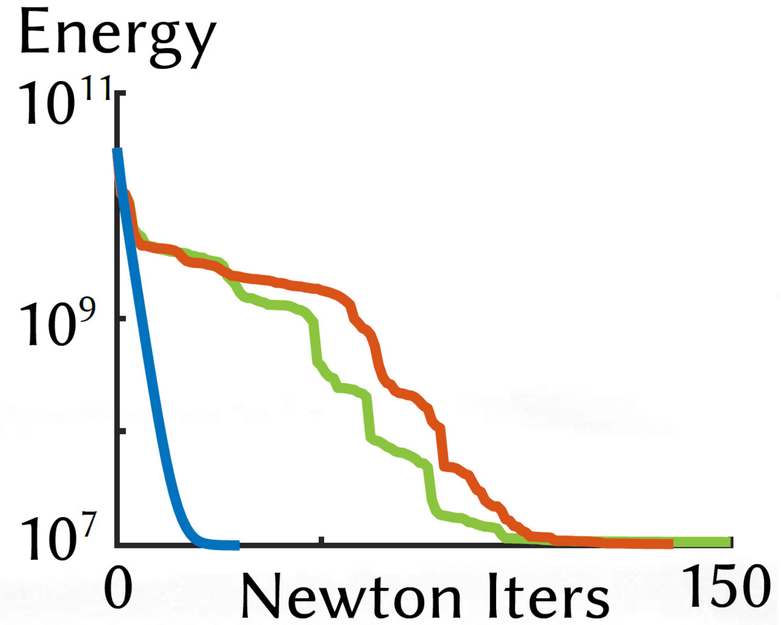
SCA 2024

Introduction to Optimization for Simulation

Honglin Chen, Columbia University

COLUMBIA UNIVERSITY



Quick Intro

- Hi everyone! I'm Honglin Chen.
- Rising 4th year PhD student @ Columbia
  - Physics-based Simulation
  - Geometry Processing
  - Optimization and (a little bit of) ML
- My 2nd SCA
  - But my 1st time presenting here!

What's optimization (and why should we study it) ?

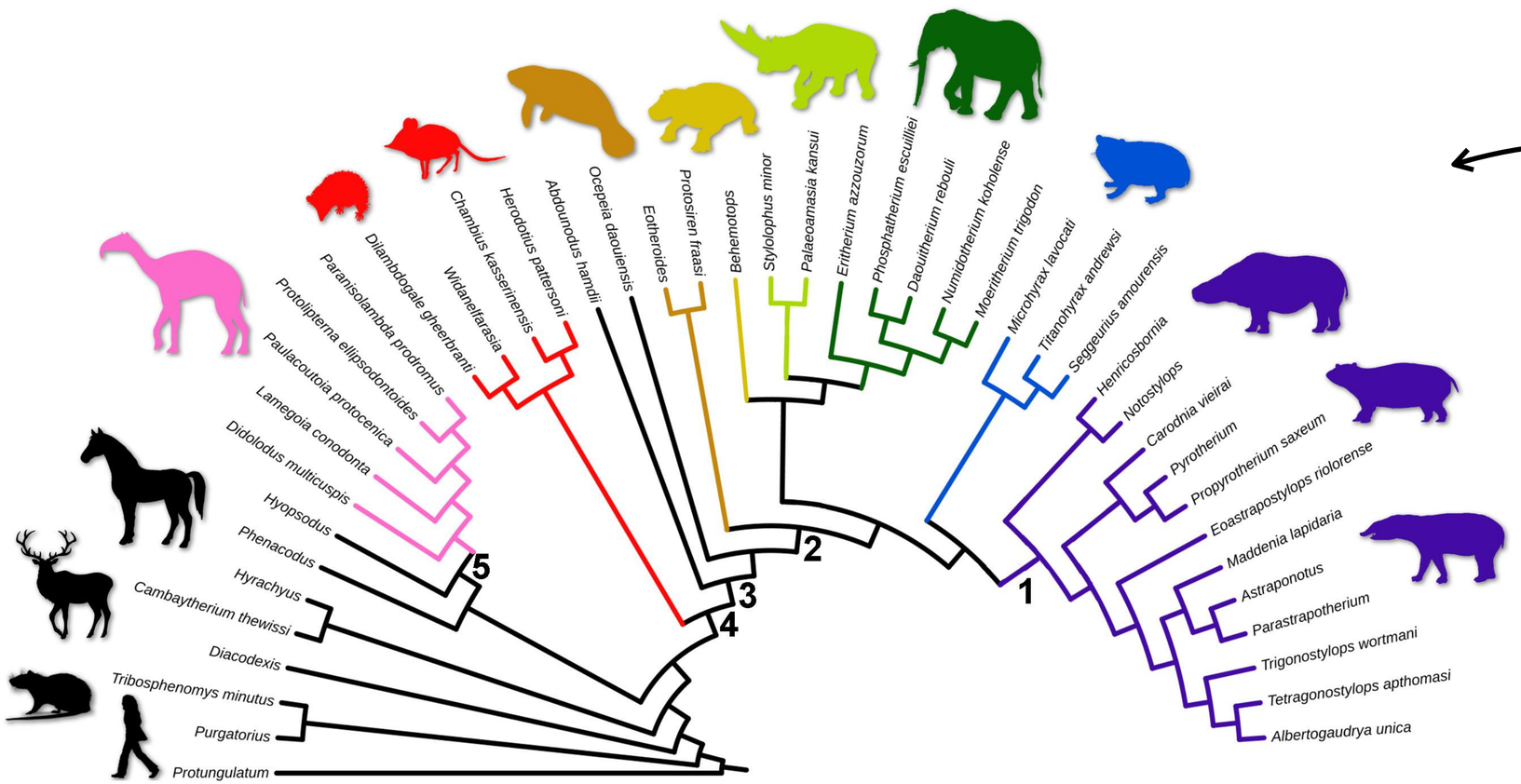
Finding the best feasible solution from all the possibilities

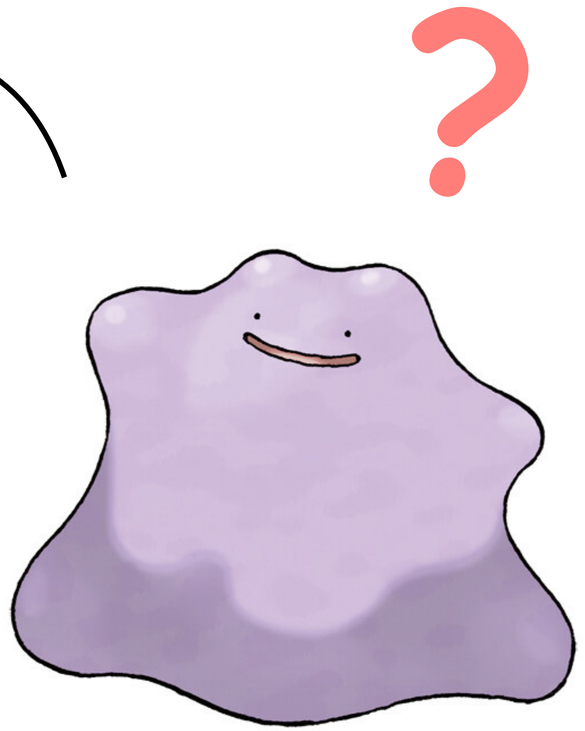
(potentially subject to some constraints)

Inspiration from here

What's an optimization problem?

Ingredient 1: optimization variables





Inspiration from here

What's an optimization problem?

Ingredient 2: objective function

Optimized for running

Optimized for flying

Inspiration from here

What's an optimization problem?

Ingredient 2: objective function

Optimized for swimming

Optimized for cuteness

Inspiration from here

What's an optimization problem?

Ingredient 2: objective function

What is this optimized for ?? 🤔

Inspiration from here

What's an optimization problem?

3 components:

- **optimization variables that parameterize the space**
- **an objective function that measures how "good" an arbitrary point in parameter space is**
- **possibly some constraints**

Inspiration from here

Why is optimization so important in graphics?

Optimization in Simulation and Geometry

And many more…

Mesh Parameterization

Elastodynamic Simulation

Who is this course for?

- New to optimization
- New to elastic simulation
- Want to learn about different techniques for elastic energy minimization

What will I learn from this course?

- How to minimize an elastic energy (defined on a mesh)?
  - Classical techniques
  - More advanced optimization techniques
- How to make your optimization run better?
  - Common challenges
  - Strategies to tackle these challenges
  - Know where to look if you want to learn more
- Tips for easy implementations
  - Tools and frameworks

Non-Goals of this Course

This course will **not**

- Cover how to formulate these energies
  - Eris will talk about that in our next course soon!
- Teach you how to minimize an energy on a neural network
- Dive into details on designing a customized solver for a specific problem

Agenda

- Elastic Energy
- Optimization Basics & Classical Algorithms
- Optimization Methods for Simulation
  - ✨Challenge 1✨: Handling Nonconvexity
  - ✨Challenge 2✨: Nontrivial Constraints
  - ✨Challenge 3✨: Large-scale Optimization

- **Elastic Energy**
- Optimization Basics & Classical Algorithms
- Optimization Methods for Simulation
  - ✨Challenge 1✨: Handling Nonconvexity
  - ✨Challenge 2✨: Nontrivial Constraints
  - ✨Challenge 3✨: Large-scale Optimization

Elastic Energy

Elastic Energy is Everywhere

Cloth

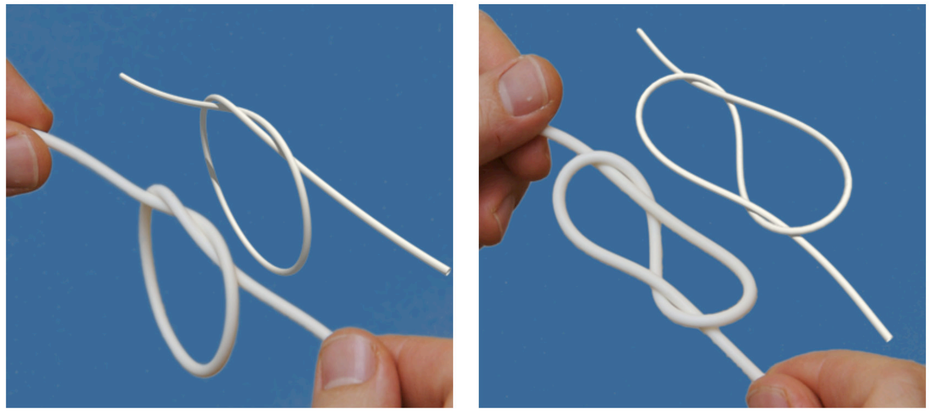
[Zhang et al. 2022]

3D Deformable Solids

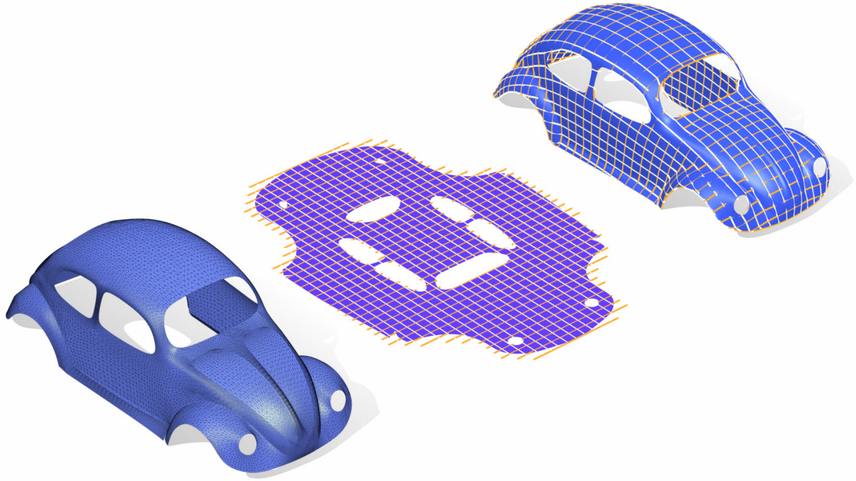
[Smith et al. 2018]

Thin Shells

[Grinspun et al. 2003]



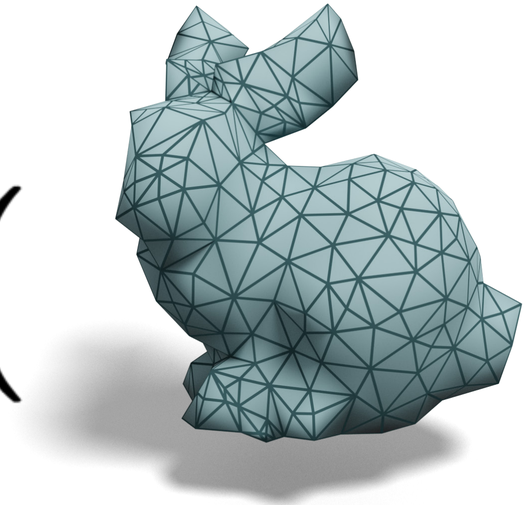
Rods [Bergou et al. 2008]

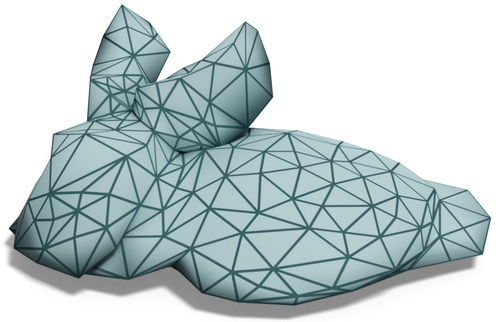


Surface Parameterization [Smith et al. 2018]

Elastic Energy

$$E_{\text{elastic}}\left(\quad , \quad \right)$$



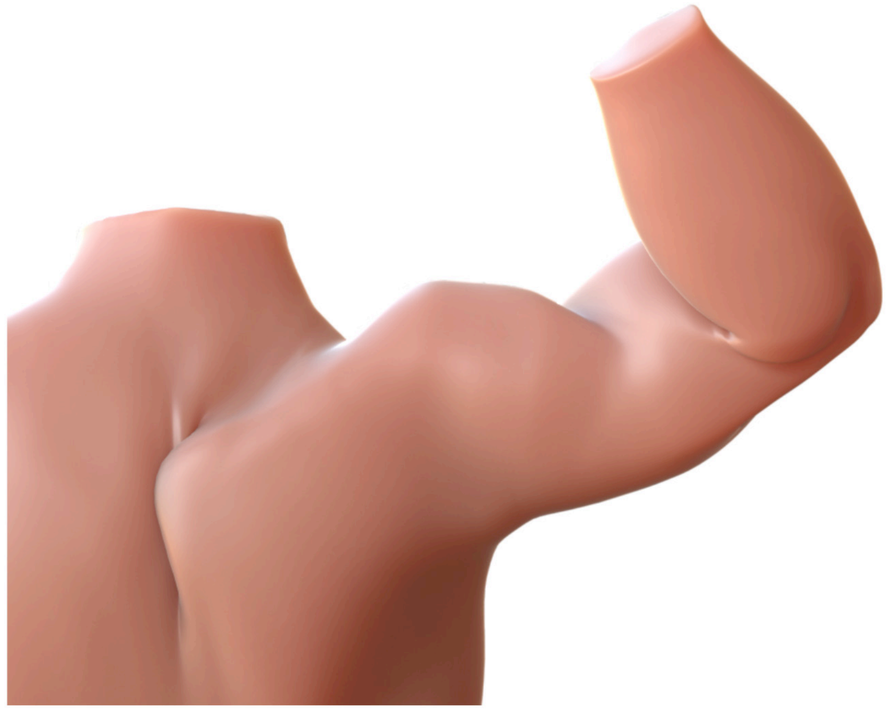


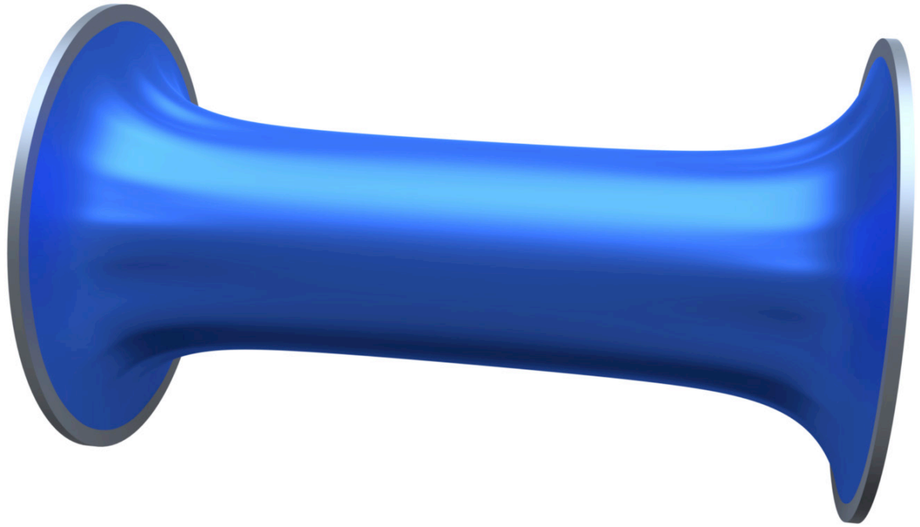
Target Shape

Current Shape

Quasistatic Simulation

$$E_{\text{elastic}}\left(\quad , \quad \right)$$

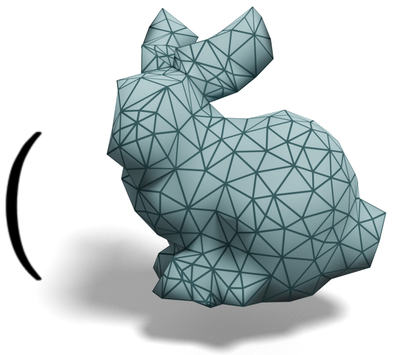


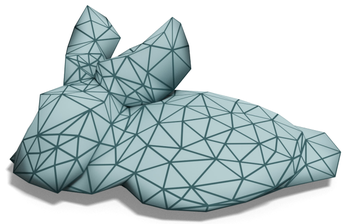


Stable Neo-Hookean Flesh Simulation.
Breannan Smith, Fernando de Goes, and Theodore Kim. ACM Trans. Graph. 2018.

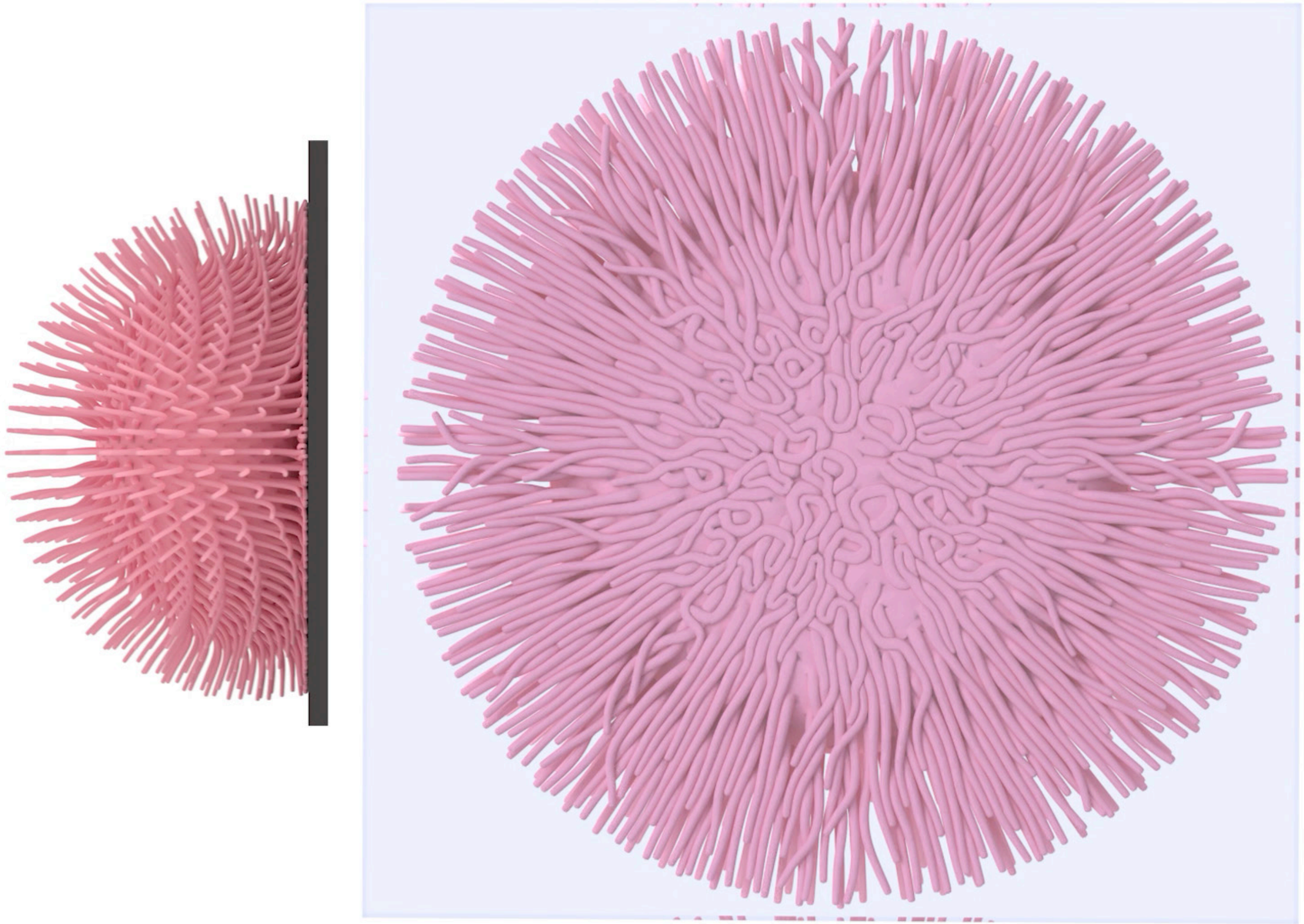
Dynamic Simulation

$$E_{\text{elastic}}\left(\quad , \quad \right) + E_{\text{kinetic}} + E_{\text{external\_force}} + E_{\text{collision}}$$



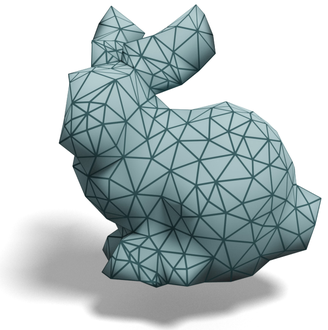
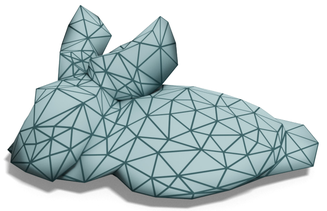


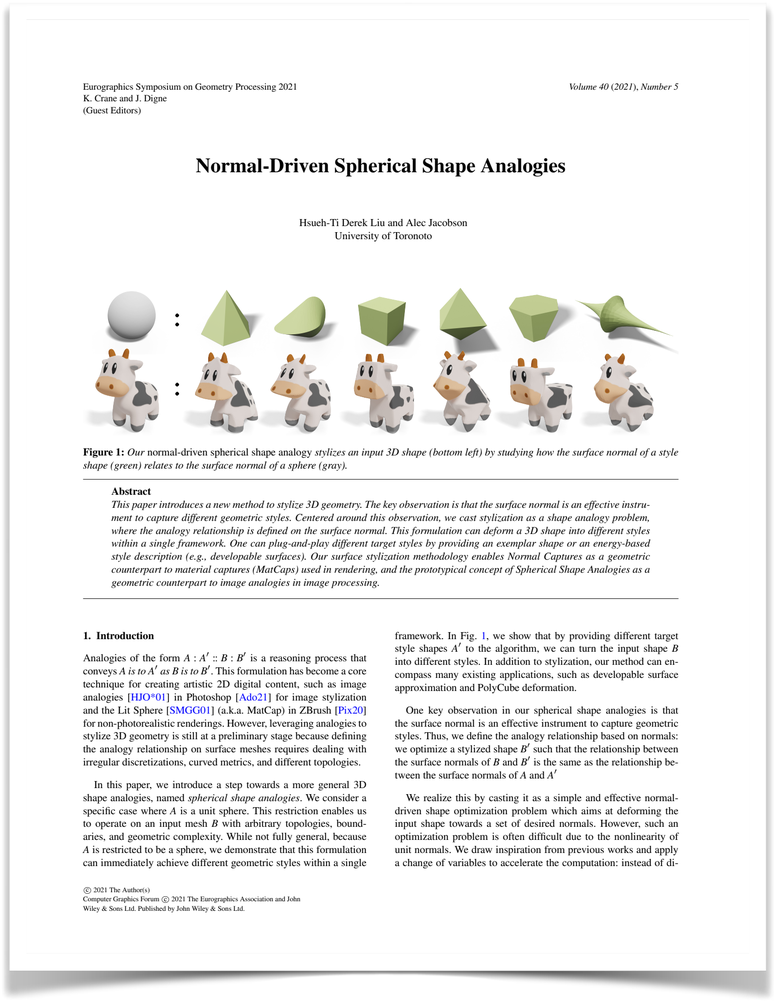
Incremental Potential Contact: Intersection- and Inversion-free Large Deformation Dynamics.
Minchen Li, Zachary Ferguson, Teseo Schneider, Timothy Langlois, Denis Zorin, Daniele Panozzo, Chenfanfu Jiang, Danny M. Kaufman.
ACM Trans. Graph. 2020.



