Supplemental Material

A DERIVATION OF PROPOSITION 3.1

In the limit $\lambda \longrightarrow \infty$, our metric-preserving constraint in Equation 4 is imposed precisely, forcing the subspace function to be linear and subject to mass orthonormality constraints (see Proposition B.1 in Appendix B):

$$f_\theta(z) = Az + b \quad \text{where} \quad A^T M A = \sigma^2 I_{d \times d}. \tag{10}$$

where $A \in \mathbb{R}^{n \times d}$, $b \in \mathbb{R}^n$, and $\theta = (A, b)$.

If we take the limit of the scaling factor $\sigma \longrightarrow 0^+$, the entries of the matrix A become small due to the mass weighted orthonormality constraints in Equation 10. Hence, for sufficiently small σ, we can expect that the outputs of our subspace function f in Equation 10 will not stay far from the value of the inhomogeneous term b.

To understand behavior of our model in this perturbative regime, we can approximate our original formulation in Equation 4 using a truncated second order Taylor expansion of the potential energy $E_{\text{pot}}(\cdot)$ centered at b:

$$\min_{A,b} \mathbb{E}_{z \sim \mathcal{N}} \left[E_{\text{pot}}(b) + g(b)^T Az + \frac{1}{2} z^T A^T H(b) Az \right] \quad \text{s.t.} \quad A^T M A = \sigma^2 I. \tag{11}$$

where $g(x)$ and $H(x)$ are the gradient and Hessian of the potential energy E_{pot}, respectively.

Since $\mathcal{N}$ has mean zero, the linear term vanishes from the problem above. Moreover, recognizing it as the Girard-Hutchinson trace estimator [Girard 1987; Hutchinson 1989], we can manipulate the quadratic term as follows:

$$z^T A^T H(b) Az = tr(z^T A^T H(b) Az) = tr(A^T H(b) Azz^T),$$

since a scalar is its own trace and by the property $tr(AB) = tr(BA)$. Since $\mathcal{N}$ has the identity matrix as its covariance, we arrive at the following simplification of Equation 11:

$$\min_{A,b} \left[E_{\text{pot}}(b) + \frac{1}{2} tr\left(A^T H(b) A \right) \right] \quad \text{s.t.} \quad A^T M A = \sigma^2 I. \tag{12}$$

As $\sigma \longrightarrow 0$, the second term of the objective in Equation 12 becomes negligible, and the potential energy term dominates. Fixing b and optimizing for A yields a generalized eigenvalue problem.

Hence, we have motivated that as $\sigma \longrightarrow 0$ and $\lambda \longrightarrow \infty$, we recover Equation 5. This formulation is exactly the classical linear method for modal analysis, discussed e.g. in Shabana [1991].

B AFFINE PROPERTY

Proposition B.1. *Suppose $f(z) : \Omega \longrightarrow \mathbb{R}^n$ is C^1 on an open, connected domain $\Omega \subseteq \mathbb{R}^d$. Then, $f(z)$ satisfies Equation 2 with equality for all $z, z' \in \Omega$ if and only if $f(z) = Az + b$ for some $A \in \mathbb{R}^{n \times d}, b \in \mathbb{R}^n$ with $A^T M A = \sigma^2 I_{d \times d}$.*

Proof. We start with the Lipschitz constant expression:

$$|f(z_a) - f(z_b)|_M = \sigma |z_a - z_b| \quad \forall z_a, z_b \in \Omega,$$

where σ is the positive Lipschitz constant. Squaring this expression implies

$$|f(z_a) - f(z_b)|_M^2 = \sigma^2 |z_a - z_b|^2 \quad \forall z_a, z_b \in \Omega.$$

Since $f \in C^1(\Omega)$, taking the derivative w.r.t. z_a implies

$$J(z_a)^T M (f(z_a) - f(z_b)) = \sigma^2 (z_a - z_b) \qquad J(z) := \left(\frac{\partial f}{\partial z} \right).$$

