated weights (MC-NNLS), and our greedy approach (Greedy). We used T = 1000 training samples for the bridge and Menger shell, and T = 50 (from the full inflation simulation) for the balloon. For these plots, we do a full NNLS regression per-iteration, so $T_s = T$ (no subset training), in order to get accurate error measurements, however, subset training (§*5.4) accelerates optimization, e.g., of our n*=900 *Menger cubature scheme (ε*=6.4%*).*

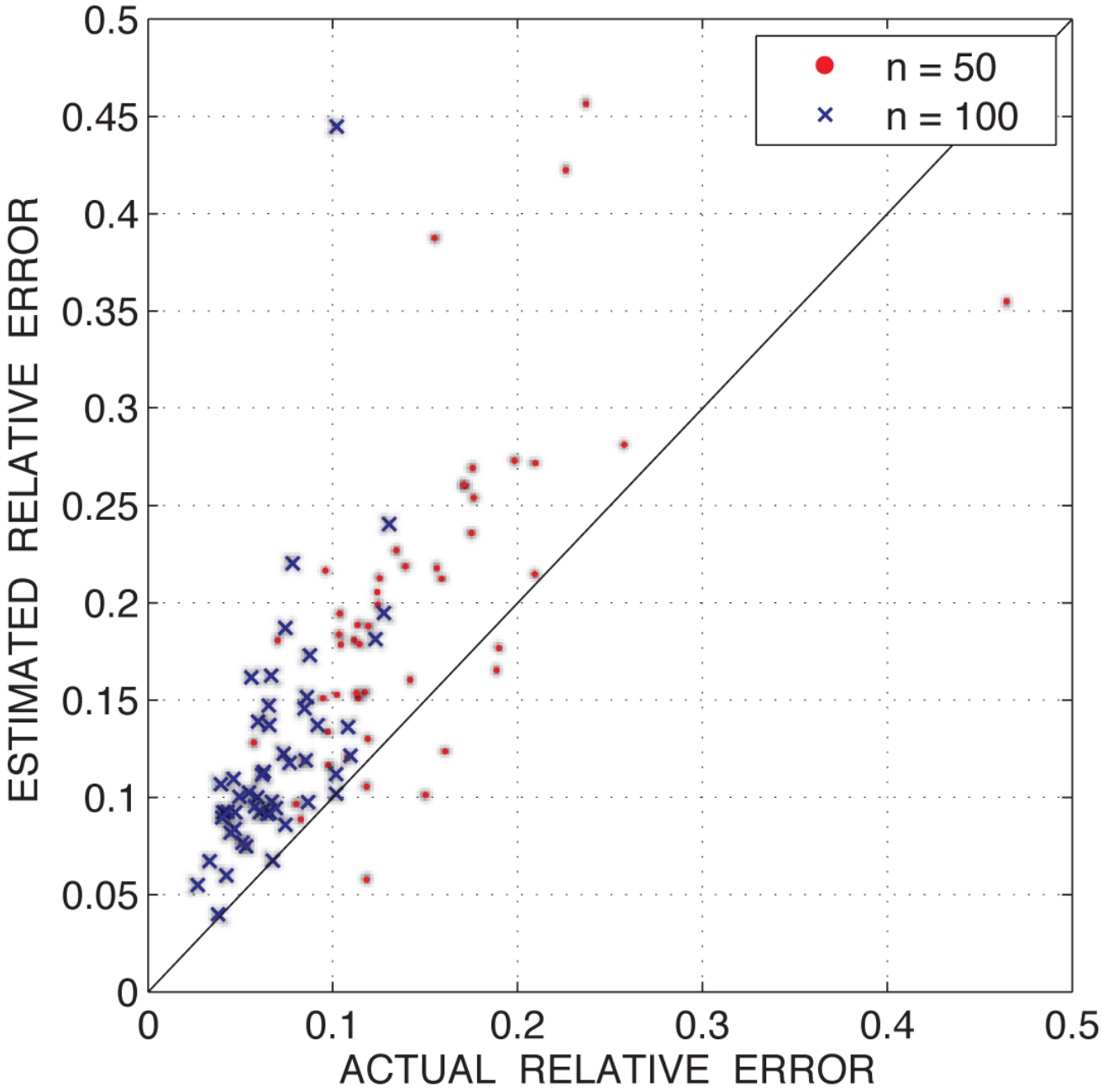


Figure 9. Error Estimator

results for two cubature pairs: n=50 *(&* 25*) and n*=100 *(&* 50*) samples (rope-bridge example with r* = 100). Each point in the plot represents the validation error of a random deformable pose. Here the n=100 error estimate appears conservative, since its estimate is higher than the actual error, i.e., points are above the "y=*x line.*"