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

Elastic Energy as Regularization

$$E_{\text{problem\_specific}} + E_{\text{elastic}}\left(\text{  }, \text{  } \right)$$



Eurographics Symposium on Geometry Processing 2021
K. Crane and J. Digne
(Guest Editors)

Volume 40 (2021), Number 5

Normal-Driven Spherical Shape Analogies

Hsueh-Ti Derek Liu and Alec Jacobson
University of Toronoto

Figure 1: *Our* normal-driven spherical shape analogy *stylizes an input 3D shape (bottom left) by studying how the surface normal of a style shape (green) relates to the surface normal of a sphere (gray).*

Abstract

This paper introduces a new method to stylize 3D geometry. The key observation is that the surface normal is an effective instrument to capture different geometric styles. Centered around this observation, we cast stylization as a shape analogy problem, where the analogy relationship is defined on the surface normal. This formulation can deform a 3D shape into different styles within a single framework. One can plug-and-play different target styles by providing an exemplar shape or an energy-based style description (e.g., developable surfaces). Our surface stylization methodology enables Normal Captures as a geometric counterpart to material captures (MatCaps) used in rendering, and the prototypical concept of Spherical Shape Analogies as a geometric counterpart to image analogies in image processing.

1. Introduction

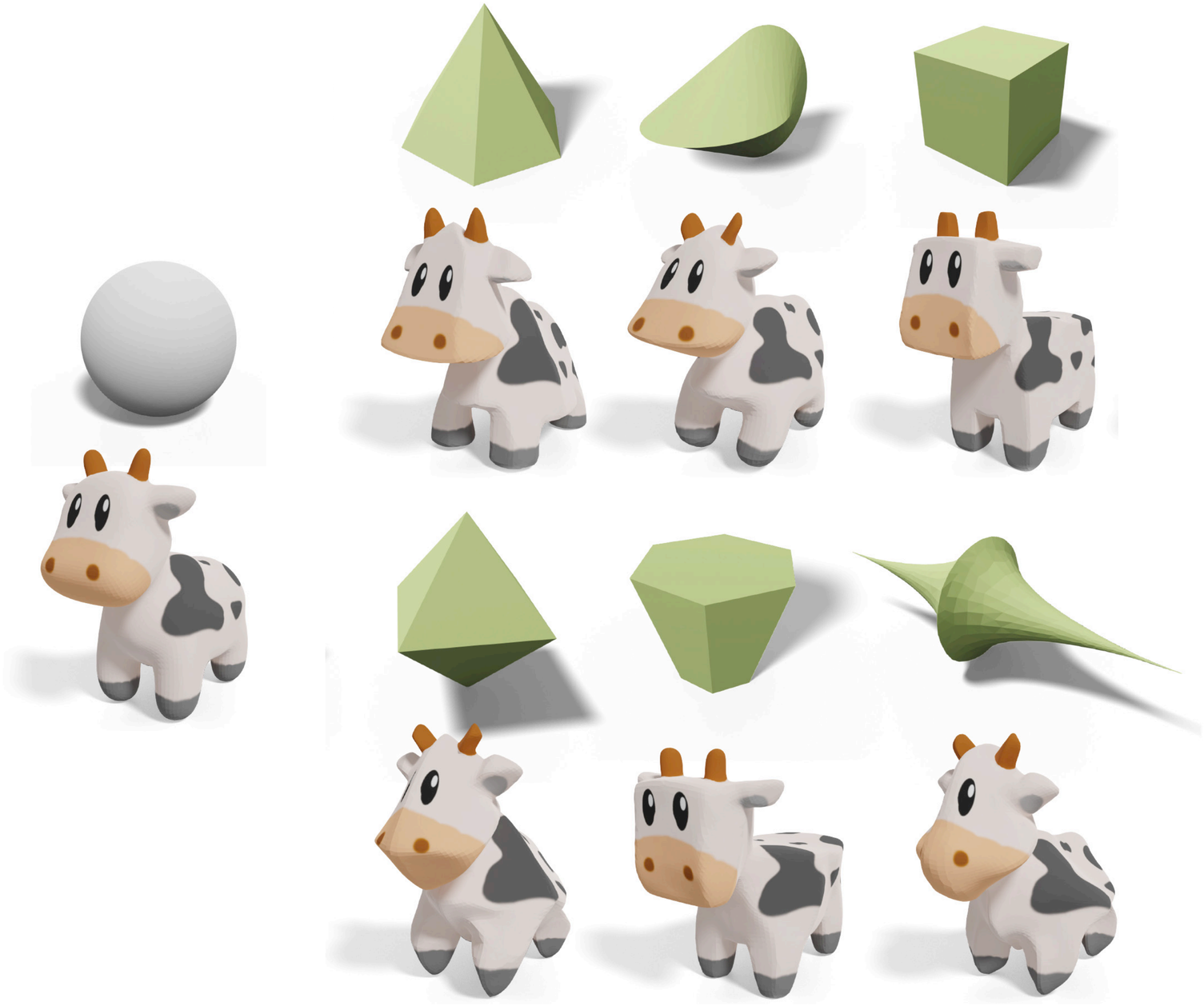
Analogies of the form $A : A' :: B : B'$ is a reasoning process that conveys *A is to A′ as B is to B′*. This formulation has become a core technique for creating artistic 2D digital content, such as image analogies [HJO*01] in Photoshop [Ado21] for image stylization and the Lit Sphere [SMGG01] (a.k.a. MatCap) in ZBrush [Pix20] for non-photorealistic renderings. However, leveraging analogies to stylize 3D geometry is still at a preliminary stage because defining the analogy relationship on surface meshes requires dealing with irregular discretizations, curved metrics, and different topologies.

In this paper, we introduce a step towards a more general 3D shape analogies, named *spherical shape analogies*. We consider a specific case where A is a unit sphere. This restriction enables us to operate on an input mesh B with arbitrary topologies, boundaries, and geometric complexity. While not fully general, because A is restricted to be a sphere, we demonstrate that this formulation can immediately achieve different geometric styles within a single framework. In Fig. 1, we show that by providing different target style shapes A' to the algorithm, we can turn the input shape B into different styles. In addition to stylization, our method can encompass many existing applications, such as developable surface approximation and PolyCube deformation.

One key observation in our spherical shape analogies is that the surface normal is an effective instrument to capture geometric styles. Thus, we define the analogy relationship based on normals: we optimize a stylized shape B' such that the relationship between the surface normals of B and B' is the same as the relationship between the surface normals of A and A'

We realize this by casting it as a simple and effective normal-driven shape optimization problem which aims at deforming the input shape towards a set of desired normals. However, such an optimization problem is often difficult due to the nonlinearity of unit normals. We draw inspiration from previous works and apply a change of variables to accelerate the computation: instead of di-

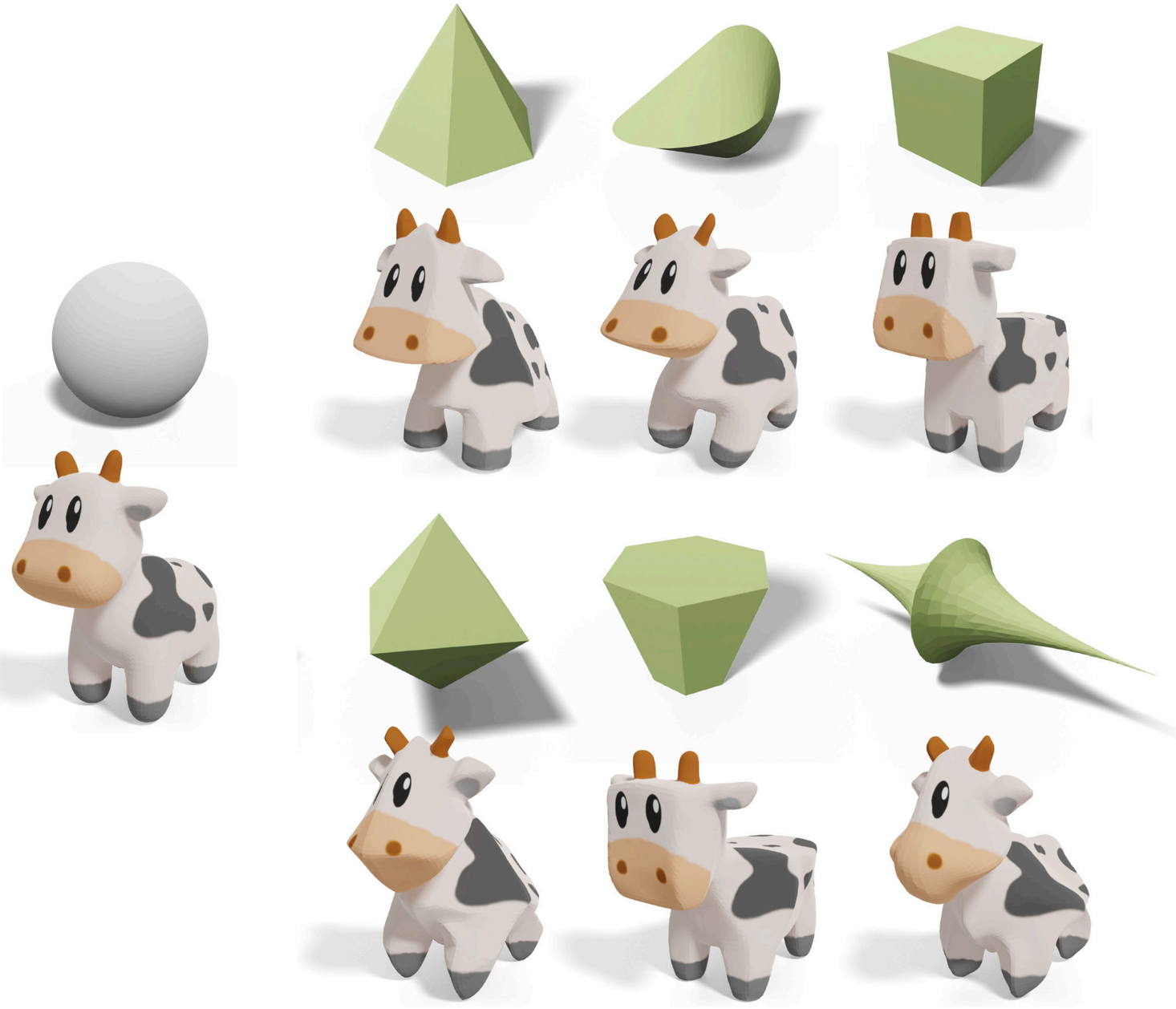
Shape Stylization



$$E_{\text{style}}$$

$$+$$

$$E_{\text{elastic}}$$



Shape Stylization

Local Deformation for Interactive Shape Editing

Honglin Chen
Columbia University
New York City, NY, USA
honglin.chen@columbia.edu

Changxi Zheng
Columbia University
New York City, NY, USA
cxz@cs.columbia.edu

Kevin Wampler
Adobe Research
Seattle, WA, USA
kwampler@adobe.com

Undeformed

Interactive edit with local deformation

Figure 1: Our method enables the user to edit shapes in an interactive and physically plausible way. The edit is *local*, meaning that the user can focus on one region of the complex scene without worrying about inadvertent changes elsewhere. To visualize the locality, in the rightmost figure we highlight the regions where the vertex displacement is larger than 10^{-3} in red. (Undeformed scene shapes thanks to [Zhang et al. 2022])

ABSTRACT

We introduce a novel regularization for localizing an elastic-energy-driven deformation to only those regions being manipulated by the user. Our local deformation features a natural region of influence, which is automatically adaptive to the geometry of the shape, the size of the deformation and the elastic energy in use. We further propose a three-block ADMM-based optimization to efficiently minimize the energy and achieve interactive frame rates. Our approach avoids the artifacts of other alternative methods, is simple and easy to implement, does not require tedious control primitive setup and generalizes across different dimensions and elastic energies. We demonstrates the effectiveness and efficiency of our localized deformation tool through a variety of local editing scenarios, including 1D, 2D, 3D elasticity and cloth deformation.

CCS CONCEPTS

• **Computing methodologies** → *Mesh geometry models.*

KEYWORDS

Local control, shape deformation, elasticity, sparsity, ADMM.

SIGGRAPH '23 Conference Proceedings, August 6–10, 2023, Los Angeles, CA, USA

ACM ISBN 979-8-4007-0159-7/23/08…$15.00
https://doi.org/10.1145/3588432.3591501

ACM Reference Format:
Honglin Chen, Changxi Zheng, and Kevin Wampler. 2023. Local Deformation for Interactive Shape Editing. In *Special Interest Group on Computer Graphics and Interactive Techniques Conference Conference Proceedings (SIGGRAPH '23 Conference Proceedings), August 6–10, 2023, Los Angeles, CA, USA.* ACM, New York, NY, USA, 15 pages. https://doi.org/10.1145/3588432.3591501

1 INTRODUCTION

Local deformation is a core component in modeling and animation. In a localized deformation, only the parts of the shape near where the user is currently manipulating move—everything else stays still, ensuring that the user can focus entirely on one region of the shape without worrying about inadvertent changes elsewhere. However, existing localized deformation tools tend to have practical impediments for interactive design: they are either too slow to run, unaware of the geometry, introduce artifacts, or require a careful control point setup.

When global deformation is acceptable, a widely useful approach is to solve for the deformation by minimizing an elastic energy defined over the shape, subject to positional constraints derived from the user's input. This paradigm has many advantages. The deformation accounts for the geometry of the shape, generalizes well to 2D, 3D, and cloth, and the elastic energy can be used to model a wide range of both real-world and stylized materials. Unfortunately elastic energy minimization is by its nature global, and jointly solves for all the degrees of freedom in the shape. This necessitates a rigging step to "pin down" certain aspects of the deformation lest the optimizer move them. This limits the applicability of such methods to situations where a suitable rig is available, or the region of influence for a deformation is known in advance.

$$E_{\text{local}}$$

$$+$$

$$E_{\text{elastic}}$$

Local Deformation

Large Steps in Inverse Rendering of Geometry

BAPTISTE NICOLET, École Polytechnique Fédérale de Lausanne (EPFL), Switzerland
ALEC JACOBSON, University of Toronto, Canada
WENZEL JAKOB, École Polytechnique Fédérale de Lausanne (EPFL), Switzerland

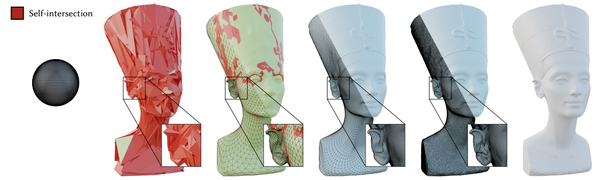


Fig. 1. **(a)** Inverse reconstruction of the Nefertiti bust from a spherical starting guess with 25 rendered views (1 shown). **(b)** Naïve application of a differentiable renderer produces an unusable tangled mesh when gradient steps pull on the silhouette without regard for distortion or self-intersections. **(c)** Regularization can alleviate such problems by making the optimization aware of mesh quality. On the flipside, this penalizes non-smooth parts of the geometry and causes unsatisfactory convergence in gradient-based optimizers. While the final mesh undeniably looks better, a closer inspection of the wireframe rendering reveals countless self-intersections. **(d)** Our method addresses both problems and converges to a high-quality mesh. **(e)** Combined with an isotropic remeshing step, our reconstruction captures fine details of the reference **(f)**. The hyper-parameters of each method were optimized to obtain the best convergence at equal time. Self-intersections are shown in red.

Inverse reconstruction from images is a central problem in many scientific and engineering disciplines. Recent progress on differentiable rendering has led to methods that can efficiently differentiate the full process of image formation with respect to millions of parameters to solve such problems via gradient-based optimization.

At the same time, the availability of cheap derivatives does not necessarily make an inverse problem easy to solve. Mesh-based representations remain a particular source of irritation: an adverse gradient step involving vertex positions could turn parts of the mesh inside-out, introduce numerous local self-intersections, or lead to inadequate usage of the vertex budget due to distortion. These types of issues are often irrecoverable in the sense that subsequent optimization steps will further exacerbate them. In other words, the optimization lacks robustness due to an objective function with substantial non-convexity.

Such robustness issues are commonly mitigated by imposing additional regularization, typically in the form of Laplacian energies that quantify and improve the smoothness of the current iterate. However, regularization introduces its own set of problems: solutions must now compromise between solving the problem and being smooth. Furthermore, gradient steps involving a Laplacian energy resemble Jacobi's iterative method for solving linear equations that is known for its exceptionally slow convergence.

We propose a simple and practical alternative that casts differentiable rendering into the framework of preconditioned gradient descent. Our preconditioner biases gradient steps towards smooth solutions without requiring the final solution to be smooth. In contrast to Jacobi-style iteration, each gradient step propagates information among all variables, enabling convergence using fewer and larger steps.

Our method is not restricted to meshes and can also accelerate the reconstruction of other representations, where smooth solutions are generally expected. We demonstrate its superior performance in the context of geometric optimization and texture reconstruction.

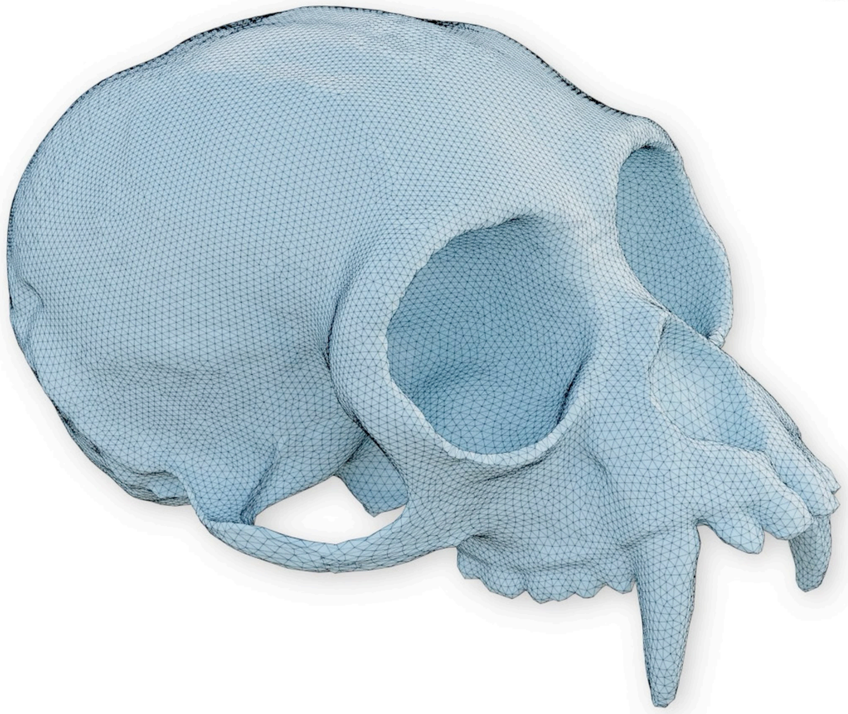
CCS Concepts: • **Computing methodologies → Rendering**; **Shape modeling**.

Additional Key Words and Phrases: differentiable rendering, geometry reconstruction, Laplacian mesh processing

ACM Reference Format:
Baptiste Nicolet, Alec Jacobson, and Wenzel Jakob. 2021. Large Steps in Inverse Rendering of Geometry. *ACM Trans. Graph.* 40, 6, Article 248 (December 2021), 13 pages. https://doi.org/10.1145/3478513.3480501

Authors' addresses: Baptiste Nicolet, École Polytechnique Fédérale de Lausanne (EPFL), Lausanne, Switzerland, baptiste.nicolet@epfl.ch; Alec Jacobson, University of Toronto, Canada, jacobson@cs.toronto.edu, wenzel.jakob@epfl.ch; Wenzel Jakob, École Polytechnique Fédérale de Lausanne (EPFL), Lausanne, Switzerland, wenzel.jakob@epfl.ch.