Taking the derivative of this expression w.r.t. z_b implies

$$J(z_a)^T M J(z_b) = \sigma^2 I_{d \times d} \quad \forall z_a, z_b \in \Omega. \tag{13}$$

Since the expression holds in the $z_a = z_b$ case, applying the expression above multiple times shows

$$(J(z_a) - J(z_b))^T M (J(z_a) - J(z_b)) = 0.$$

Since $M \succeq 0$, we thus have

$$J(z_a) = J(z_b) = \text{const.} \quad \forall z_a, z_b \in \Omega.$$

Since f has a constant Jacobian in Ω, it is automatically affine:

$$f(z) = Az + b \quad \forall z \in \Omega.$$

Moreover, $A^T M A = \sigma^2 I_{d \times d}$ thanks to Equation 13.

To prove the reverse direction, note that the relationship

$$|A(z_a - z_b)|_M \quad s.t. \quad A^T M A = \sigma^2 I_{d \times d}$$

implies

$$\sigma |z_a - z_b| .$$

□

C ADDITIONAL DETAILS

This sections gathers additional implementation and experimental details.

C.1 Selecting Hyperparameters

Like most learning-based approaches, our method requires a choice of hyperparameters to weight the objective function, in our case the penalty strength λ and metric scaling parameter σ. To be clear, adjusting two parameters is a modest burden as neural networks go, and our networks train very quickly (Section 4.2); we discuss hyperparameter selection in-depth here to facilitate the application of our method to new physical systems. Importantly, although hyperparameters may need new settings for new classes of systems (*e.g.* cloth *vs.* kinematic mechanisms), they can be reused to fit many instances of a particular system (*e.g.* many different cloth systems). Table 1 gives hyperparameters for all examples in this work.

The hyperparameter σ should be chosen based on the desired behavior of the subspace. Large values yield a subspace which spans extreme states, while smaller values concentrate the subspace tightly around low-energy configurations. The diversity of the subspace also affects the ideal neural network size. Subspaces with large σ which span a larger kinematic range may also require a larger network to accurately resolve the subspace, whereas smaller networks may be sufficient for a subspace with small σ that only represents a narrow range of motions.

Note also that σ depends on the physical units in which the configuration is measured. We recommend initially choosing a large value for σ and visualizing randomly-sampled system configurations during training, recalling that Section 4.1 linearly grows the subspace diversity as training proceeds. For example, if the subspace spans a suitable range of configurations 1/3 of the way through training, then $\sigma \longleftarrow 1/3\sigma$ is a reasonable choice of parameter, and training can be repeated with this value.

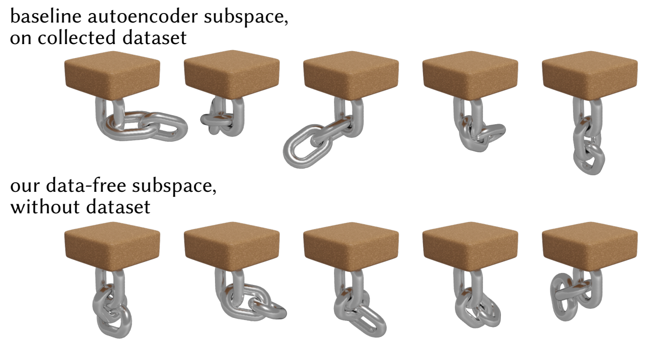
baseline autoencoder subspace, on collected dataset
our data-free subspace, without dataset

Figure 11: A comparison of samples from the latent space of an autoencoder trained on a manually-collected dataset (*top row*), and our data-free approach on the same system (*bottom row*). Both use the same latent dimension ($d = 4$).