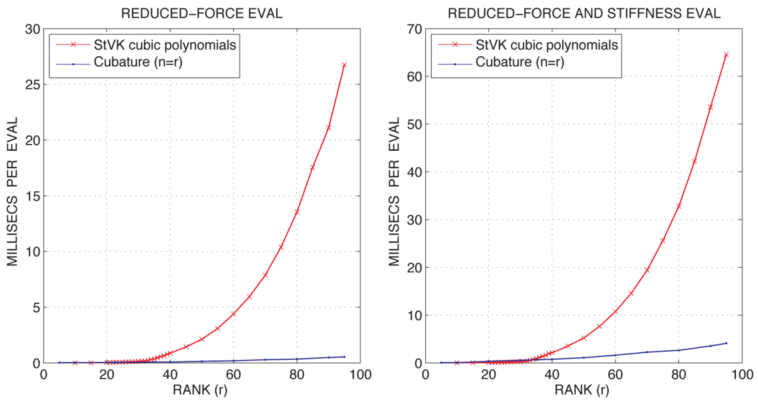


Figure 10. Comparison to StVK reduced-force evaluation costs

We compared a St.Venant-Kirchhoff (StVK) polynomial model, versus a cubature-based approximation using n = *r elements. Our* $O(r^2)$ *reduced-force evaluation becomes faster around r* = 23, *and scales substantially better that the* $O(r^4)$ StVK polynomial model. When evaluating both reduced forces and stiffness matrices ($O(r^3)$), *the cubature scheme becomes faster around r*=35.

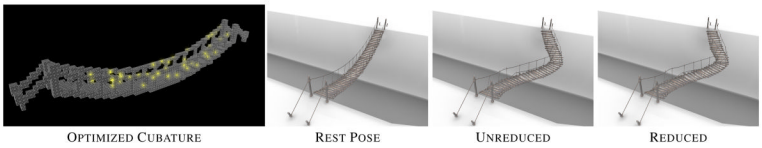


Figure 11. Optimizing cubature for an embedded rope bridge

(Far-Left) Rasterized polygon-soup bridge with cubature tetrahedra/points highlighted. (Mid-Left) The undeformed bridge is subjected to a sideways impulse at its midpoint, and simulated using the (Mid-Right) unreduced implicit Newmark integrator, and the (Far-Right) implicit Newmark subspace integrator. Optimized cubature achieved 200 timesteps/second (implicit Newmark subspace) whereas unreduced implicit Newmark (with PARDISO solver) achieved 0.25 timesteps/sec.

Figure 12. Reanalysis of balloon inflation
using cubature optimized from PCA-based simulation data. The dynamic Mooney-Rivlin rubber balloon can be inflated interactively.

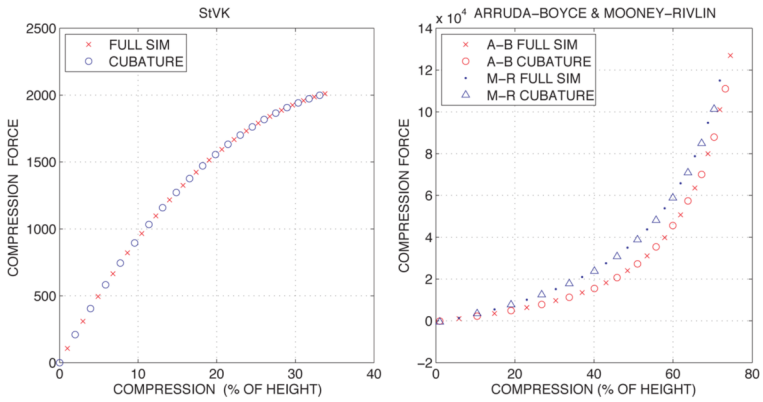


Figure 13. Virtual compression test

The force response reproduced by optimized cubature closely matches the full simulation for all compression amounts tested. The Arruda-Boyce and Mooney-Rivlin materials could not be compressed beyond 85% due to element inversion, whereas the St.Venant-Kirchhoff (StVK) material softened significantly at about 35% compression, ultimately leading to inversion and stiffness matrix indefiniteness.