Target:



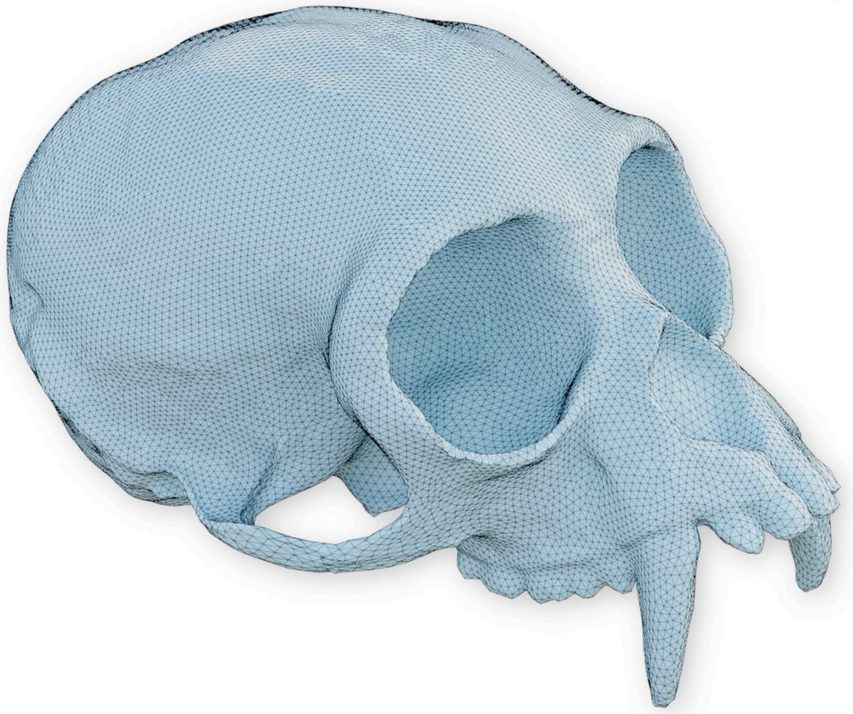
Inverse Rendering

$$E_{\text{render}}$$

$$+$$

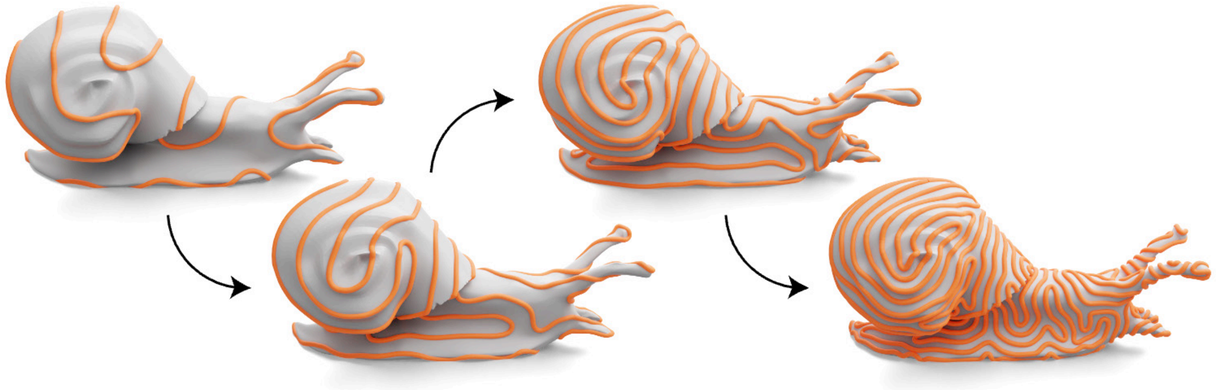
$$E_{\text{elastic}}$$

Target:



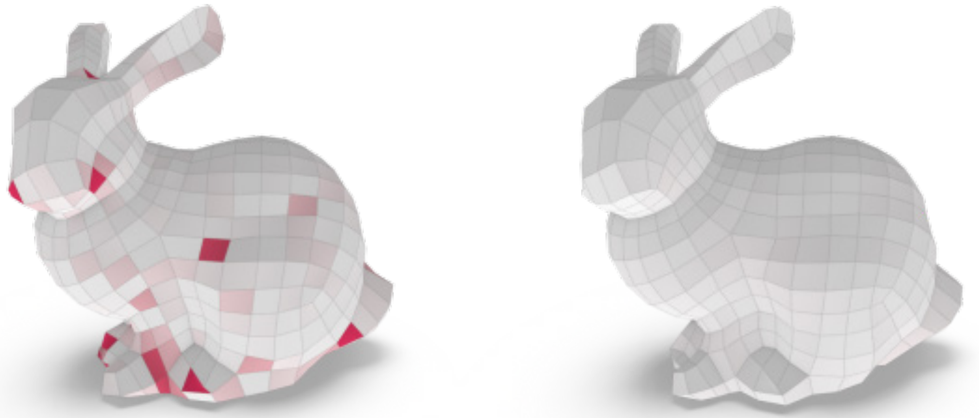
Inverse Rendering

More than just Elastic Energy



Surface-Filling Curves

[Noma et al. 2024]

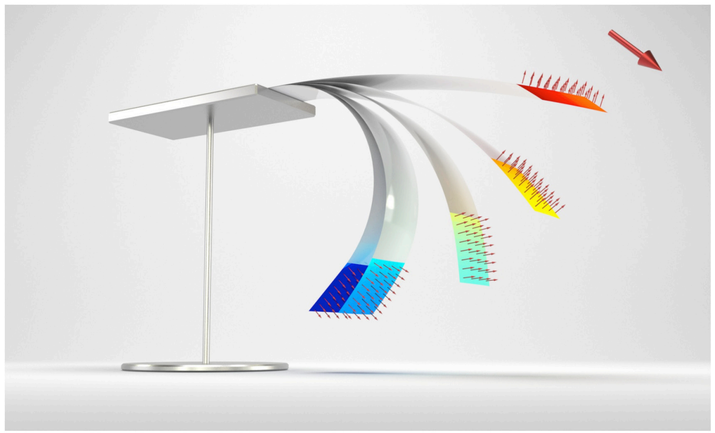


Quad Mesh Planarization

[Liu et al. 2006]

Developable Surfaces

[Sellán et al. 2020]



Magnetic Simulation

[Chen et al. 2022]

And many more...!

- Elastic Energy
- **Optimization Basics & Classical Algorithms**
- Optimization Methods for Simulation
  - ✨Challenge 1✨: Handling Nonconvexity
  - ✨Challenge 2✨: Nontrivial Constraints
  - ✨Challenge 3✨: Large-scale Optimization

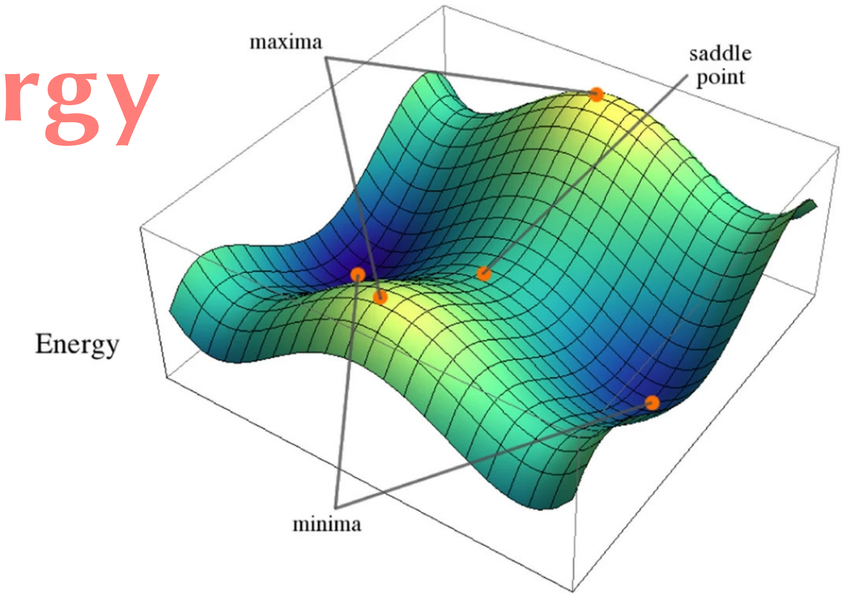
Optimization Basics

What is Energy Minimization?

$$\min_{x} f(x)$$

Optimization Variables

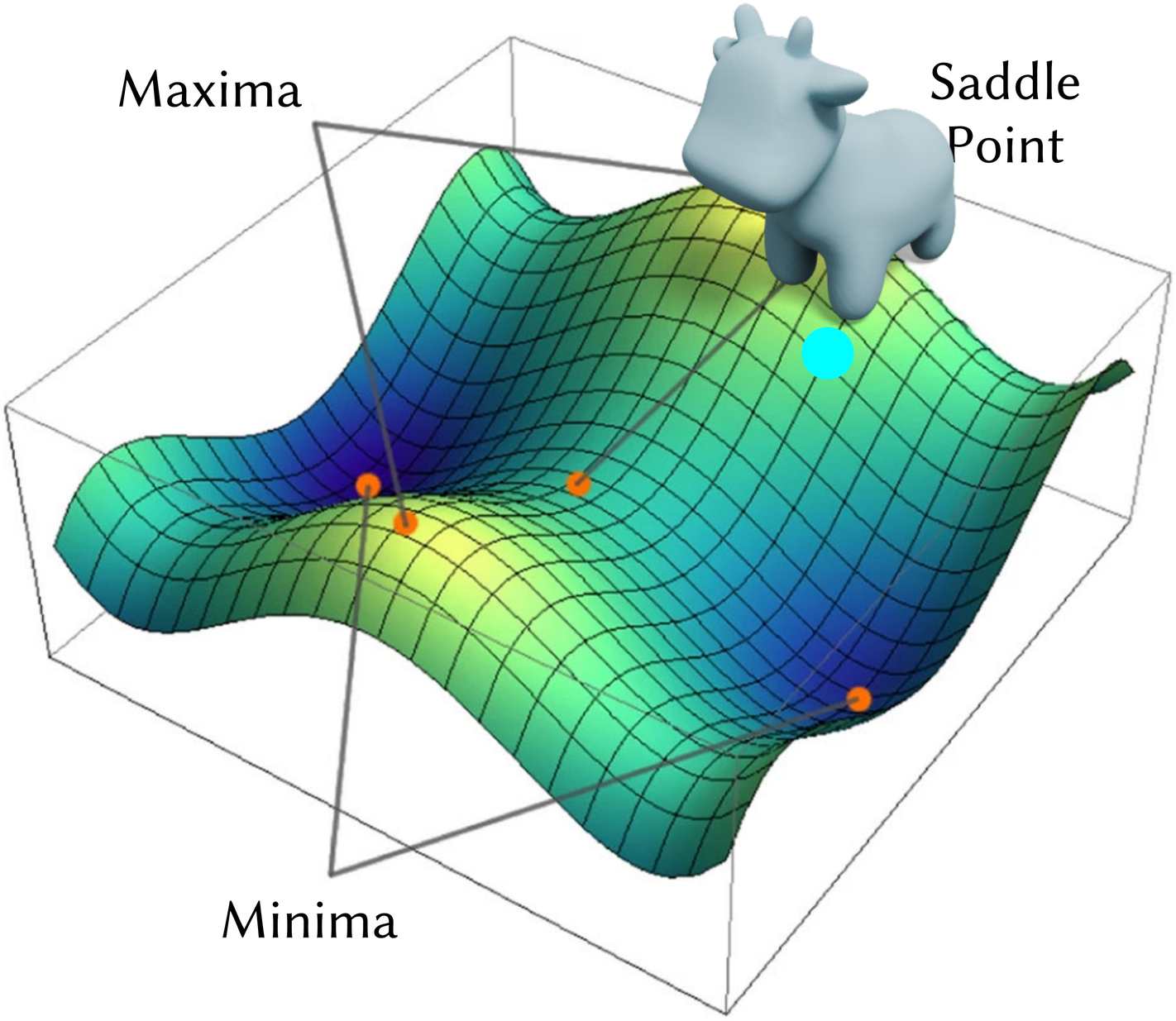
Energy

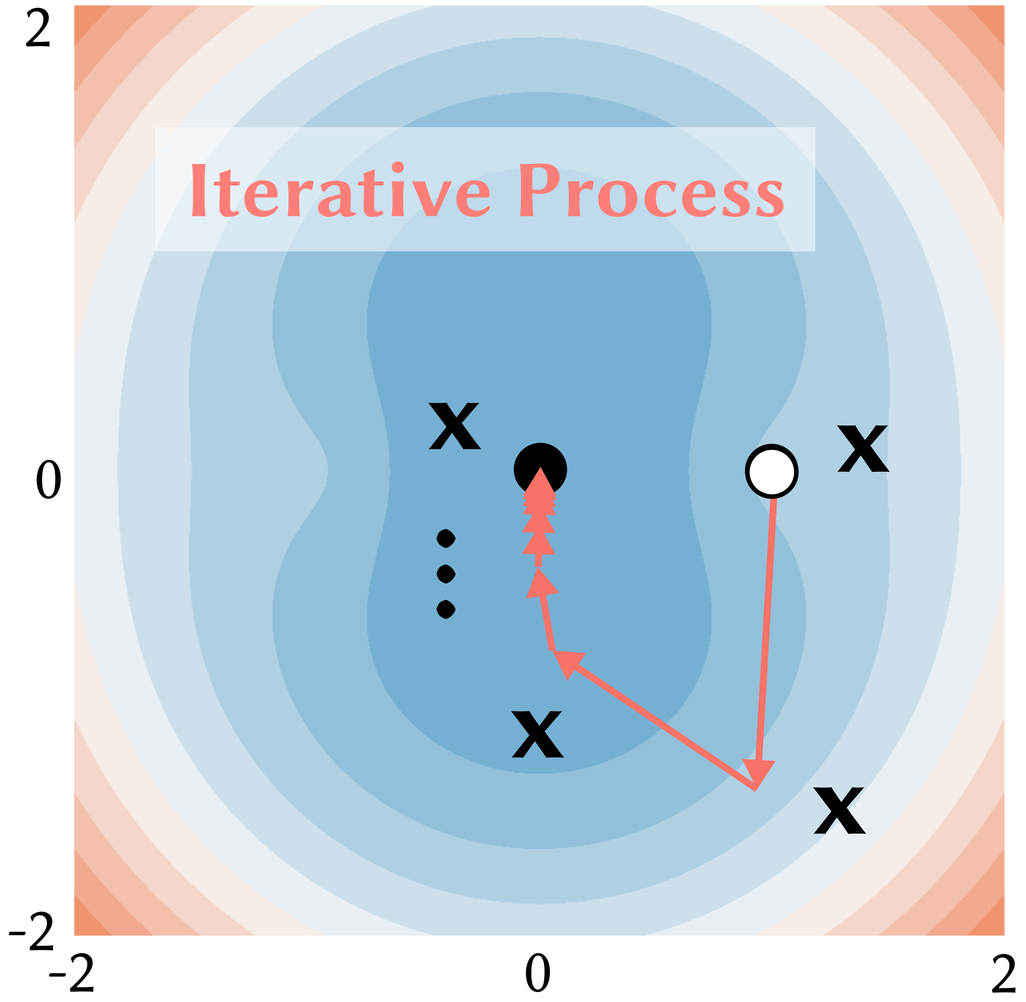


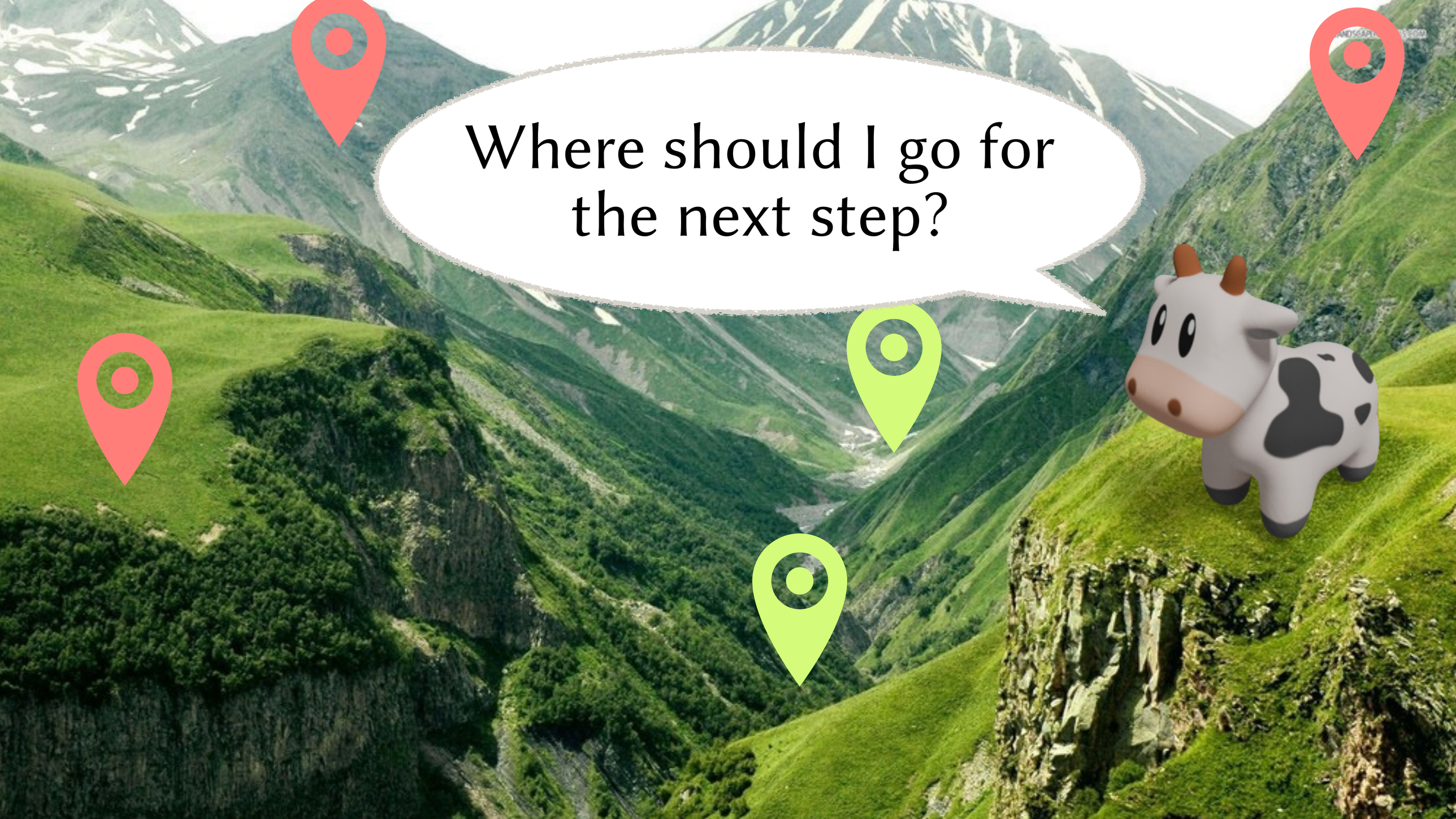

High Energy
Undesired State
Low Energy
Desired State
Spot

How can I go to the lowest point of the energy landscape?

Iterative Optimization





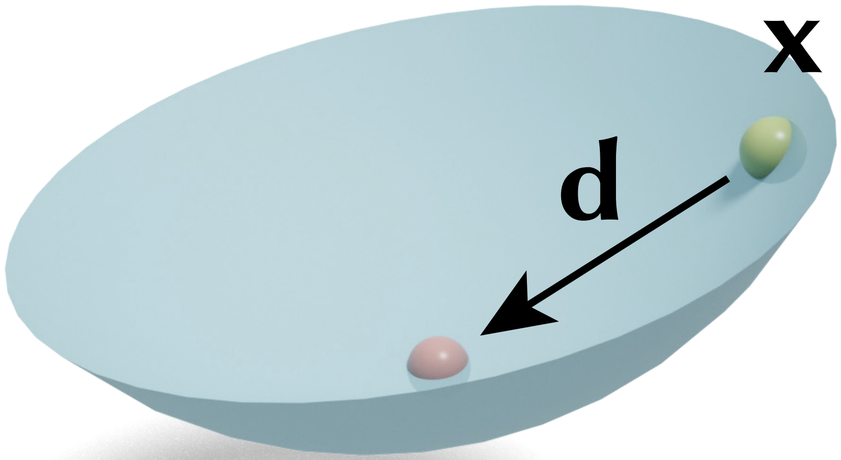

Where should I go for
the next step?

1. Which direction?

2. How far should I go?

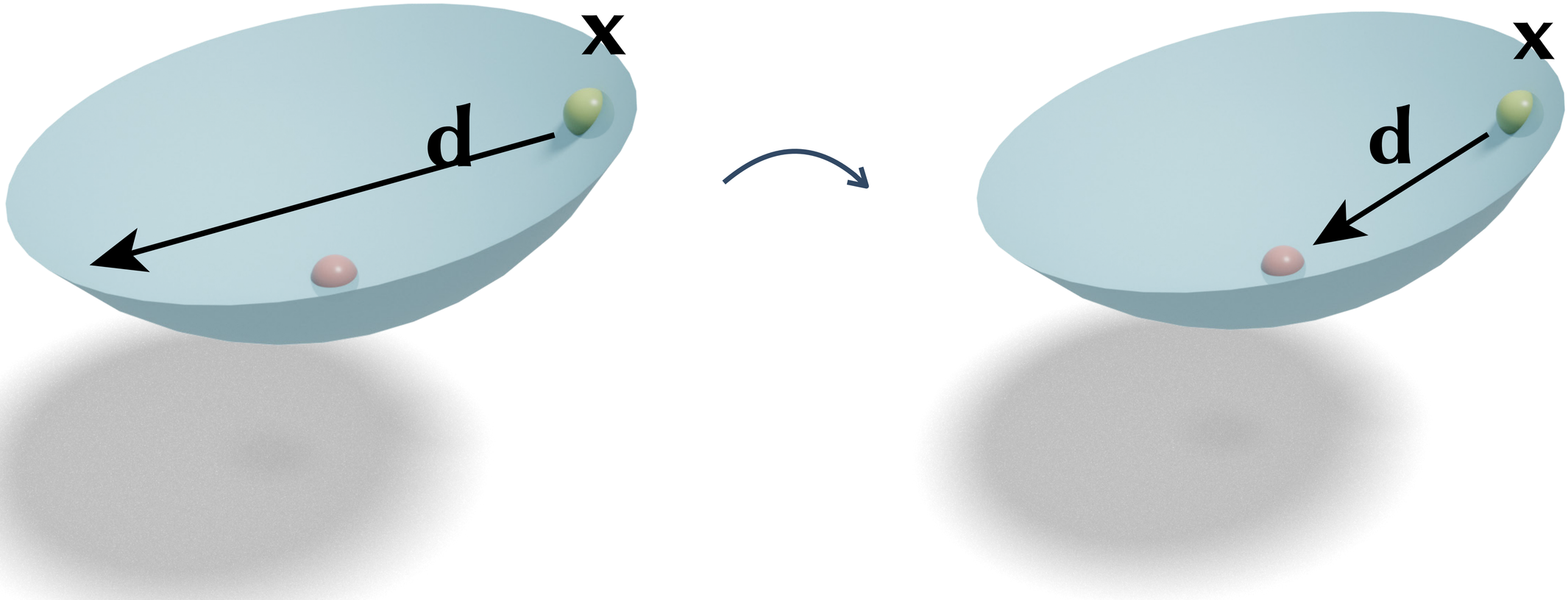
Search Direction

A descent direction is a vector that points towards a local minimum of an objective function.