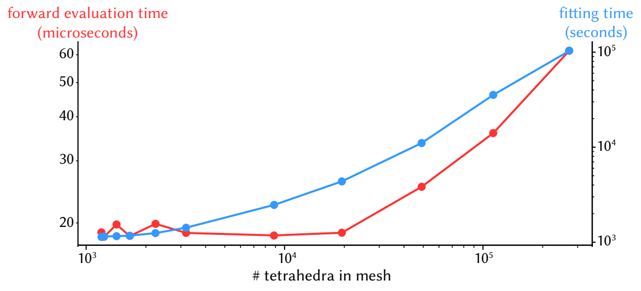

Figure 12: We evaluate the performance scaling of our method, fitting elastic deformation subspaces to the same object tetrahedralized at various resolutions. Each data point is a fitted subspace at a different mesh resolution; the left red axis gives the time for a single forward evaluation of the subspace map, while the right blue axis gives the fitting time.

The hyperparameter λ weights the approximately-isometric objective; it should be chosen ensure Equation 3 has an effect, but also does not dominate the objective and enforce a restrictive affine subspace (see Section B). This is easily assessed by measuring the unitless ratio in Equation 2 during training, if is far from 1 then λ should be increased, and if it very close to 1 (*e.g.* within 10^{-3}) then λ should be reduced.

C.2 Data-Free vs. Supervised Methods

The primary advantage of our formulation is that it fits a subspace using only potential energy function for a system, and does not require a training dataset. Nonetheless, it is useful to consider how the quality of the subspaces compares with a baseline supervised method if a dataset were available. To that end, we gather a dataset by interactively simulating the chain from Figure 6, here with fewer links to facilitate real-time robust full simulation. The resulting dataset contains 40k sampled states of the system. Our method is used to fit a subspace with a 5×128 MLP from a $d = 4$ latent space, which does not require the dataset. As a simple baseline model, we train an autoencoder, where the decoder is an MLP identical to our subspace model, and the encoder is a matching 5×128 hidden layer MLP. The autoencoder is fit via reconstruction loss, along with a weak regularizer to encourage a 0-centered latent space. Figure 11 shows samples from the resulting spaces.

C.3 Performance Scaling

To measure the performance scaling of our method, we fit a series of subspaces to an elastic deformation system where the shape from Figure 7 is discretized at various mesh resolutions ranging from $\approx$ 1k to $\approx$ 150k degrees of freedom. Figure 12 gives the corresponding time cost, measured on the same setup as in Section 4.2. Our method scales well to larger mesh sizes, especially for forward evaluation.

We naively use the exact same training scheme from Section 4.2 here and throughout this work. Likewise, all problem scales use the same 5×128 MLP model, increasing only the output dimension of the last layer to match the degrees of freedom for the system. In practice one might adjust model sizes and training schedules for problems with vastly ranging orders of magnitude to tune performance. Furthermore, approaches such as adaptive cubature [An et al. 2008] are well-suited to accelerate potential energy evaluation for high-resolution deformable models, which could greatly accelerate the fitting procedure.

C.4 Experiment Details

Sampling for [Fulton et al. 2019]. Section 5.2 shows a preliminary application where our data-free subspace is used to sample data for an existing downstream supervised approach, sidestepping the need for dataset collection. In particular, we automatically generate training data for the AutoDef method [Fulton et al. 2019], a recent approach which offers fast deformable simulations but requires significant effort to collect training data. To do so, we first fit our subspace as usual to a single elephant mesh from the AutoDef experiment set, using the training parameters listed in Table 1. Then, we randomly sample 1000 simulation states by taking random sinusoidal motions in latent space, and applying our fitted subspace map $q \leftarrow f_\theta(z)$ to get the corresponding system configurations. The states are encoded as displacements from the rest pose as expected by the AutoDef formulation. These displacements are then used in-place of a manually collected training dataset to fit AutoDef as described in Fulton et al. [2019]. The original AutoDef work proposes a nontrivial pipeline of user interaction to generate training data; we find that in this initial experiment substituting our automatically-sampled unsupervised population yields comparable results without the need to manually collect data.