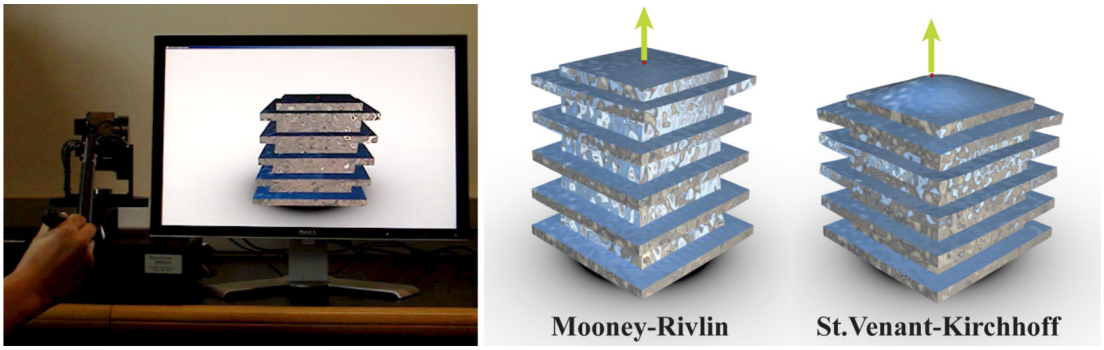


Figure 14. Haptic force-feedback rendering examples
The implicit Newmark subspace integrators could deliver 5000 Hz rates.

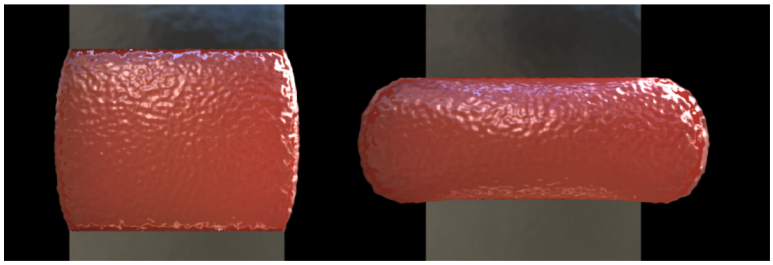


Figure 15.
Compressing the Arruda-Boyce material cube

Table 1

Model statistics including number of tetrahedra (N_{tet}); rank (r) of linear deformation basis **U**; number of cubature points/elements (n); time to evaluate both **f**(**q**) *and* **K**(**q**); time to solve an r-by-r dense symmetric positive-definite linear system using LAPACK for implicit Newmark subspace integration; time-stepping rate of Newmark subspace integration used in demos, for either explicit or semi-implicit (one Newton-Raphson iteration) schemes; time for the cubature optimization preprocess (using 4-16 Xeon cores); relative training error **ε** of the resulting cubature scheme.

Model	Material	N_{tet}	Basis	Rank (*r*)	*n*	f Eval	K Eval	Solve	Timesteps/sec	Optimization	Error
Bridge*	St. Venant-Kirchhoff	29,800	LMA	100	100	0.48 ms	3.0 ms	0.29 ms	265 (implicit)	47 secs	8%
Balloon*	Mooney-Rivlin	118,208	PCA	30	90	0.15 ms	0.88 ms	0.03 ms	943 (implicit)	34 secs	1.5%
Menger*	St. Venant-Kirchhoff	393,216	LMA	200	900	9.7 ms	75 ms	2.2 ms	101 (explicit)	1.8 hours	6.4%
Haptic†	St. Venant-Kirchhoff	21,376	LMA	10	32	0.02 ms	0.19 ms	0.008 ms	4230 (implicit)	10 secs	3%
Haptic†	Mooney-Rivlin	21,376	LMA	10	12	0.01 ms	0.07 ms	0.008 ms	8750 (implicit)	5 secs	3%

*With the exception of cubature optimization, the timing experiments were done on a 2.4GHz Intel Core2

†*The experiments on a 3.0GHz Intel Xeon. The Intel Math Kernel Library was used for BLAS operations (dual-core enabled). Cubature training was done using the Greedy algorithm with* $|\mathbb{C}| = 100$ *(except for the balloon and haptic examples which used* $|\mathbb{C}| = 1000$*) and subset training (*$T_S = 10$*, full NNLS solves every r/2 iterations; except haptic examples use comprehensive training,* $T_S = T$*).*