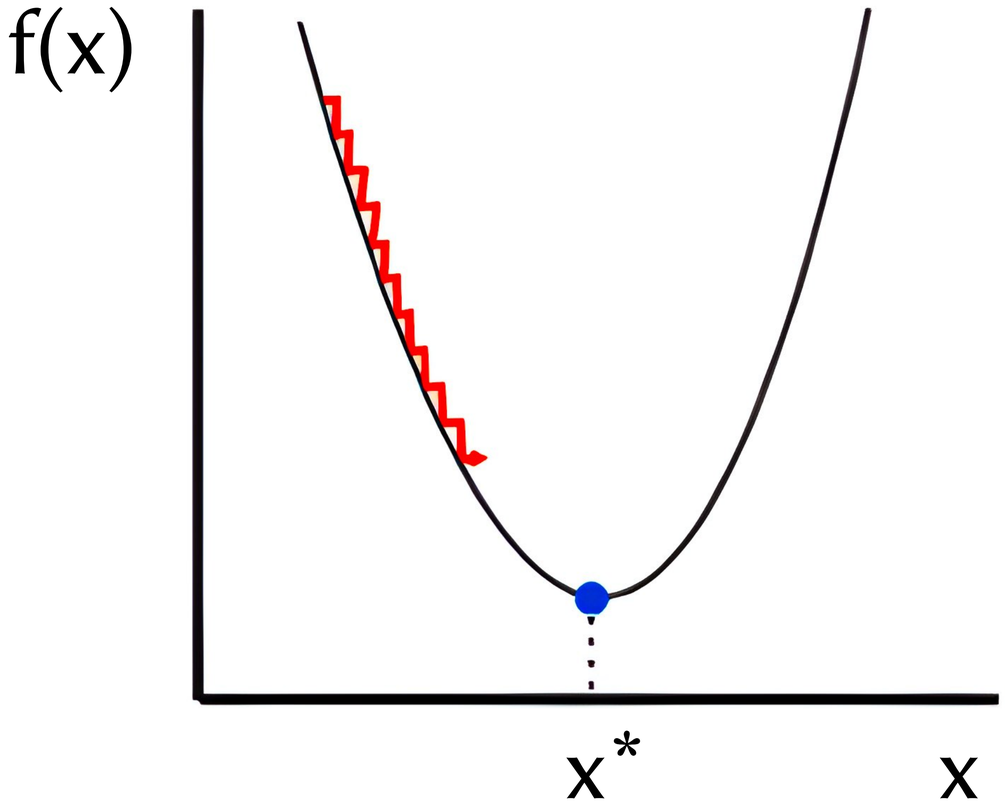


Step Size

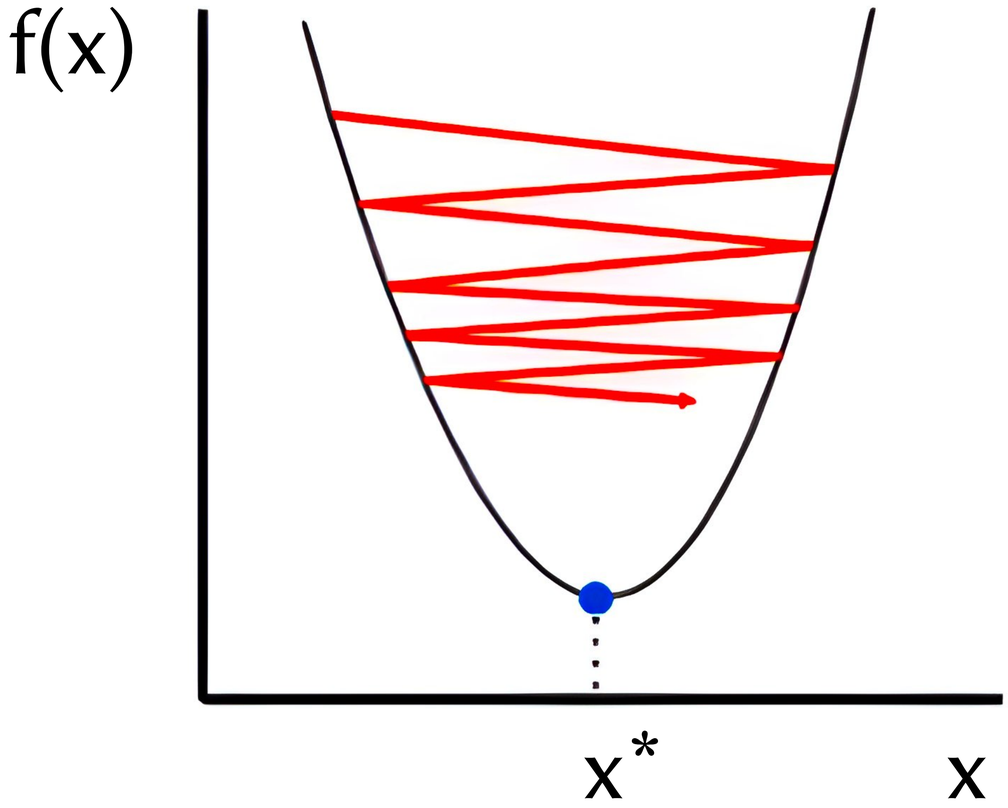
The step size determines how far we should move along the search direction.



Picking the Right Step Size



Too small: converge vert slowly



Too big: overshoot and even diverge

Backtracking Line Search

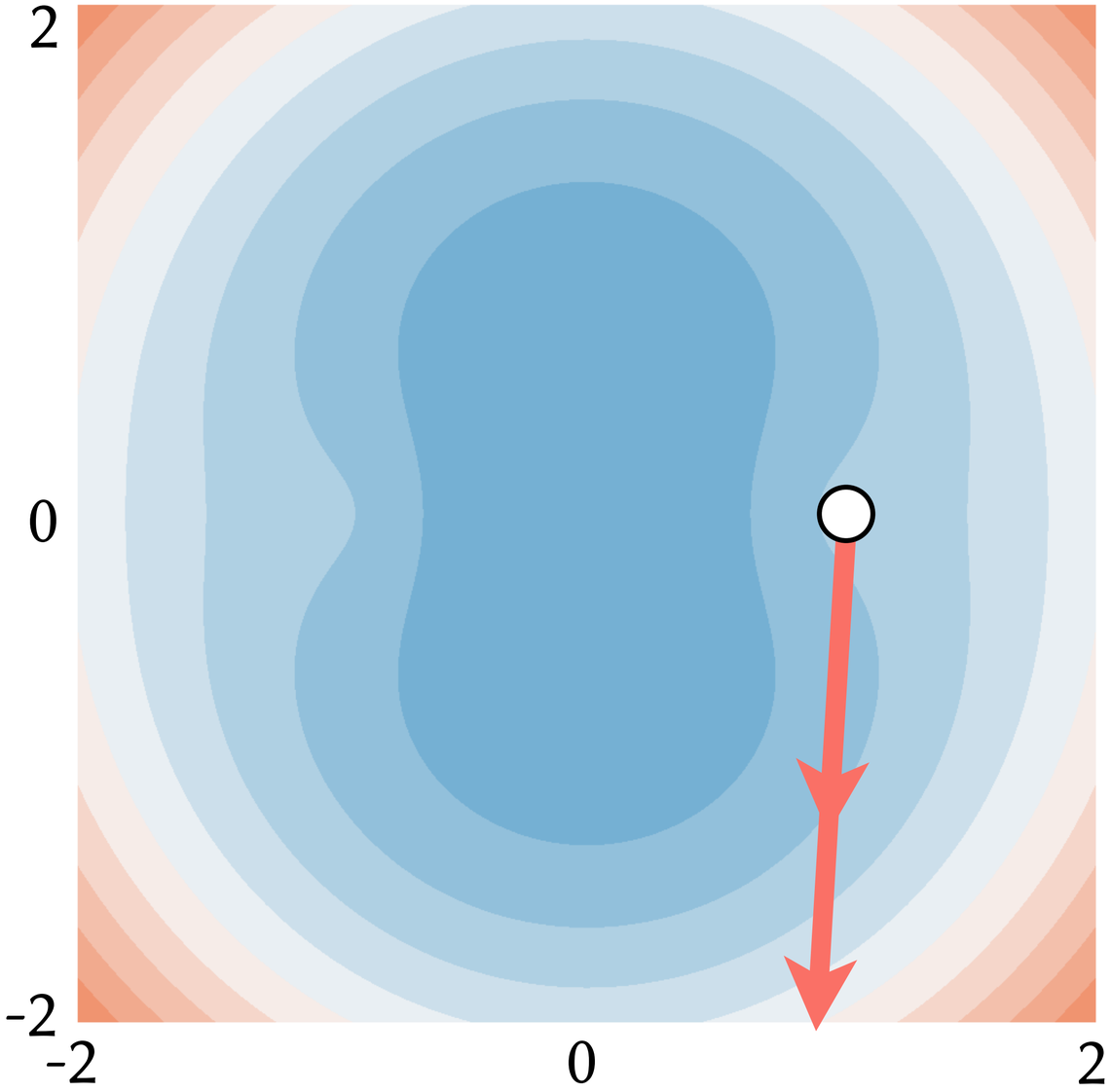
Result: α

$\alpha \leftarrow 1;$

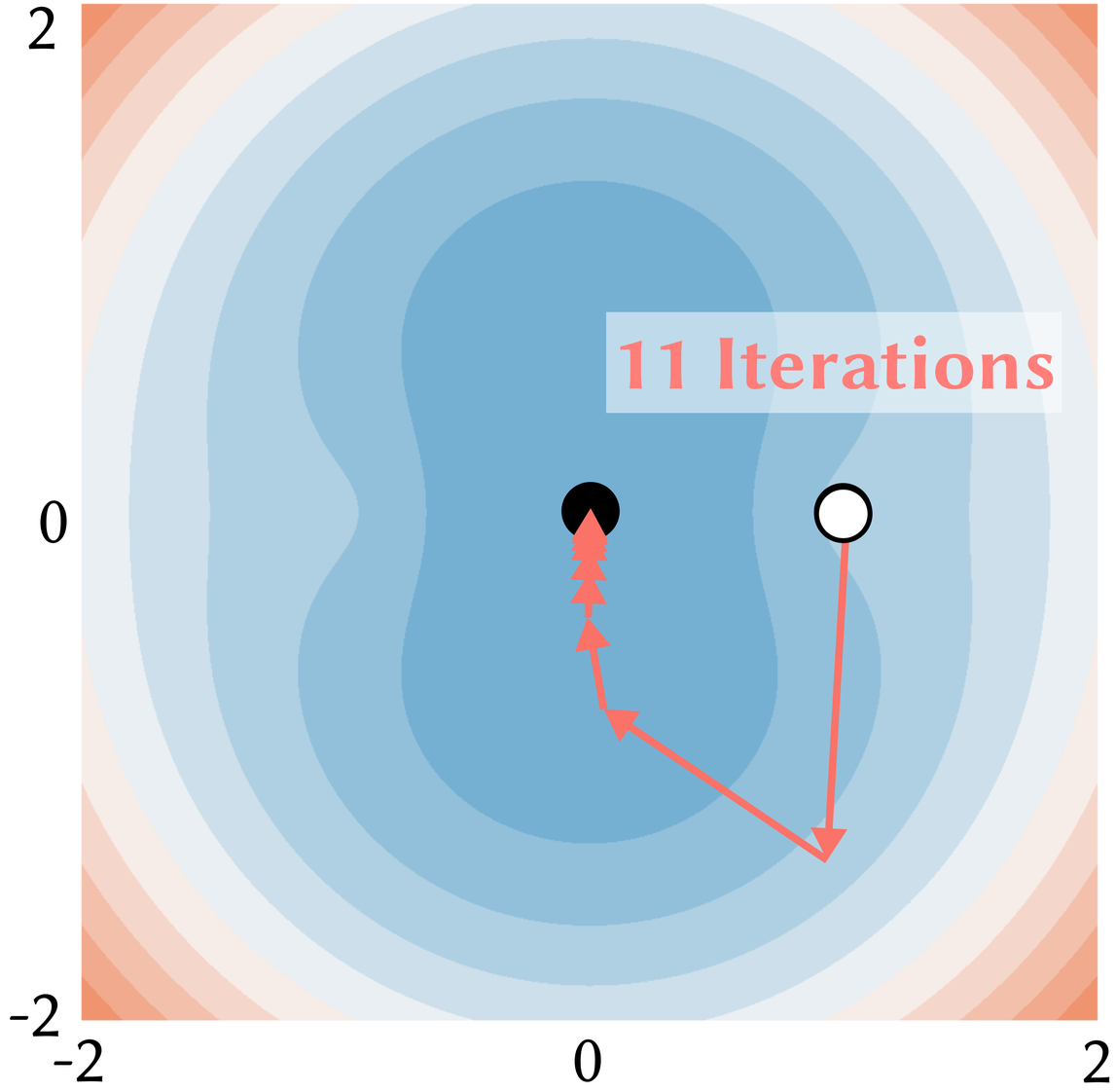
while $E(x^i + \alpha p) > E(x^i)$ **do**

  $\alpha \leftarrow \alpha/2;$

Descent Direction



Energy Minimization



- Elastic Energy
- **Optimization Basics & Classical Algorithms**
- Optimization Methods for Simulation
  - ✨Challenge 1✨: Handling Nonconvexity
  - ✨Challenge 2✨: Nontrivial Constraints
  - ✨Challenge 3✨: Large-scale Optimization

Classical Algorithms

How can you minimize a function without knowing much about it?

Assume it is much simpler than it really is

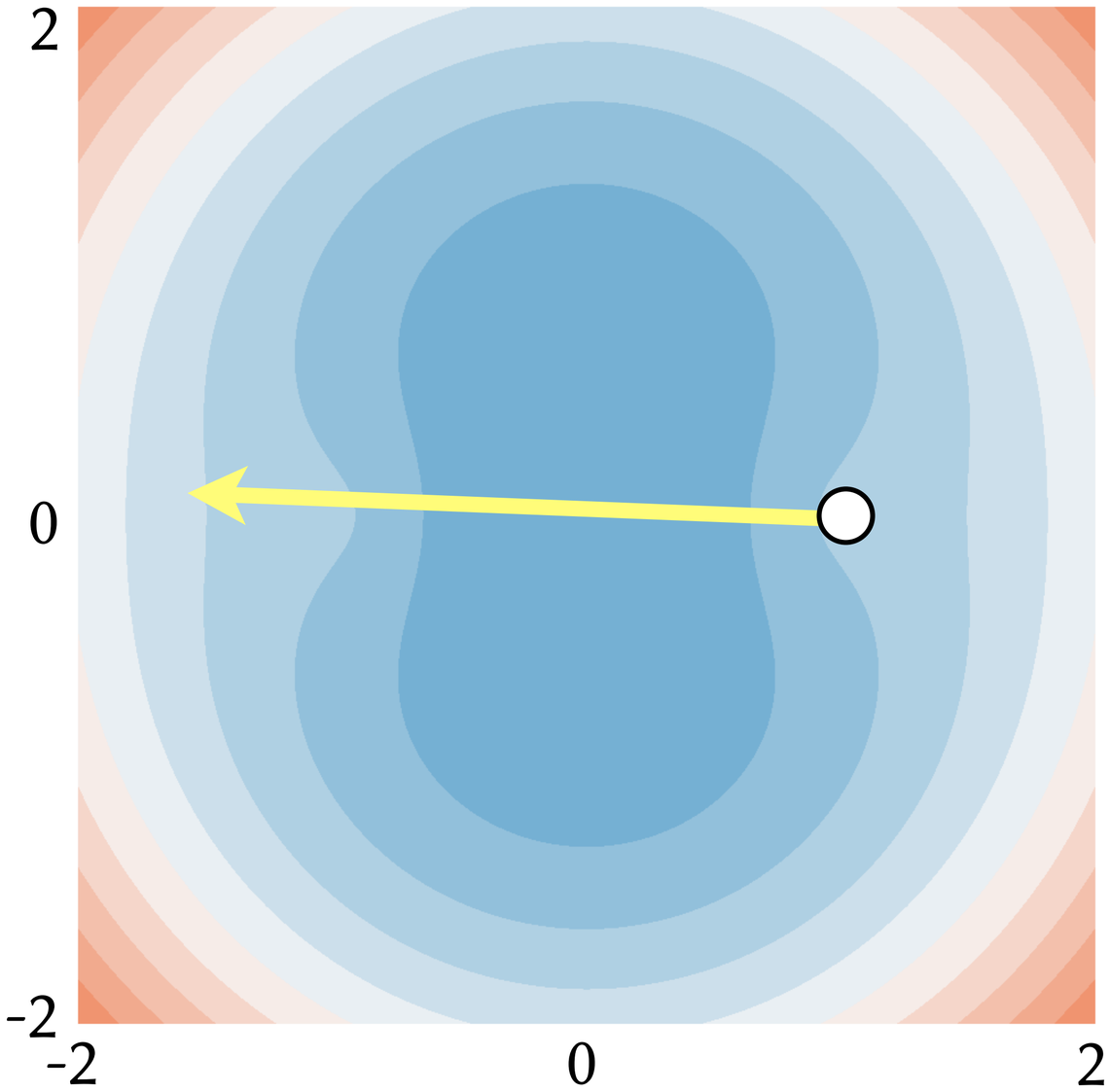
Taylor Expansion for Approximation

If f is continuously twice differentiable, then

$$f(\mathbf{x}+\mathbf{d}) \approx f(\mathbf{x}) + \mathbf{g}^\top \mathbf{d} + \frac{1}{2}\mathbf{d}^\top \mathbf{H}\mathbf{d} + \mathrm{O}(\|\mathbf{d}\|^3)$$

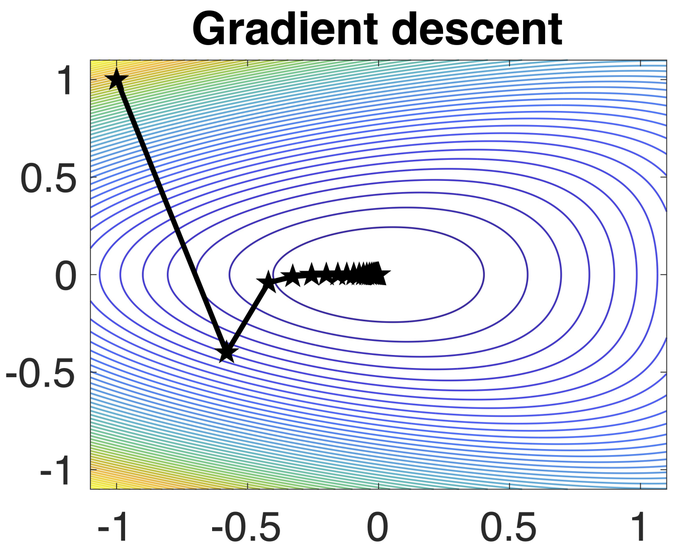
Gradient **Hessian**

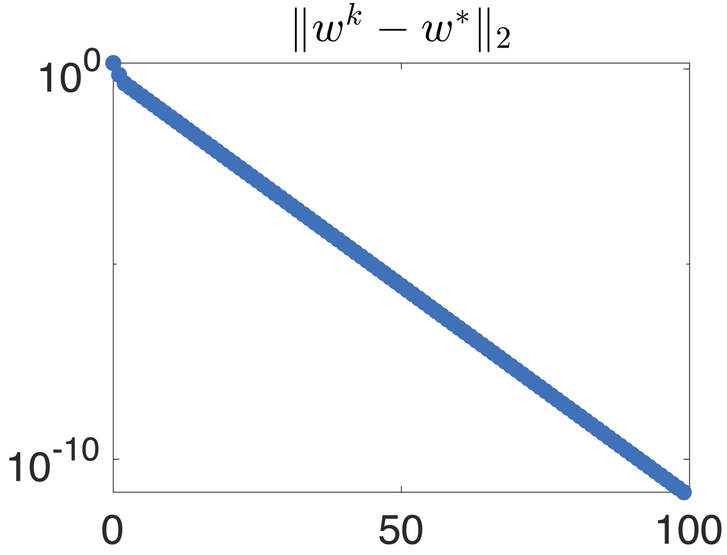
Gradient Descent: Use First-order Information

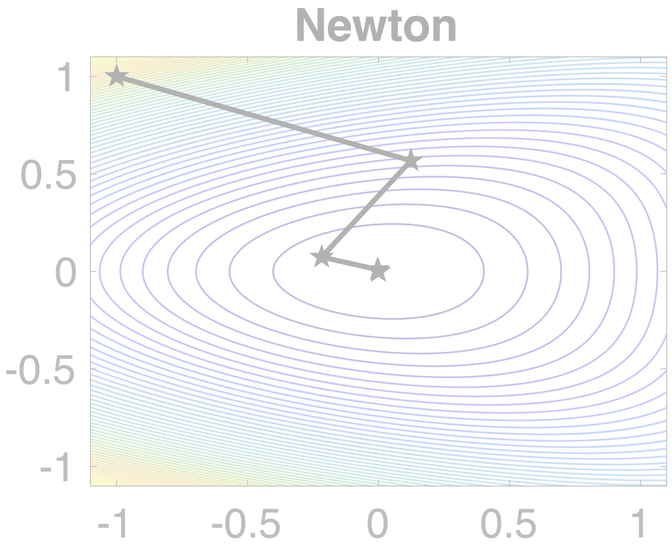


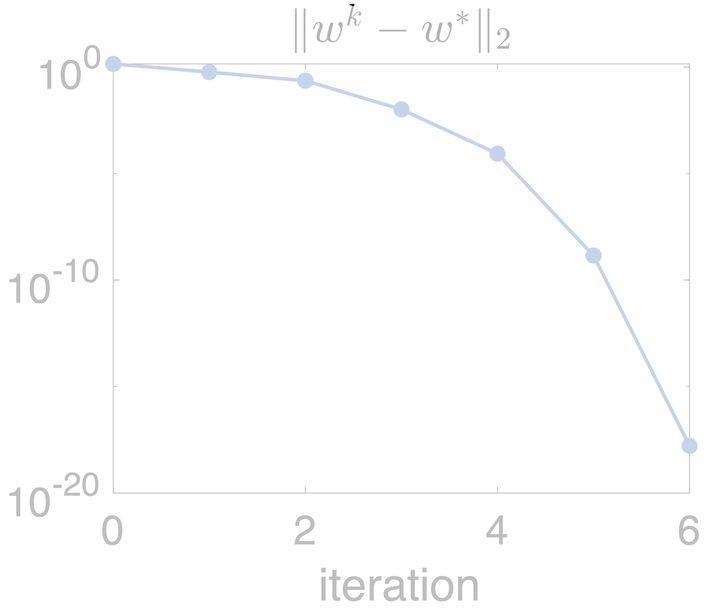
$$\mathbf{d} = -\mathbf{g}$$

Gradient Descent Can be Slow







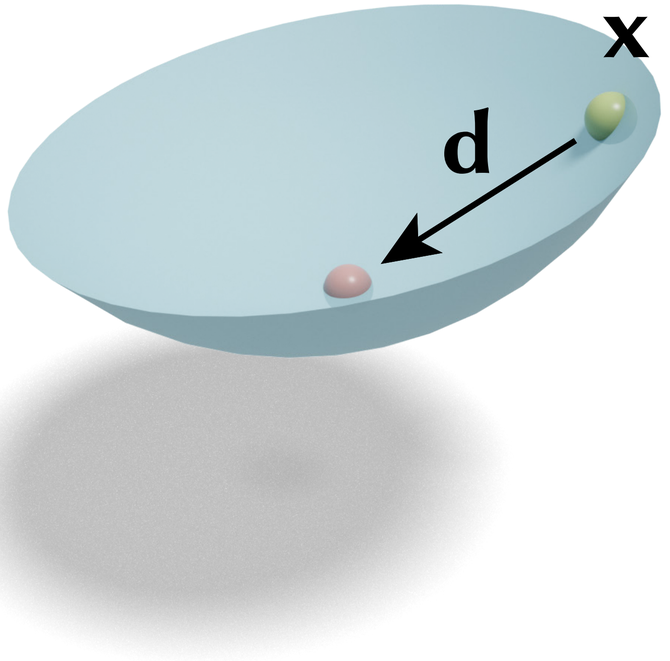


Newton's Method: Use Second-order Information

"Pretend" the function is quadratic:

$$\min_{\mathbf{d}} f(\mathbf{x}) + \mathbf{g}^\top \mathbf{d} + \frac{1}{2}\mathbf{d}^\top \mathbf{H}\mathbf{d}$$

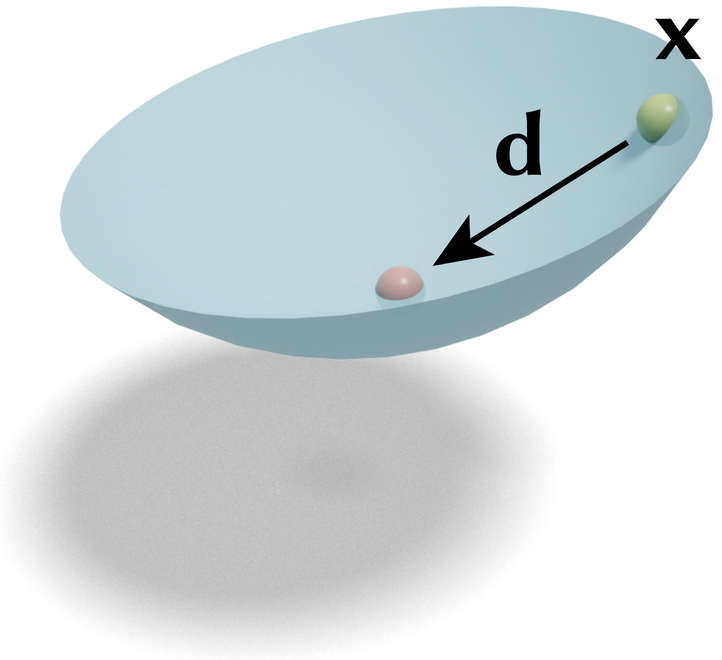
Set the gradient w.r.t. d to be 0



Newton's Method

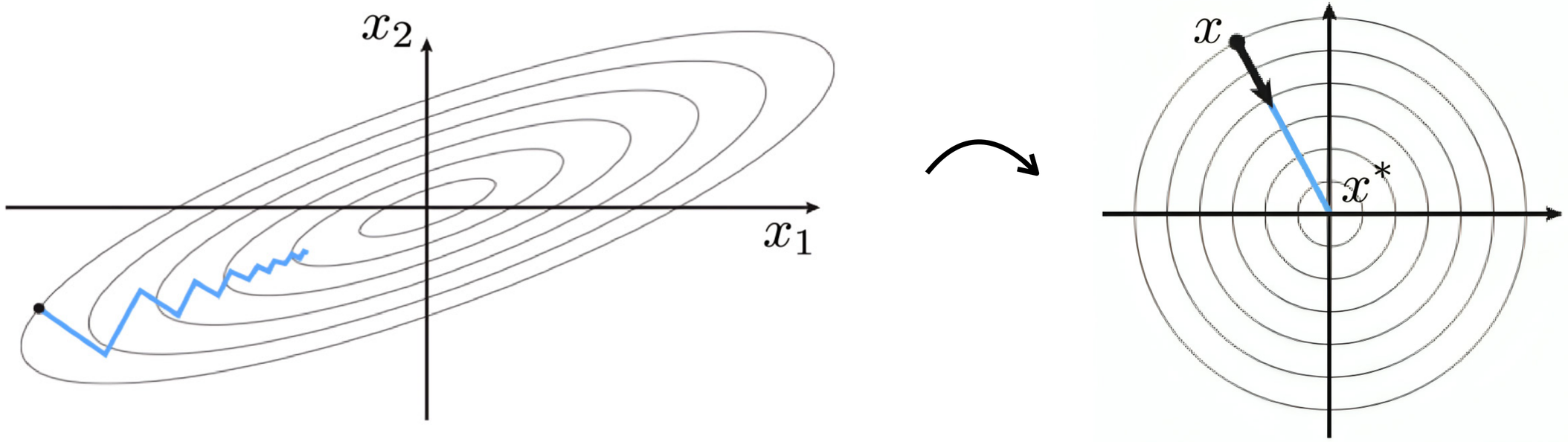
If the function f is convex,

$$\mathbf{d} = -\mathbf{H}^{-1}\mathbf{g}$$

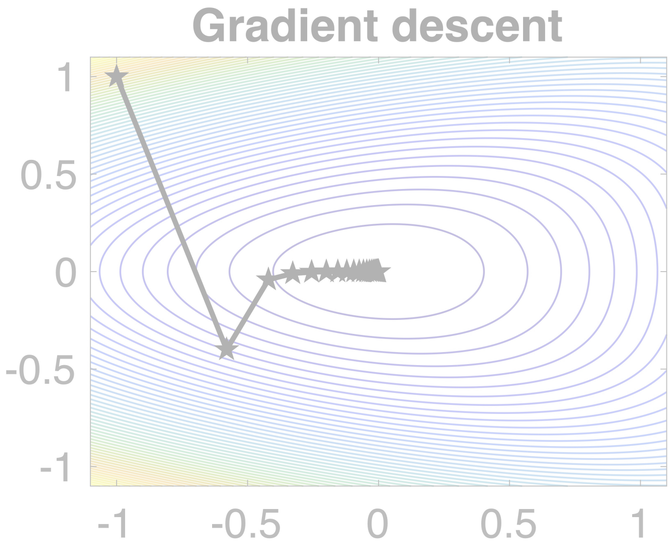


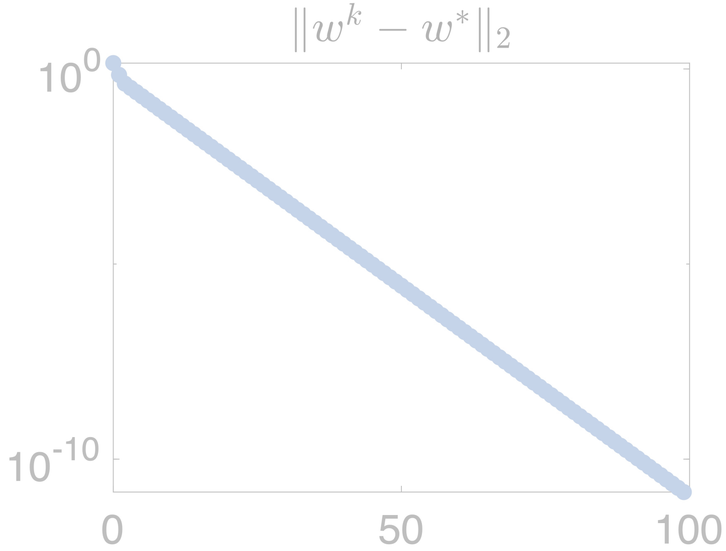
Newton's Method

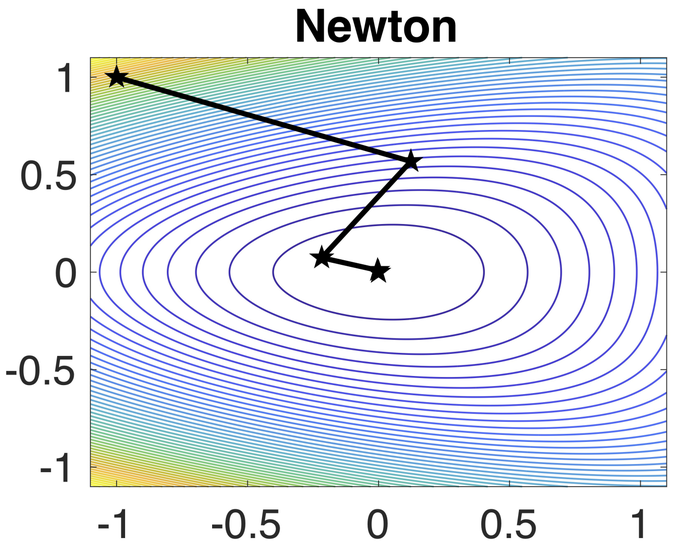
- Idea: apply a coordinate transformation so that the local energy landscape looks more like a "round bowl"
  - The gradient now directly points toward the nearby minimizer

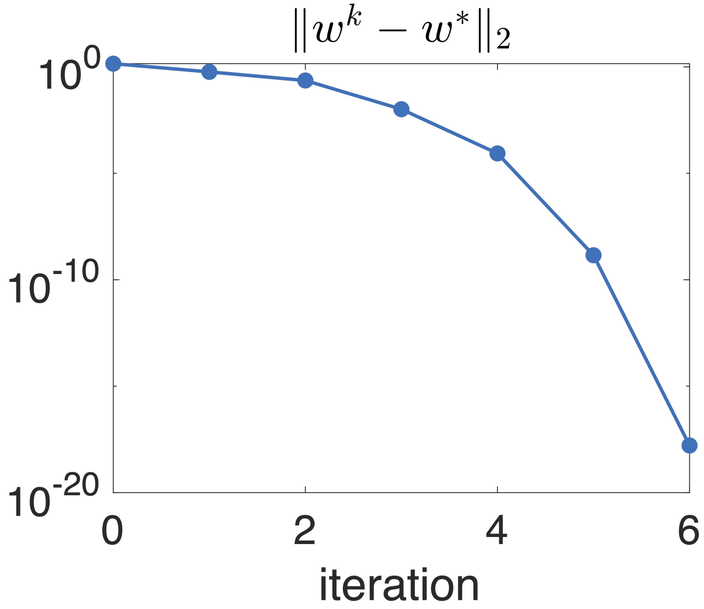


Fast Convergence of Newton's Method









Auto-differentiation Libraries

PyTorch

TinyAD

Easy to compute the gradient and the (sparse) Hessian :)

In C++

TinyAD

```
// Compute symmetric Dirichlet energy
Eigen::Matrix2<T> J = M * Mr.inverse();
return A * (J.squaredNorm() + J.inverse().squaredNorm());
```

```
#include <TinyAD/ScalarFunction.hh>

// Set up a function with 2D vertex positions as variables
auto func = TinyAD::scalar_function<2>(mesh.vertices());

// Add an objective term per triangle. Each connecting 3 vertices
func.add_elements<3>(mesh.faces(), [&] (auto& element)
{
    // Element is evaluated with either double or TinyAD::Double<6>
    using T = TINYAD_SCALAR_TYPE(element);

    // Get variable 2D vertex positions of triangle t
    OpenMesh::SmartFaceHandle t = element.handle;
    Eigen::Vector2<T> a = element.variables(t.halfedge().to());
    Eigen::Vector2<T> b = element.variables(t.halfedge().next().to());
    Eigen::Vector2<T> c = element.variables(t.halfedge().from());

    return ...
});

// Evaluate the funcion using any of these methods:
double f = func.eval(x);
auto [f, g] = func.eval_with_gradient(x);
auto [f, g, H] = func.eval_with_derivatives(x);
auto [f, g, H_proj] = func.eval_with_hessian_proj(x);
...
```

Just write out the energy

Link: https://github.com/patr-schm/TinyAD

- Elastic Energy
- Optimization Basics & Classical Algorithms
- **Optimization Methods for Simulation**
  - ✨Challenge 1✨: Handling Nonconvexity
  - ✨Challenge 2✨: Nontrivial Constraints
  - ✨Challenge 3✨: Large-scale Optimization

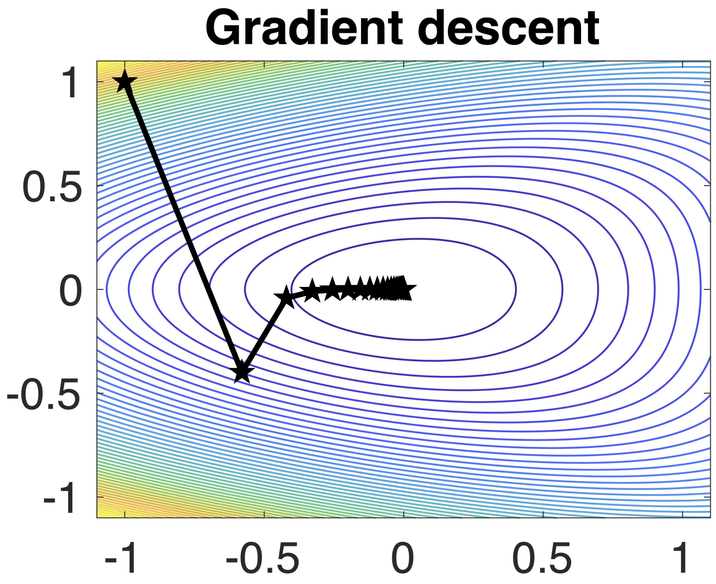
Optimization Methods for Simulation

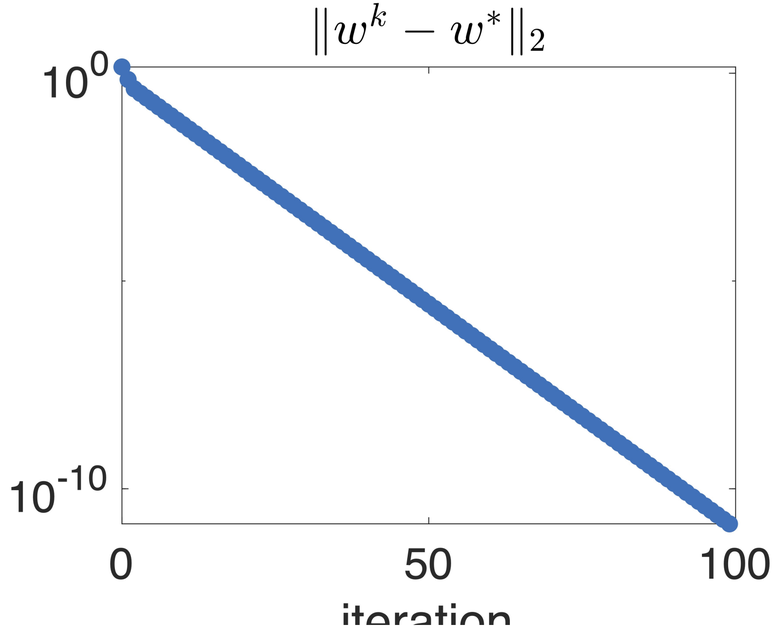
What are the challenges of applying those classical optimization methods to simulation?

Why not just use gradient descent?

$$\mathbf{d} = -\mathbf{g}$$

- The convergence of vanilla gradient descent can be very slow in higher dimensions

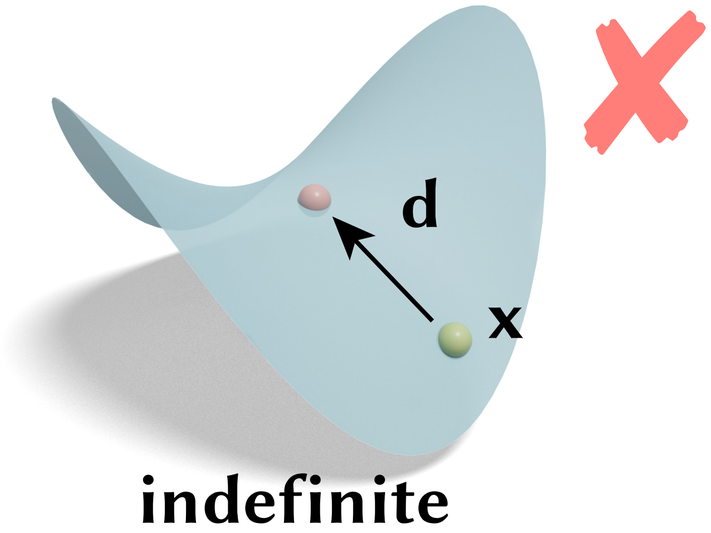




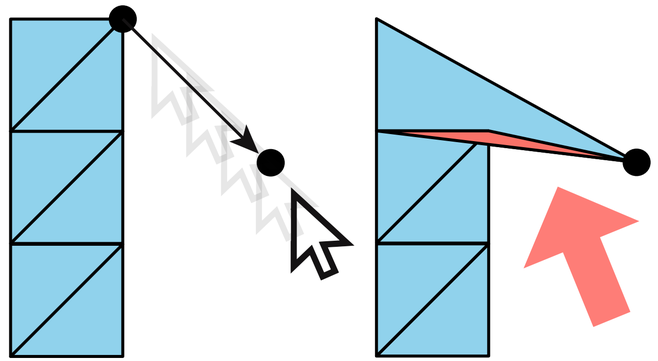
Why not just directly use Newton's method?

$$\mathbf{d} = -\mathbf{H}^{-1}\mathbf{g}$$

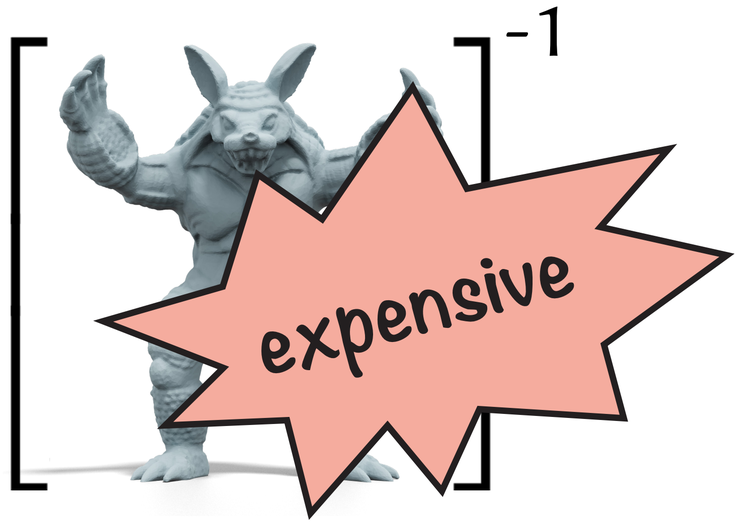
- Nonconvexity



- Infeasible points



- Large-scale



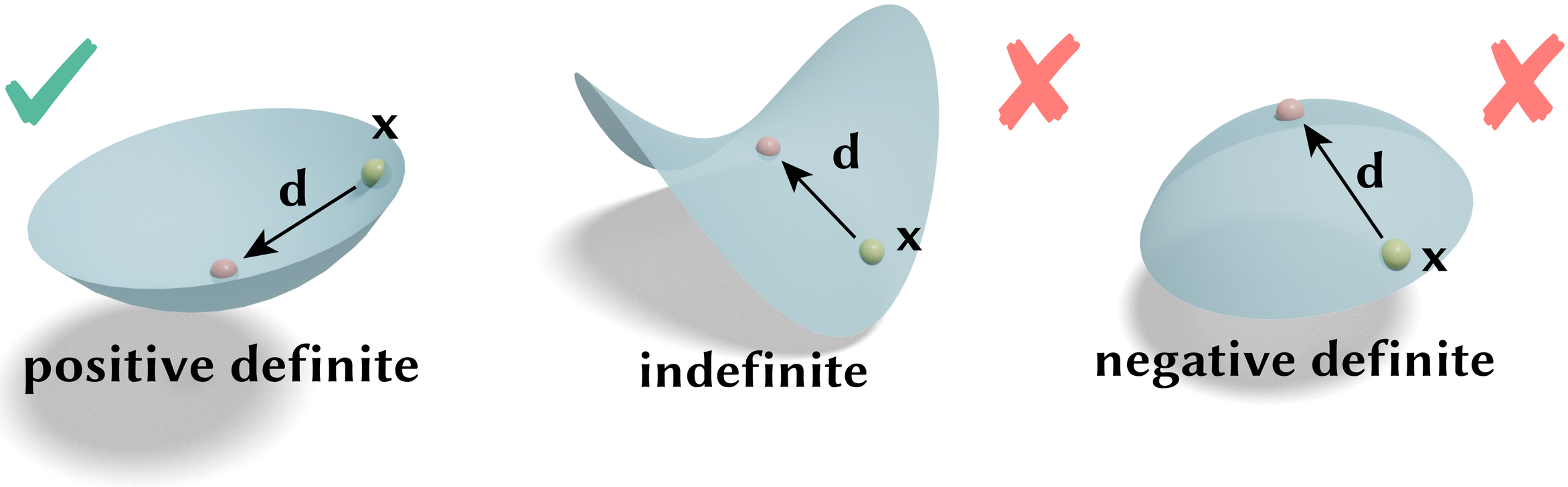
3N x 3N

- Elastic Energy
- Optimization Basics & Classical Algorithms
- Optimization Methods for Simulation
  - ✨**Challenge 1**✨**: Handling Nonconvexity**
  - ✨Challenge 2✨: Nontrivial Constraints
  - ✨Challenge 3✨: Large-scale Optimization

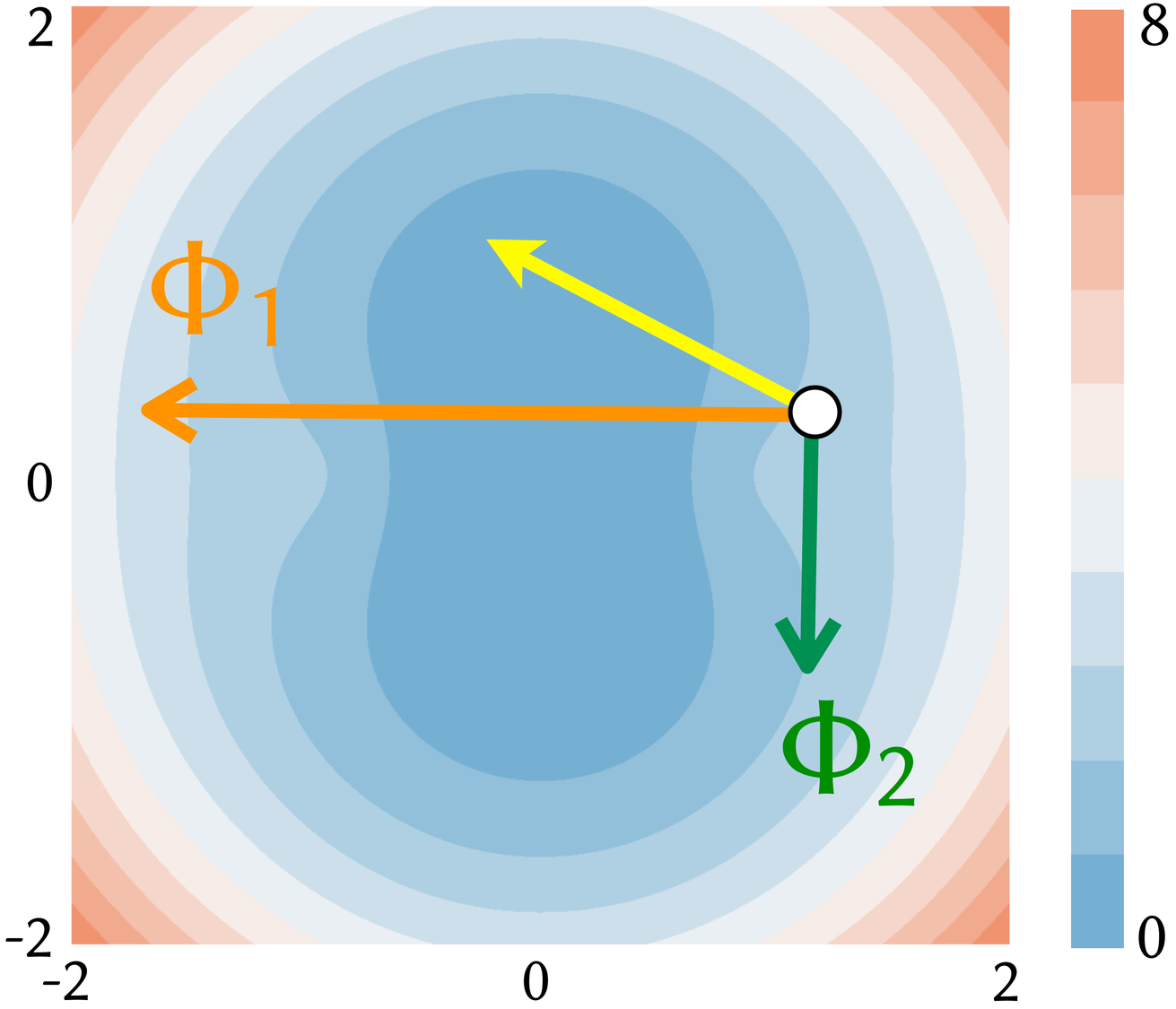
Challenge 1: Handling Nonconvexity

Challenge 1: Handling Nonconvexity

$$\mathbf{d} = -\mathbf{H}^{-1}\mathbf{g}$$



Why is Nonconvexity Problematic?

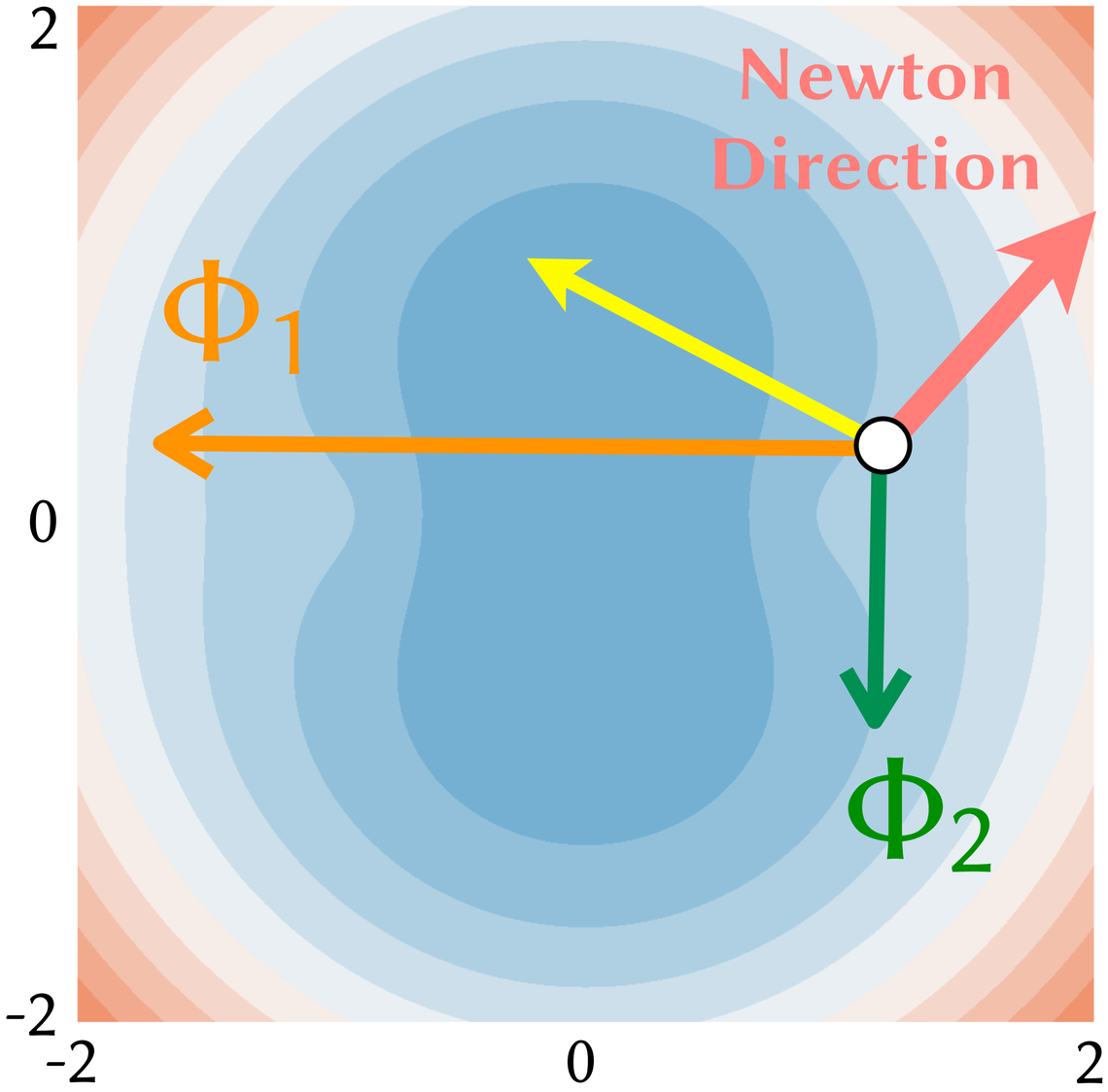


$$-\nabla f$$

$$\mathbf{H} = \Phi \begin{bmatrix} -6 & 0 \\ 0 & 3 \end{bmatrix} \Phi^\top$$

$$f(x,y) = \left(\sqrt{(x+1)^2+y^2}-1\right)^2 + \left(\sqrt{(x-1)^2+y^2}-1\right)^2$$

Choose $(x,y) = (1, 0.2)$



$$-\mathbf{H}^{-1}\nabla f$$

$$= -\left(\Phi \begin{bmatrix} -6 & 0 \\ 0 & 3 \end{bmatrix} \Phi^\top\right)^{-1} \nabla f$$

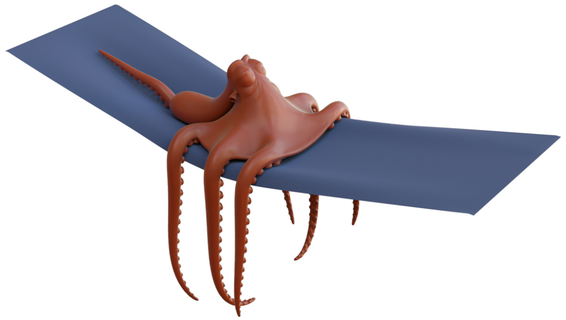
$$= -\left(\Phi \begin{bmatrix} -\frac{1}{6} & 0 \\ 0 & \frac{1}{3} \end{bmatrix} \Phi^\top\right) \nabla f$$

Where does this non-convexity come from?

Course, SIGGRAPH 2022

Dynamic Deformables: Implementation and Production Practicalities *(Now With Code!)*

Instructors:
Theodore Kim, Yale University
David Eberle, Pixar Animation Studios



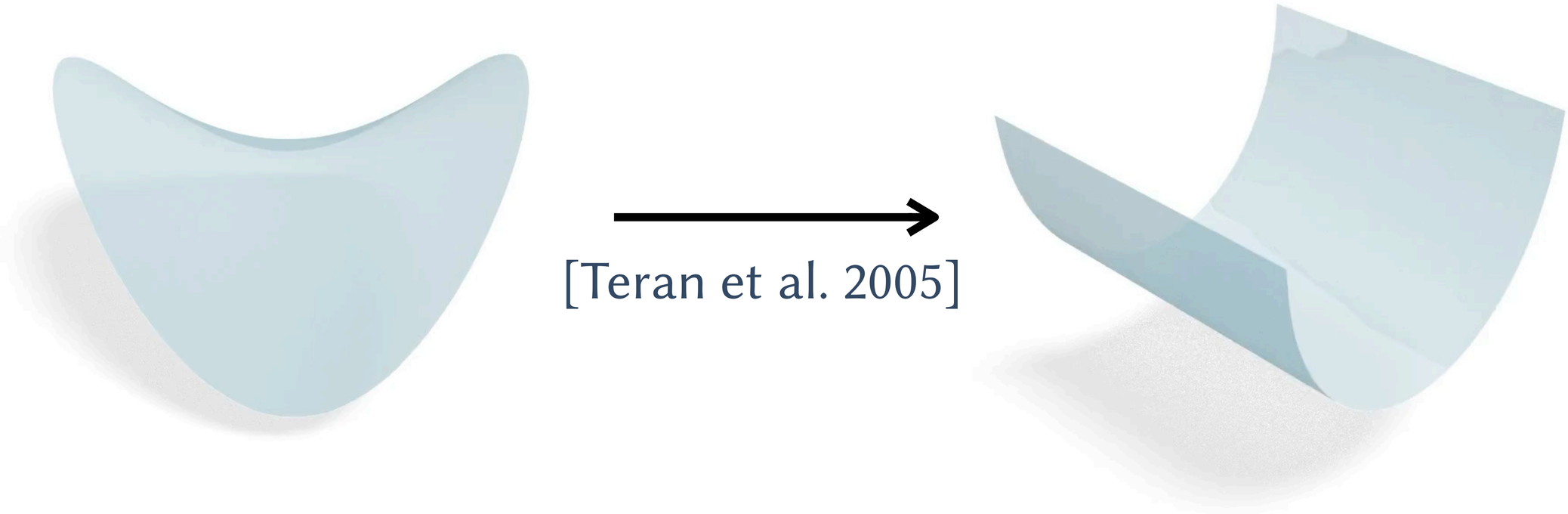
Built on: April 25, 2024

Projected Newton

$$\mathbf{d} = -(\mathbf{H}^{+})^{-1}\mathbf{g}$$

Positive Definite

Eigenvalue Clamping



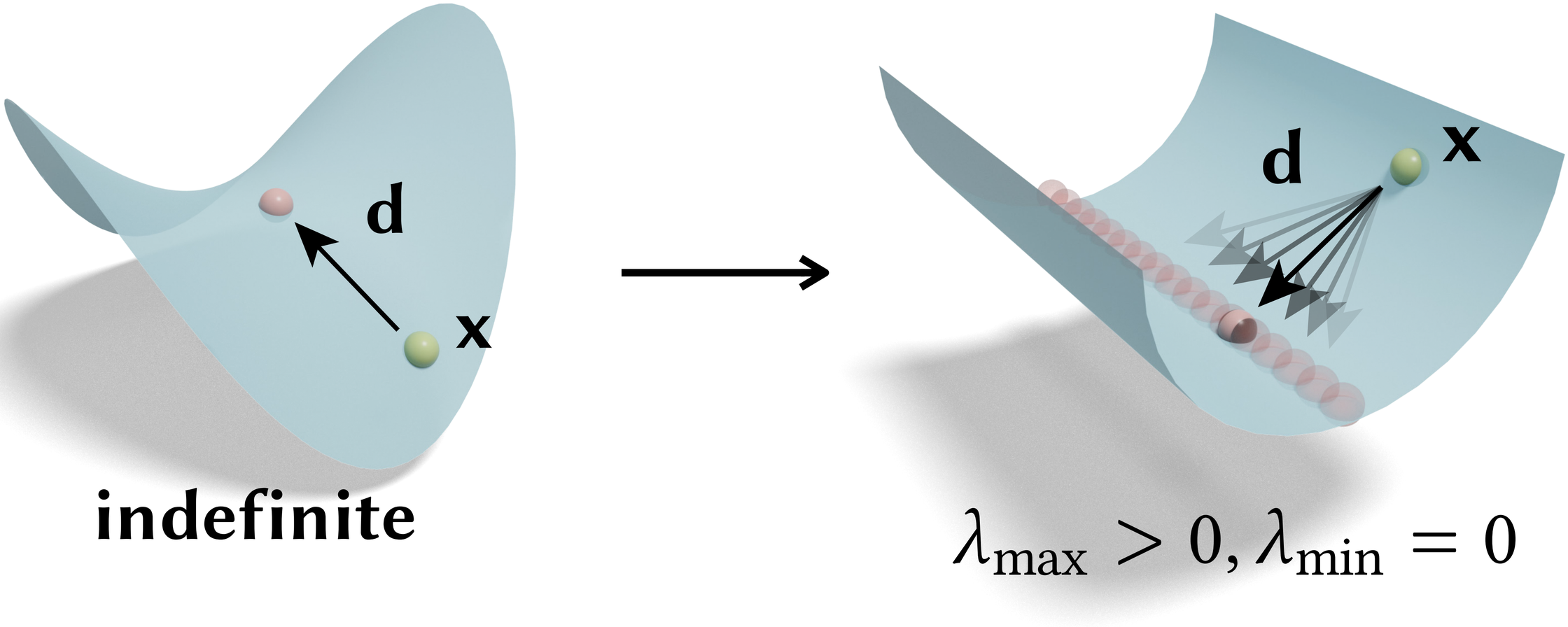
[Teran et al. 2005]

$$\mathbf{H} = \sum_i \mathbf{P}_i^\top \mathbf{H}_i \mathbf{P}_i$$

$$\mathbf{H}_i = \mathbf{U}\Lambda\mathbf{U}^T$$

$$\Lambda_k^+ = \max(\Lambda_k, \epsilon)$$

Eigenvalue Clamping

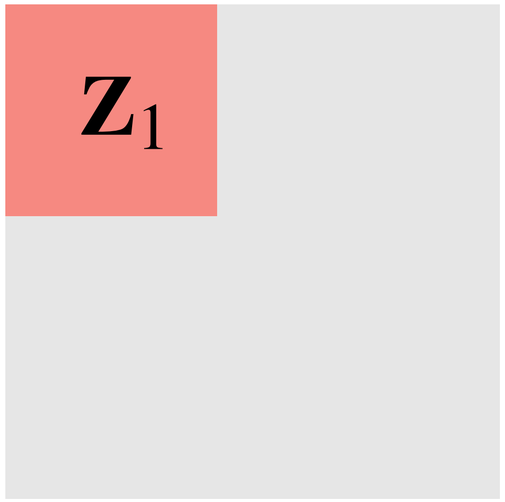


Additive Contribution of Neighboring Elements

+ Also boundary conditions

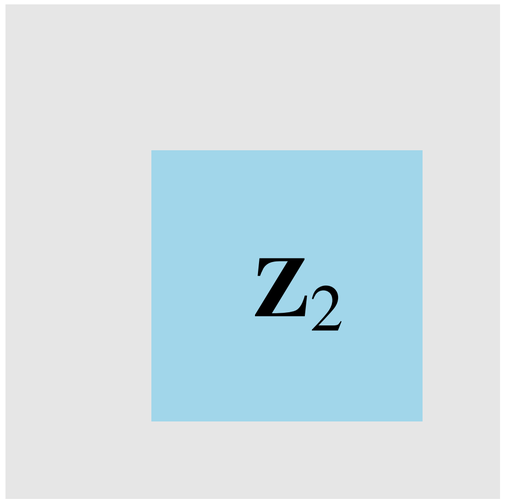
$X \succeq 0$

$=$



$P_1^\top Z_1 P_1$

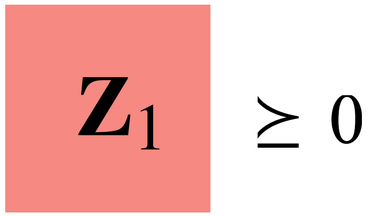
$+$



$P_2^\top Z_2 P_2$

$+$

$P_3^\top Z_3 P_3$



$\succeq 0$

$\succeq 0$

$\succeq 0$

The global Hessian can often be positive definite ($X > 0$)

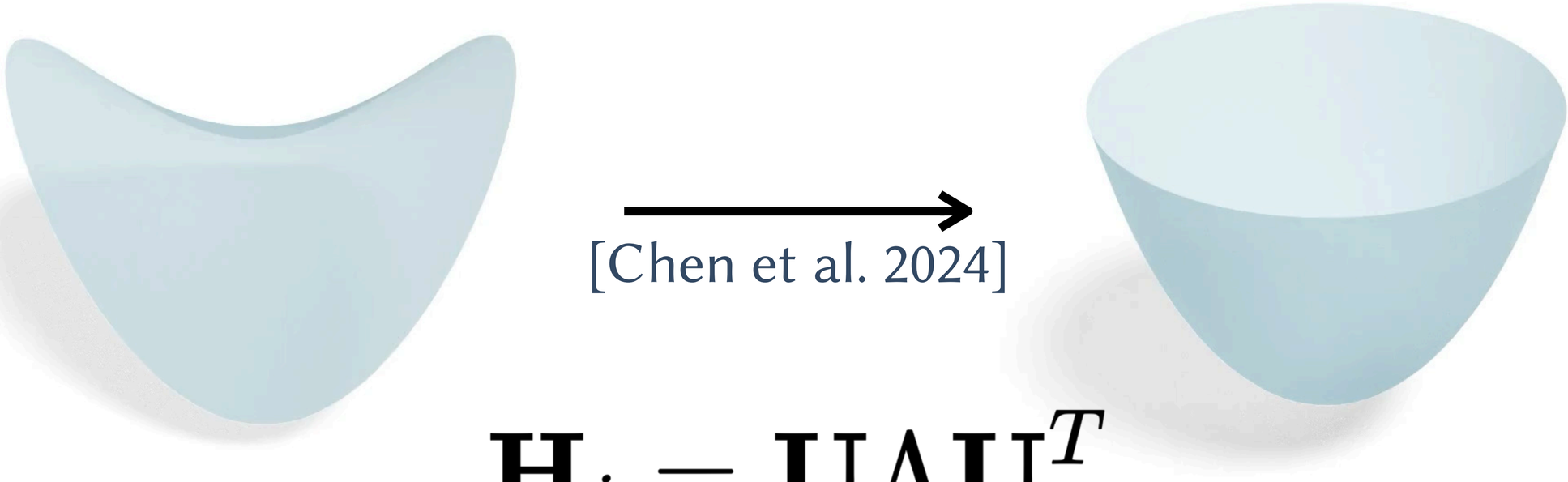
Add a multiple of Identity Matrix

[Tikhonov 1943]

$$\mathbf{H}_i = \mathbf{U}\mathbf{\Lambda}\mathbf{U}^T$$

$$\mathbf{\Lambda}_k^+ = \mathbf{\Lambda}_k + m\mathbf{I}$$

Absolute Eigenvalue Projection



[Chen et al. 2024]

$$\mathbf{H}_i = \mathbf{U}\Lambda\mathbf{U}^T$$

$$\Lambda_k^+ = |\Lambda_k|$$

(for highly non-convex scenarios)

Recap: Projected Newton

$$\mathbf{d} = -(\mathbf{H}^{+})^{-1}\mathbf{g}$$

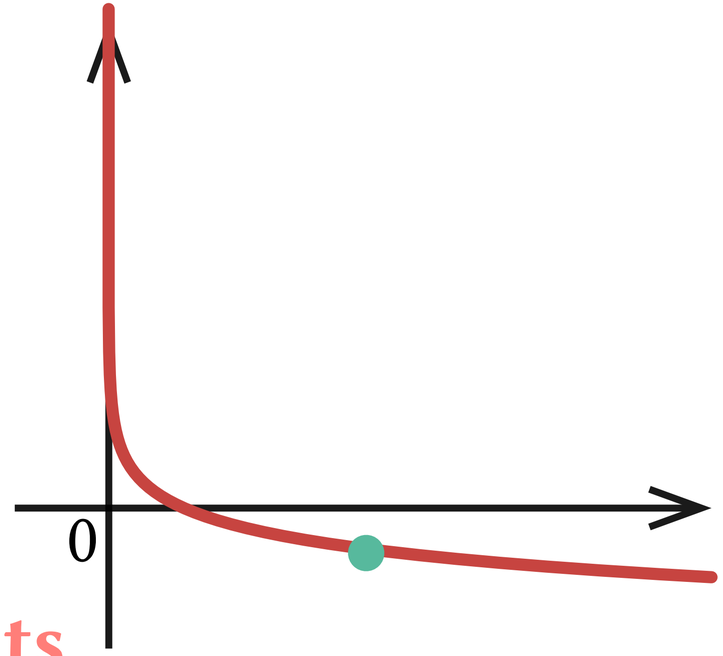
Positive Definite

- Elastic Energy
- Optimization Basics & Classical Algorithms
- Optimization Methods for Simulation
  - ✨Challenge 1✨: Handling Nonconvexity
  - ✨**Challenge 2**✨**: Nontrivial Constraints**
  - ✨Challenge 3✨: Large-scale Optimization

Challenge 2: Nontrivial Constraints

Nontrivial Constraints as a Barrier Function

Undefined when x < 0



Infeasible Points

$$f(x) = -\log(x)$$

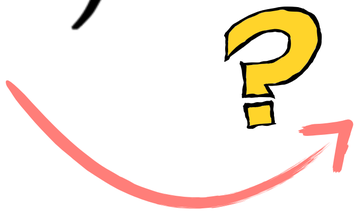
Barrier term

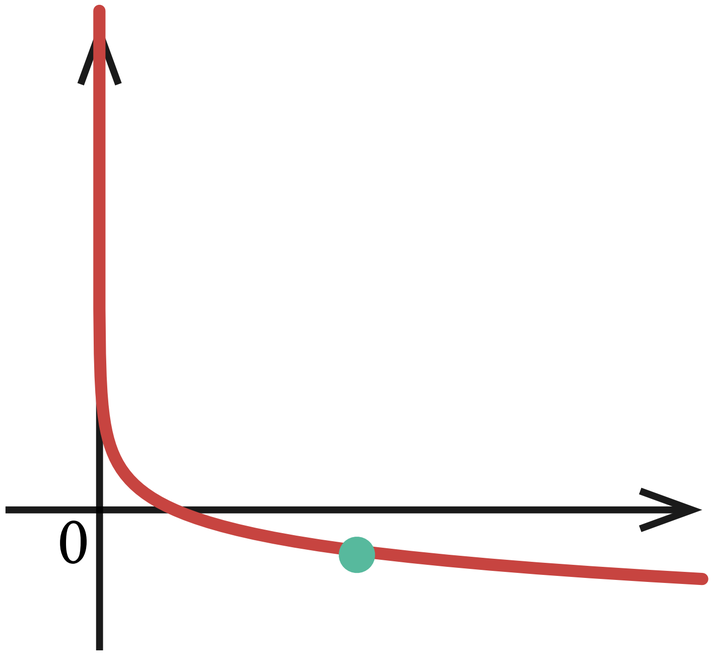
Example 1: Neo-Hookean Energy

$$E_{\text{stretch}}(\ ,\ ) + E_{\text{vol}}(\ ,\ )$$

$$-\mu \log(\ ,\ )$$

(inversion)





Strategy 1: Remove the infeasibility in the energy

$$E_{\text{stretch}}(\ ,\ ) + E_{\text{vol}}(\ ,\ )$$

$$-\mu \log(\ ,\ )$$

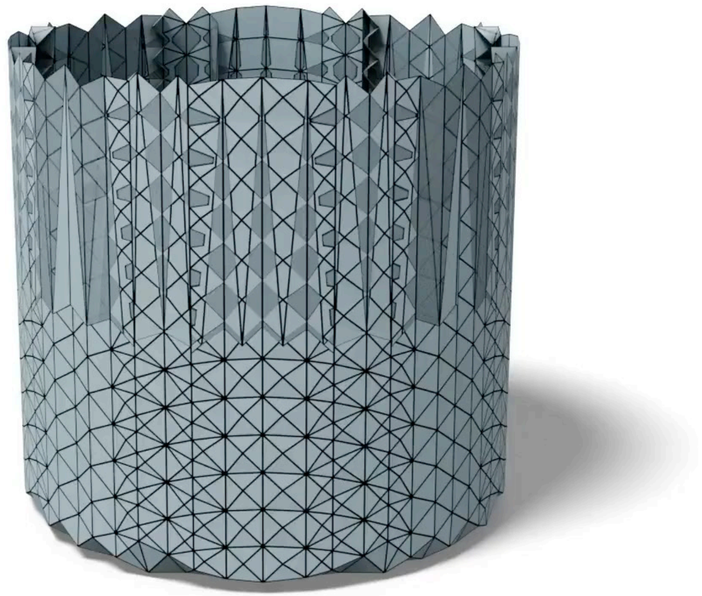
[Smith et al. 2018]

Stable Neo-Hookean Flesh Simulation.
Breannan Smith, Fernando de Goes, and Theodore Kim. ACM Trans. Graph. 2018.

Strategy 1: Remove the infeasibility in the energy

✓ **Stable even when inversion exists before or during the optimization**

✗ **No guarantee to be inversion-free**



Example 2: Incremental Potential Contact (IPC)

Incremental Potential Contact: Intersection- and Inversion-free, Large-Deformation Dynamics

MINCHEN LI, University of Pennsylvania & Adobe Research
ZACHARY FERGUSON and TESEO SCHNEIDER, New York University
TIMOTHY LANGLOIS, Adobe Research
DENIS ZORIN and DANIELE PANOZZO, New York University
CHENFANFU JIANG, University of Pennsylvania
DANNY M. KAUFMAN, Adobe Research

Fig. 1. **Squeeze out:** Incremental Potential Contact (IPC) enables high-rate time stepping, here with $h = 0.01$s, of extreme nonlinear elastodynamics with contact that is intersection- and inversion-free at all time steps, irrespective of the degree of compression and contact. Here a plate compresses and then forces a collection of complex soft elastic FE models (181K tetrahedra in total, with a neo-Hookean material) through a thin, codimensional obstacle tube. The models are then compressed entirely together forming a tight mush to fit through the gap and then once through they cleanly separate into a stable pile.

Contacts weave through every aspect of our physical world, from daily household chores to acts of nature. Modeling and predictive computation of these phenomena for solid mechanics is important to every discipline concerned with the motion of mechanical systems, including engineering and animation. Nevertheless, efficiently time-stepping accurate and consistent simulations of real-world contacting elastica remains an outstanding computational challenge. To model the complex interaction of deforming solids in contact we propose Incremental Potential Contact (IPC) – a new model and algorithm for variationally solving implicitly time-stepped nonlinear elastodynamics. IPC maintains an intersection- and inversion-free trajectory regardless of material parameters, time step sizes, impact velocities, severity of deformation, or boundary conditions enforced.

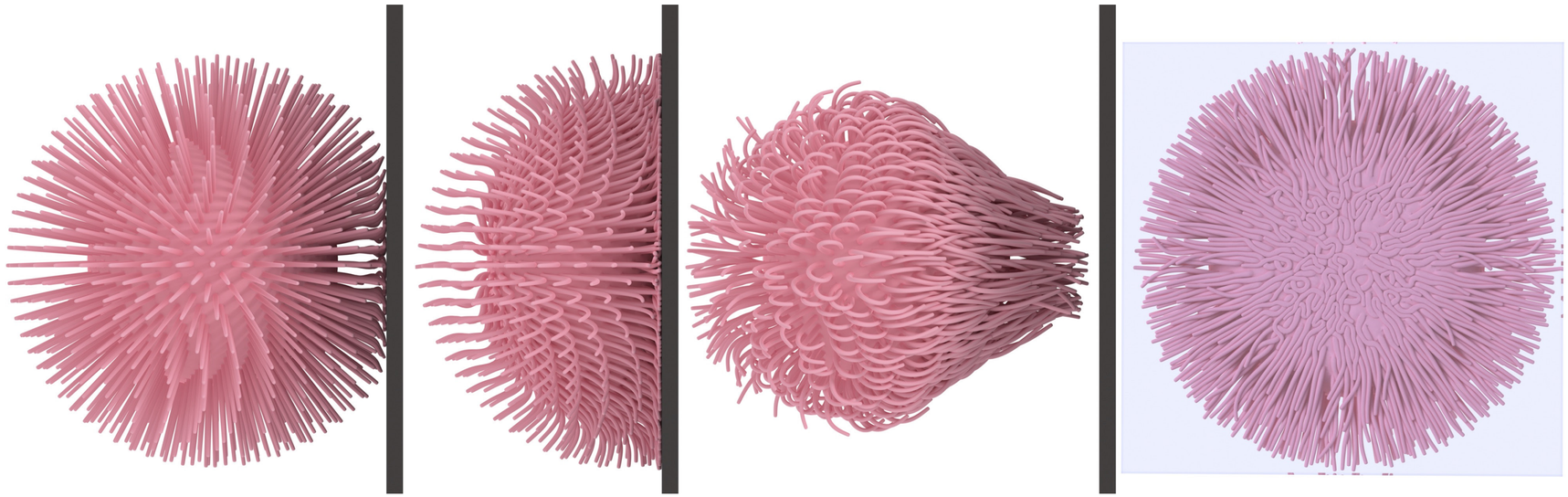
Constructed with a custom nonlinear solver, IPC enables efficient resolution of time-stepping problems with separate, user-exposed accuracy tolerances that allow independent specification of the physical accuracy of the dynamics and the geometric accuracy of surface-to-surface conformation. This enables users to decouple, as needed per application, desired accuracies for a simulation's dynamics and geometry.

The resulting time stepper solves contact problems that are intersection-free (and thus robust), inversion-free, efficient (at speeds comparable to or faster than available methods that lack both convergence and feasibility), and accurate (solved to user-specified accuracies). To our knowledge this is the first implicit time-stepping method, across both the engineering and graphics literature that can consistently enforce these guarantees as we vary simulation parameters.

In an extensive comparison of available simulation methods, research libraries and commercial codes we confirm that available engineering and computer graphics methods, while each succeeding admirably in custom-tuned regimes, often fail with instabilities, egregious constraint violations and/or inaccurate and implausible solutions, as we vary input materials, contact numbers and time step. We also exercise IPC across a wide range

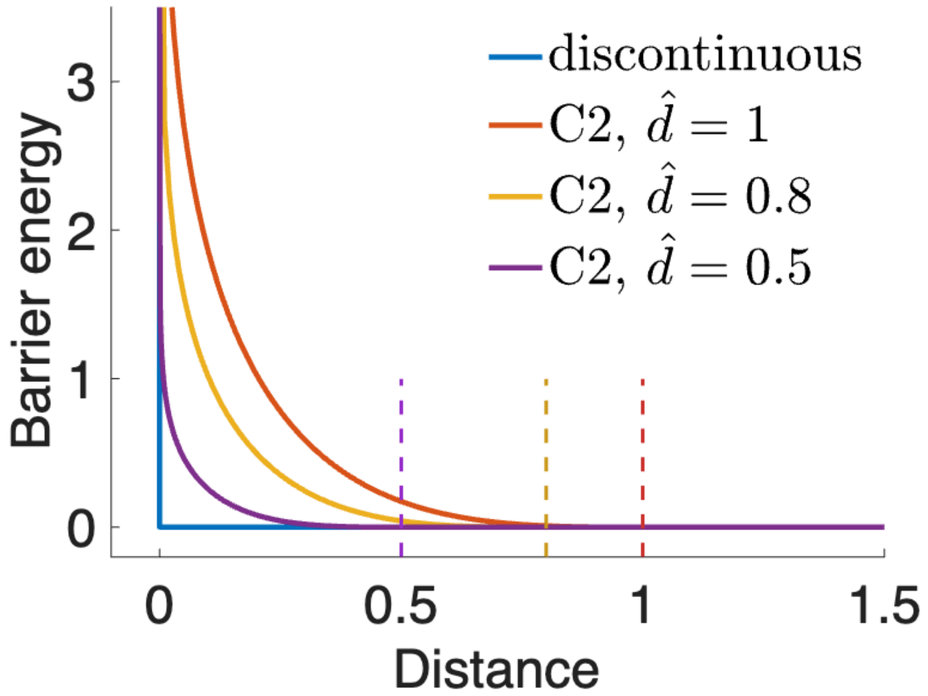
Authors' addresses: Minchen Li, University of Pennsylvania & Adobe Research, minchenli@gmail.com; Zachary Ferguson; Teseo Schneider, New York University; Timothy Langlois, Adobe Research; Denis Zorin; Daniele Panozzo, New York University; Chenfanfu Jiang, University of Pennsylvania; Danny M. Kaufman, Adobe Research, dannykaufman@gmail.com.

0730-0301/2020/7-ART49 $15.00
https://doi.org/10.1145/3386569.3392425

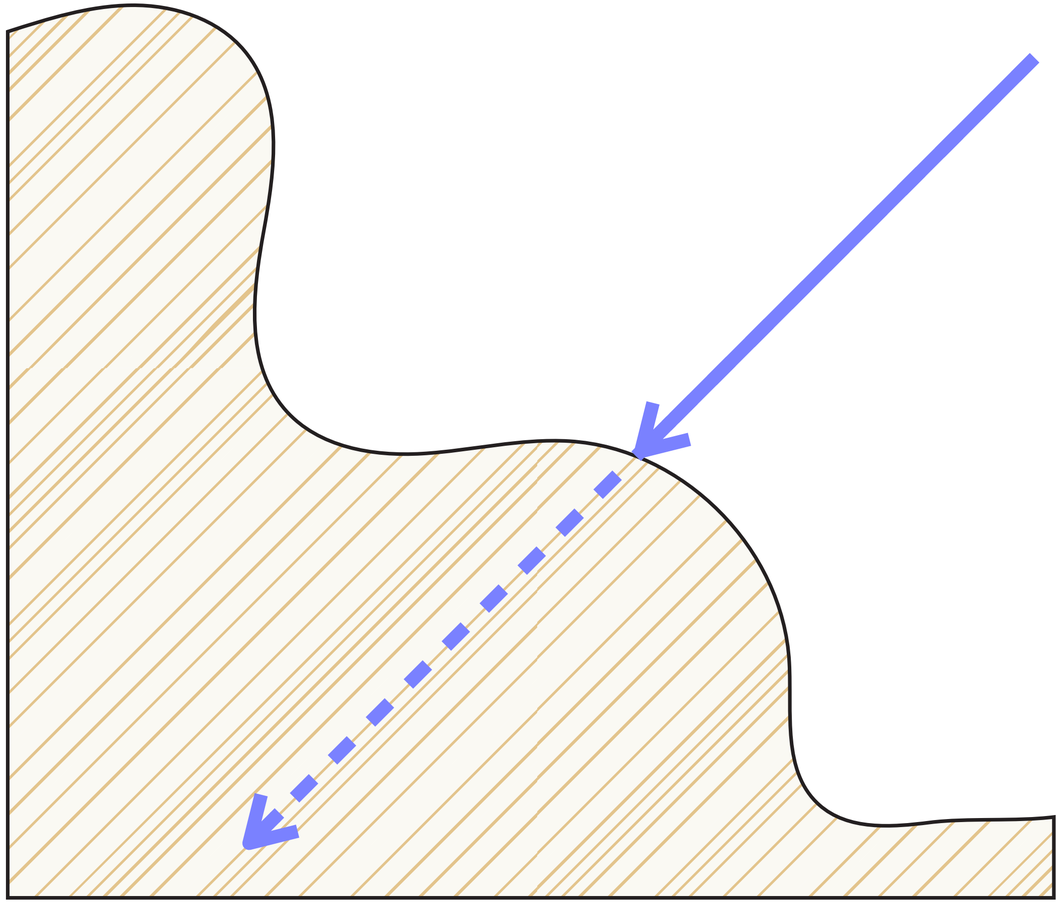


Incremental Potential Contact

$$b(d,\hat{d}) = \begin{cases} -(d-\hat{d})^2 \ln\left(\frac{d}{\hat{d}}\right), & 0 < d < \hat{d} \\ 0 & d \geq \hat{d} \end{cases}$$



Strategy 2: Avoid going to infeasible regions



feasibility-aware line search

✓ **Guarantee to satisfy the constraints**

✗ **Line search is more complex and may clamp down the step size too much**

- Elastic Energy
- Optimization Basics & Classical Algorithms
- Optimization Methods for Simulation
  - ✨Challenge 1✨: Handling Nonconvexity
  - ✨Challenge 2✨: Nontrivial Constraints
  - ✨**Challenge 3**✨**: Large-scale Optimization**

Challenge 3: Large-scale Optimization

Large-Scale Optimization

$$\mathbf{d} = -\mathbf{H}^{-1}\mathbf{g}$$

3N x 3N

Total Computational Cost

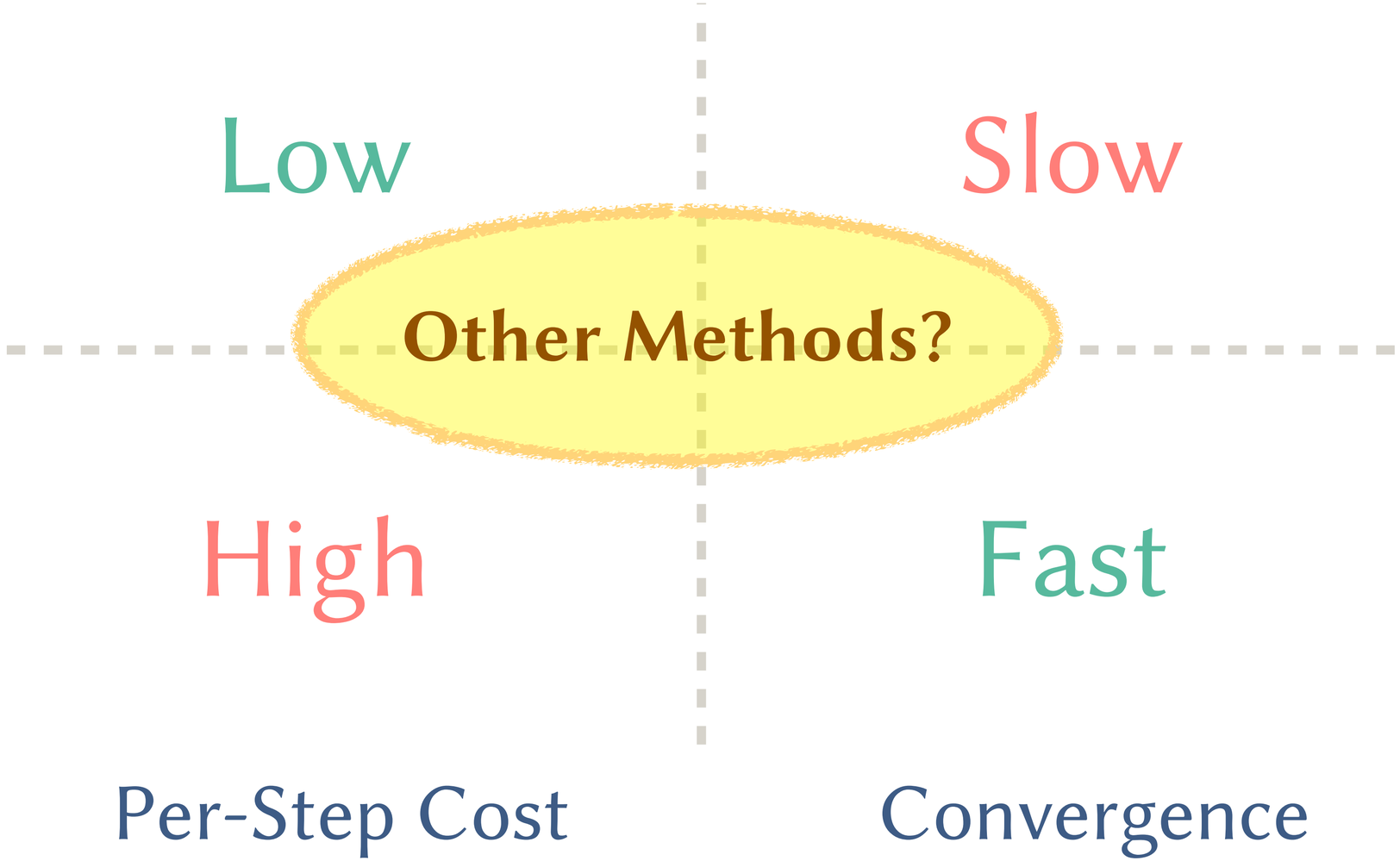
Total Cost =

Per-step Cost x Iterations

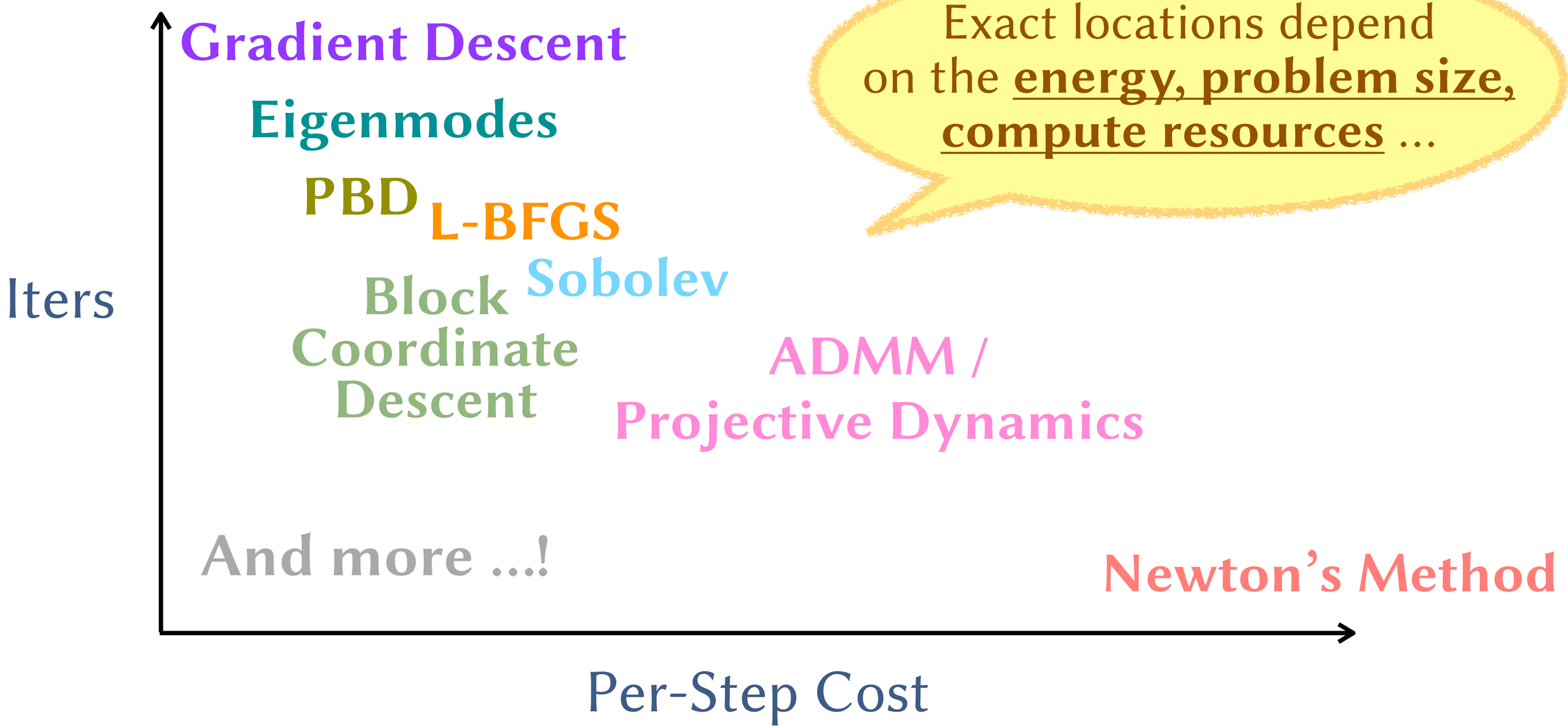
Total Cost = Per-step Cost x Iterations

Gradient Descent
(First-order)

Newton's Method
(Second-order)



An Inaccurate Map



Exact locations depend on the **energy, problem size, compute resources** ...

Overall Goal

$$\mathbf{d} = -\mathbf{H}^{-1}\mathbf{g}$$

Simpler Hessian

Better gradient

Strategy 1: Simplify H

$$\mathbf{d} = -\mathbf{H}^{-1}\mathbf{g}$$

Simpler Hessian

Steepest Descent

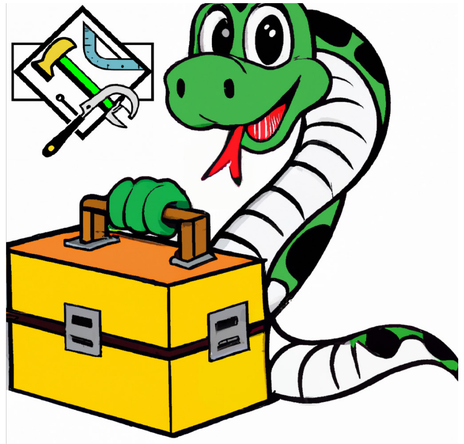
$$\mathbf{d} = -\mathbf{H}^{-1}\mathbf{g}$$

$$\mathbf{d} = -\mathbf{I}^{-1}\mathbf{g}$$

(Gradient Descent)

Sobolev-Preconditioned Gradient Descent

$$\mathbf{d} = -\mathbf{L}^{-1}\mathbf{g}$$



Gpytoolbox (python)

libigl (C++)

Limited-memory BFGS (L-BFGS)

$$\mathbf{d} = -\mathbf{H}^{-1}\mathbf{g}$$

Low-rank Update

$\widetilde{\mathbf{H}}_k^{-1}$ → $\widetilde{\mathbf{H}}_{k+1}^{-1}$

Previous Approximation
(Known)

BFGS

(Rank-2 Update)

$$\mathbf{H}_{k+1}^{-1} = \left(\mathbf{I} - \frac{\mathbf{s}_k \mathbf{y}_k^\top}{\mathbf{y}_k^\top \mathbf{s}_k}\right) \mathbf{H}_k^{-1} \left(\mathbf{I} - \frac{\mathbf{y}_k \mathbf{s}_k^\top}{\mathbf{y}_k^\top \mathbf{s}_k}\right) + \frac{\mathbf{s}_k \mathbf{s}_k^\top}{\mathbf{y}_k^\top \mathbf{s}_k}$$

$$\mathbf{y}_k = \mathbf{g}_{k+1} - \mathbf{g}_k$$

$$\mathbf{s}_k = \mathbf{x}_{k+1} - \mathbf{x}_k$$

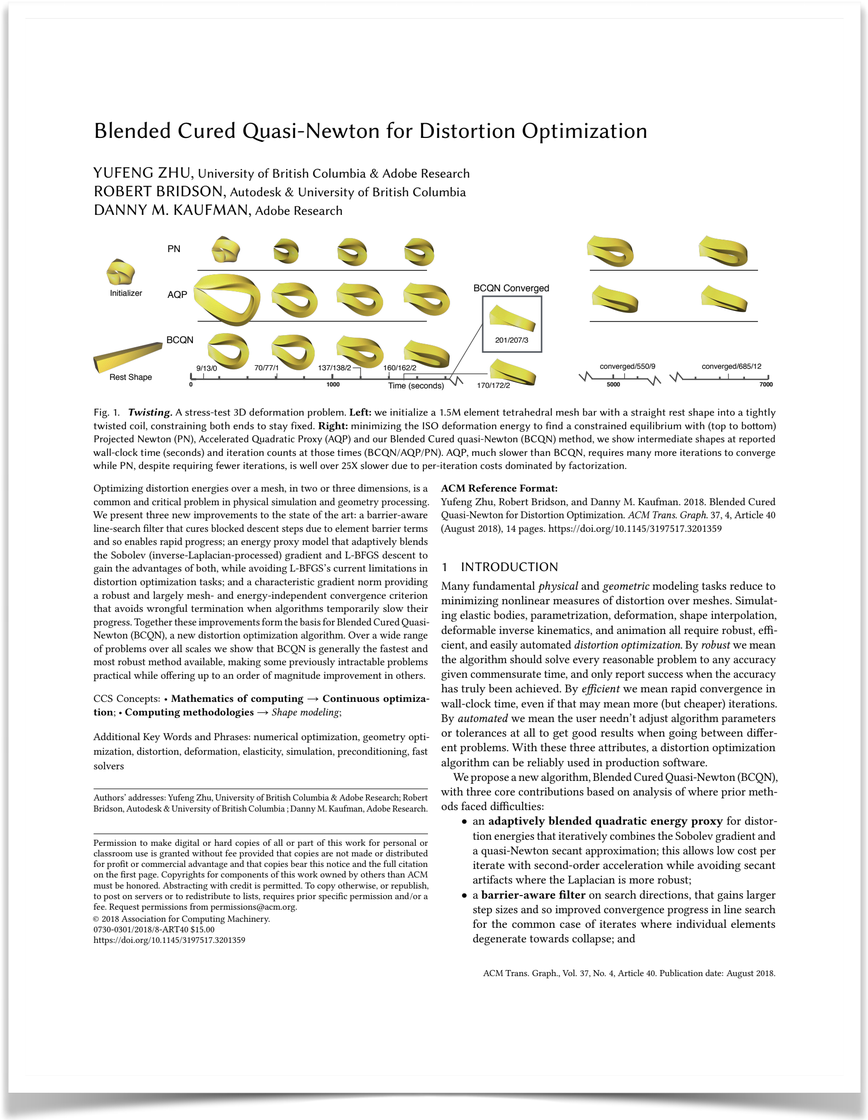
L-BFGS

Use only the last **m** pairs of $\{\mathbf{y}_k, \mathbf{s}_k\}$ and $\mathbf{H}_0^{-1}$

PyTorch

SciPy

Sobolev Gradient 🤝 L-BFGS



Blended Cured Quasi-Newton for Distortion Optimization

YUFENG ZHU, University of British Columbia & Adobe Research
ROBERT BRIDSON, Autodesk & University of British Columbia
DANNY M. KAUFMAN, Adobe Research

Fig. 1. ***Twisting.*** A stress-test 3D deformation problem. **Left:** we initialize a 1.5M element tetrahedral mesh bar with a straight rest shape into a tightly twisted coil, constraining both ends to stay fixed. **Right:** minimizing the ISO deformation energy to find a constrained equilibrium with (top to bottom) Projected Newton (PN), Accelerated Quadratic Proxy (AQP) and our Blended Cured quasi-Newton (BCQN) method, we show intermediate shapes at reported wall-clock time (seconds) and iteration counts at those times (BCQN/AQP/PN). AQP, much slower than BCQN, requires many more iterations to converge while PN, despite requiring fewer iterations, is well over 25X slower due to per-iteration costs dominated by factorization.

Optimizing distortion energies over a mesh, in two or three dimensions, is a common and critical problem in physical simulation and geometry processing. We present three new improvements to the state of the art: a barrier-aware line-search filter that cures blocked descent steps due to element barrier terms and so enables rapid progress; an energy proxy model that adaptively blends the Sobolev (inverse-Laplacian-processed) gradient and L-BFGS descent to gain the advantages of both, while avoiding L-BFGS's current limitations in distortion optimization tasks; and a characteristic gradient norm providing a robust and largely mesh- and energy-independent convergence criterion that avoids wrongful termination when algorithms temporarily slow their progress. Together these improvements form the basis for Blended Cured Quasi-Newton (BCQN), a new distortion optimization algorithm. Over a wide range of problems over all scales we show that BCQN is generally the fastest and most robust method available, making some previously intractable problems practical while offering up to an order of magnitude improvement in others.

CCS Concepts: • **Mathematics of computing** → **Continuous optimization;** • **Computing methodologies** → *Shape modeling*;

Additional Key Words and Phrases: numerical optimization, geometry optimization, distortion, deformation, elasticity, simulation, preconditioning, fast solvers

Authors' addresses: Yufeng Zhu, University of British Columbia & Adobe Research; Robert Bridson, Autodesk & University of British Columbia ; Danny M. Kaufman, Adobe Research.

0730-0301/2018/8-ART40 $15.00
https://doi.org/10.1145/3197517.3201359

ACM Reference Format:
Yufeng Zhu, Robert Bridson, and Danny M. Kaufman. 2018. Blended Cured Quasi-Newton for Distortion Optimization. *ACM Trans. Graph.* 37, 4, Article 40 (August 2018), 14 pages. https://doi.org/10.1145/3197517.3201359

1 INTRODUCTION

Many fundamental *physical* and *geometric* modeling tasks reduce to minimizing nonlinear measures of distortion over meshes. Simulating elastic bodies, parametrization, deformation, shape interpolation, deformable inverse kinematics, and animation all require robust, efficient, and easily automated *distortion optimization*. By *robust* we mean the algorithm should solve every reasonable problem to any accuracy given commensurate time, and only report success when the accuracy has truly been achieved. By *efficient* we mean rapid convergence in wall-clock time, even if that may mean more (but cheaper) iterations. By *automated* we mean the user needn't adjust algorithm parameters or tolerances at all to get good results when going between different problems. With these three attributes, a distortion optimization algorithm can be reliably used in production software.

We propose a new algorithm, Blended Cured Quasi-Newton (BCQN), with three core contributions based on analysis of where prior methods faced difficulties:

- an **adaptively blended quadratic energy proxy** for distortion energies that iteratively combines the Sobolev gradient and a quasi-Newton secant approximation; this allows low cost per iterate with second-order acceleration while avoiding secant artifacts where the Laplacian is more robust;
- a **barrier-aware filter** on search directions, that gains larger step sizes and so improved convergence progress in line search for the common case of iterates where individual elements degenerate towards collapse; and

Low-rank update

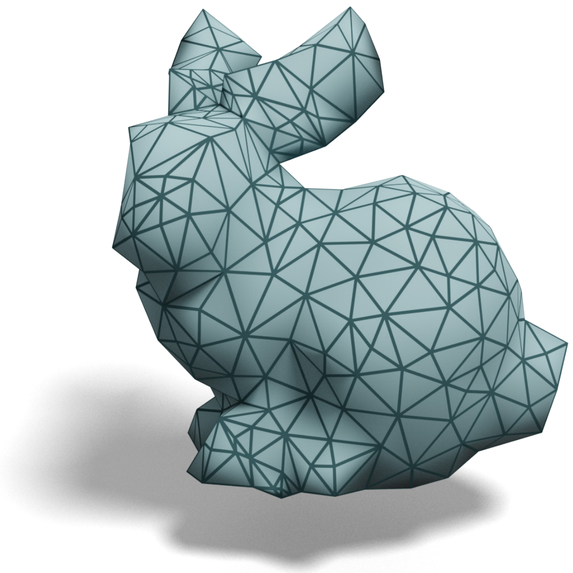
$$\{\mathbf{z}_k, \mathbf{s}_k\} \text{ and } \mathbf{L}^{-1}$$

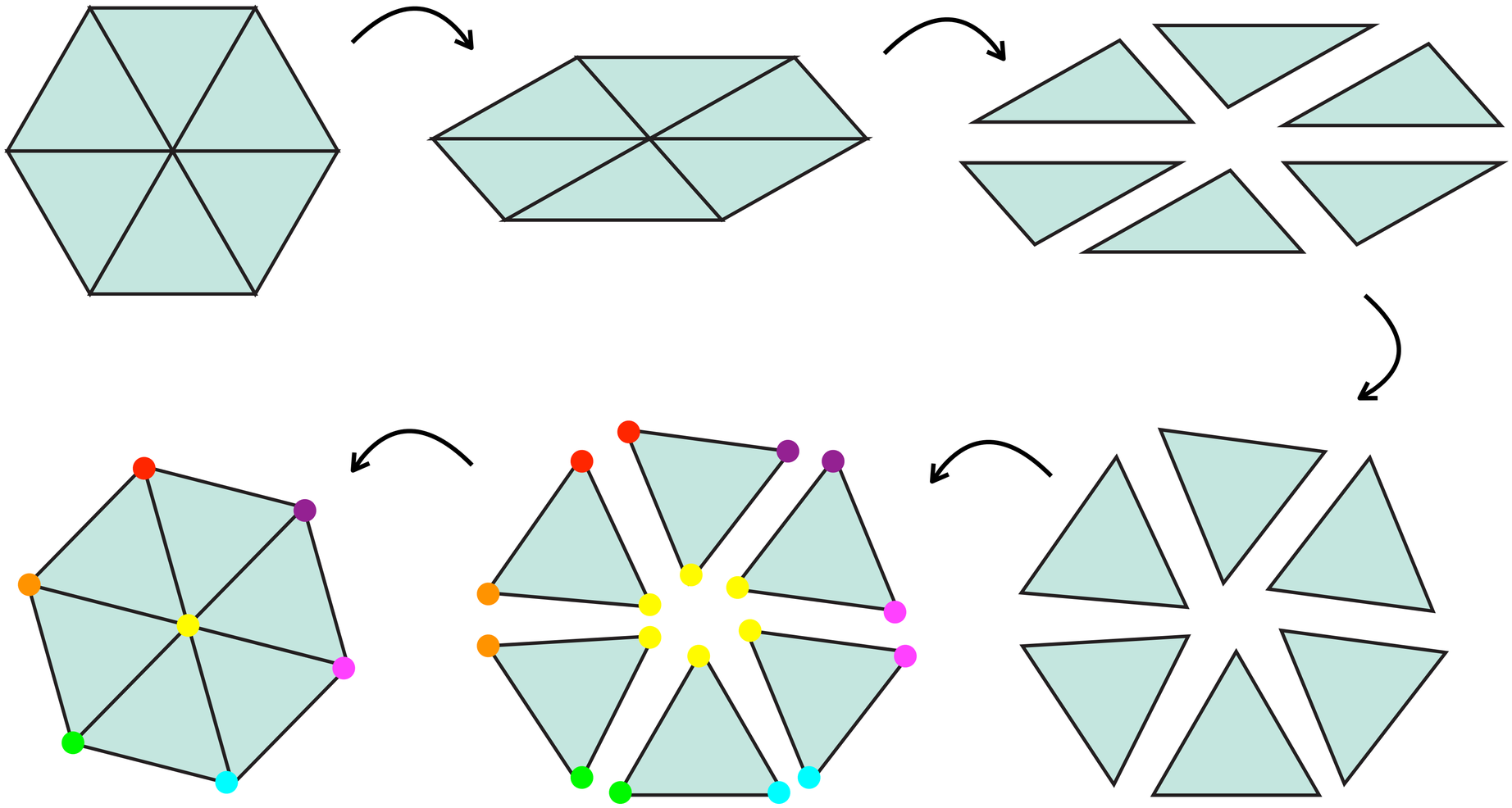
$$\mathbf{z}_k = (1-\beta_k)\mathbf{y}_k + \beta_k \mathbf{L}\mathbf{s}_k$$

$$\mathbf{y}_k = \mathbf{g}_{k+1} - \mathbf{g}_k$$

$$\mathbf{s}_k = \mathbf{x}_{k+1} - \mathbf{x}_k$$

Strategy 2: Exploit the Locality



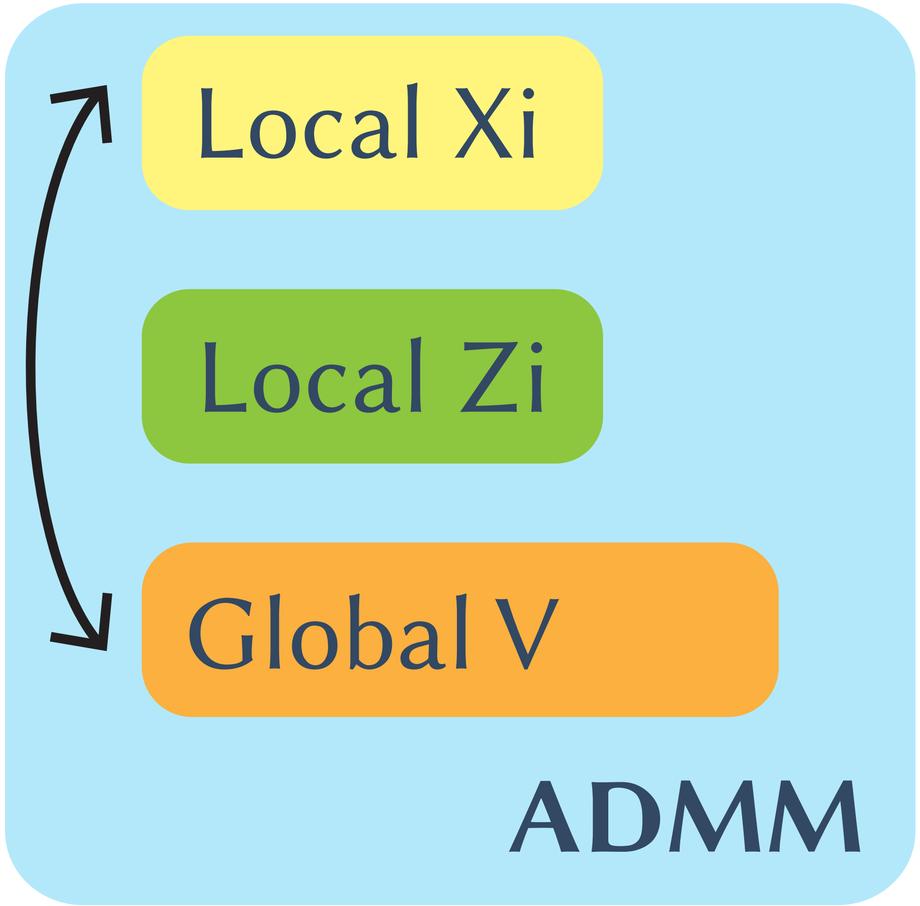


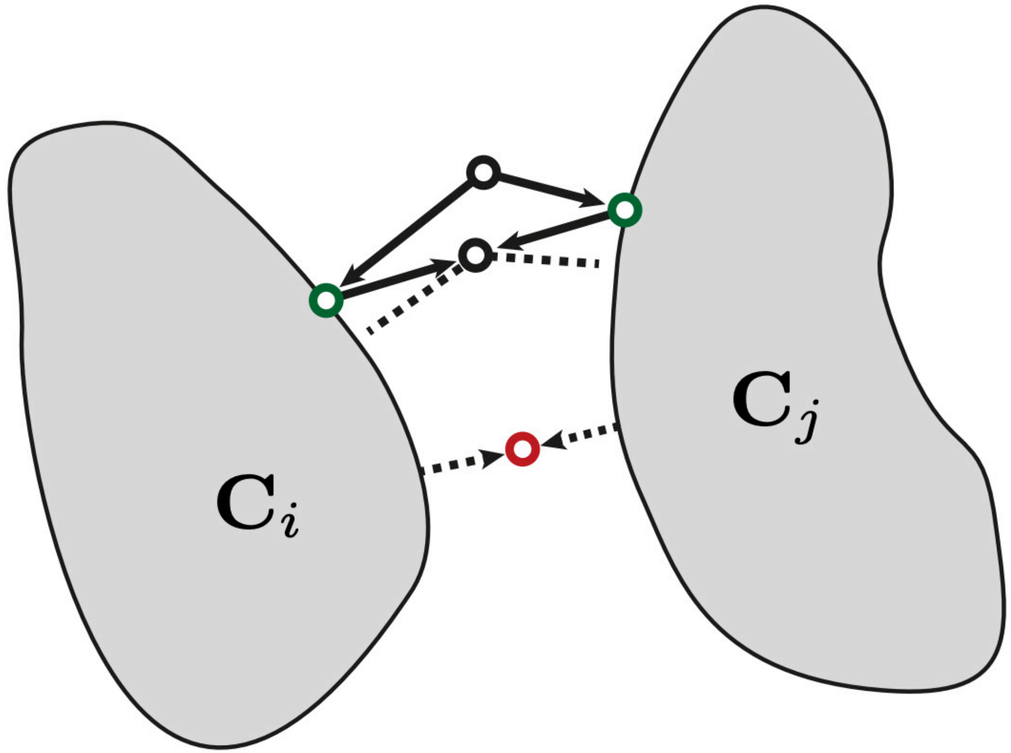
ADMM / Local-Global Solve / Projective Dynamics

$$\underset{\mathbf{v},\{\mathbf{x}_i\}}{\text{minimize}} \sum_{i \in V}$$

Elasticity + (Optional) **Other**

ADMM / Local-Global Solve / Projective Dynamics





Block Coordinate Descent

Vertex Block Descent

ANKA HE CHEN, University of Utah, USA
ZIHENG LIU, University of Utah, USA
YIN YANG, University of Utah, USA
CEM YUKSEL, University of Utah & Roblox, USA

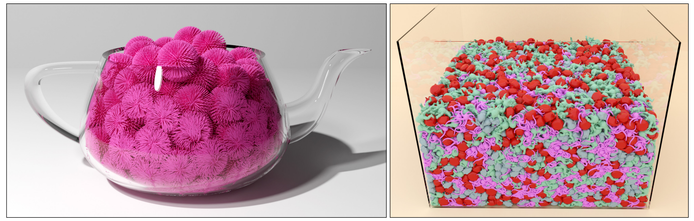


Fig. 1. *Example simulation results using our solver, both of those methods involve more than 100 million DoFs and 1 million active collisions.*

We introduce vertex block descent, a block coordinate descent solution for the variational form of implicit Euler through vertex-level Gauss-Seidel iterations. It operates with local vertex position updates that achieve reductions in global variational energy with maximized parallelism. This forms a physics solver that can achieve numerical convergence with unconditional stability and exceptional computation performance. It can also fit in a given computation budget by simply limiting the iteration count while maintaining its stability and superior convergence rate.

We present and evaluate our method in the context of elastic body dynamics, providing details of all essential components and showing that it outperforms alternative techniques. In addition, we discuss and show examples of how our method can be used for other simulation systems, including particle-based simulations and rigid bodies.

CCS Concepts: • **Computing methodologies** → **Physical simulation**; *Collision detection.*

Additional Key Words and Phrases: physics-based simulation, elastic body, rigid body, time integration

1 INTRODUCTION

Physics-based simulation is the cornerstone of most graphics applications and the demands from simulation systems to deliver improved stability, accelerated computational performance, and enhanced visual realism are ever-growing. Particularly in real-time graphics applications, the stability and performance requirements are so strict that realism can sometimes be begrudgingly considered of secondary importance.

Notwithstanding the substantial amount of research and groundbreaking discoveries made on physics solvers over the past decades, existing methods still leave some things to be desired. They either deliver high-quality results, but fail to meet the computational demands of many applications or fit in a given computation time by sacrificing quality. Stability, on the other hand, is always a challenge, particularly with strict computation budgets.

In this paper, we introduce *vertex block descent (VBD)*, a physics solver that offers unconditional stability, superior computational performance than prior methods, and the ability to achieve numerical convergence to an implicit Euler integration. Though our method is a general solution that can be used for a variety of simulation problems, we present and evaluate it in the context of elastic body dynamics. Then, we briefly discuss how our method can be applied to some other simulation systems, including particle-based simulations and rigid bodies.

Our VBD method is based on block coordinate descent that performs vertex-based Gauss-Seidel iterations to solve the variational form of implicit Euler. For elastic body dynamics, each iteration runs a loop over the mesh vertices, adjusting the position of a single vertex at a time, temporarily fixing all others. This offers maximized parallelism when coupled with vertex-based mesh coloring, which can achieve an order of magnitude fewer colors (i.e. serialized workloads) as compared to element-based parallelization. Our local position-based updates can ensure that the variational energy is always reduced. Therefore, our method maintains its stability even with a single iteration per time step and large time steps, operating with unconverged solutions containing a large amount of residual. With more iterations, it converges faster than its alternatives. Thus, it can more easily fit in a given computation budget, while maintaining stability with improved convergence.

We present all essential components of using VBD for elastic body dynamics, including formulations for damping, constraints,

Authors' addresses: Anka He Chen, ankachan92@gmail.com, University of Utah, Salt Lake City, UT, USA; Ziheng Liu, NA, University of Utah, Salt Lake City, UT, USA; Yin Yang, NA, University of Utah, Salt Lake City, UT, USA; Cem Yuksel, cem@cemyuksel.com, University of Utah & Roblox, Salt Lake City, UT, USA.

For each vertex (or element):

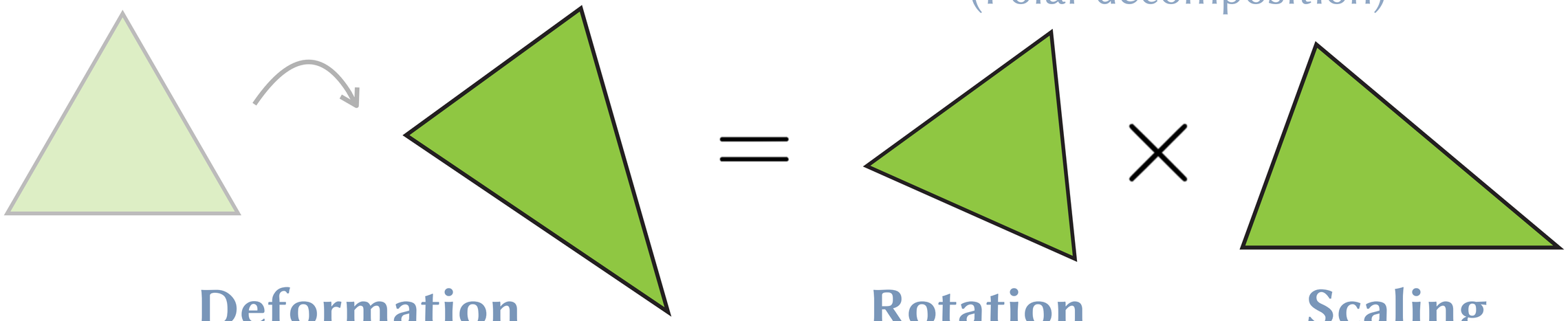
$$\mathbf{d}_i = -\mathbf{g}_i$$

or

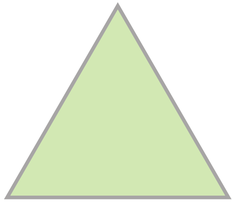
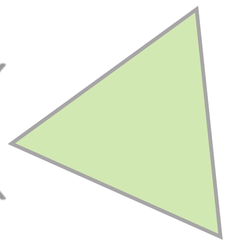
$$\mathbf{d}_i = -\mathbf{H}_i^{-1}\mathbf{g}_i$$

Strategy 3: Make it Rotation-Aware

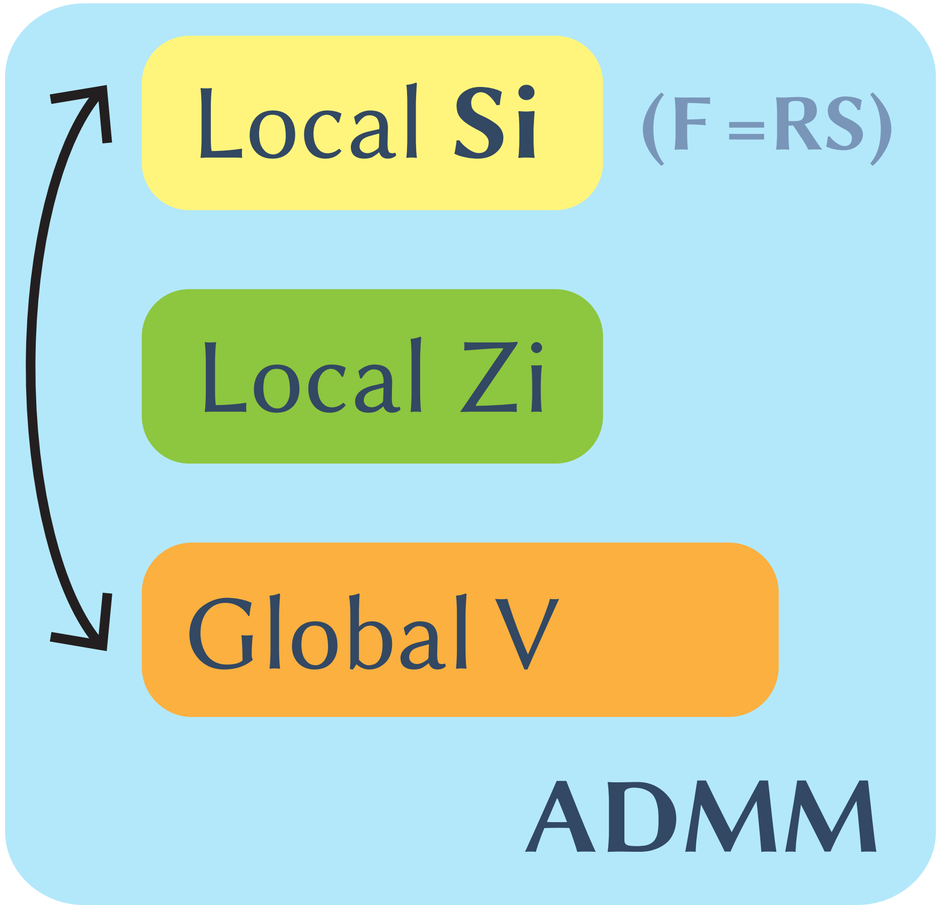
(Polar decomposition)



Deformation **Rotation** **Scaling**

Rotation-invariant: $f($  $) = f($  $)$

Rotation-aware ADMM / Local-Global Solve



WRAPD: Weighted Rotation-aware ADMM for Parameterization and Deformation

GEORGE E. BROWN, University of Minnesota
RAHUL NARAIN, Indian Institute of Technology Delhi

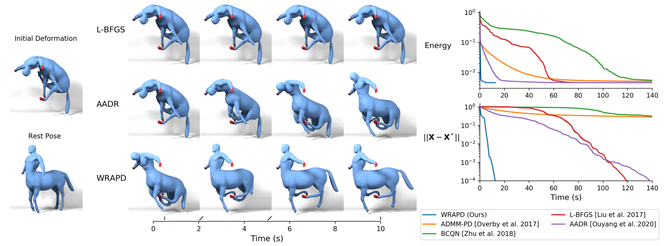


Fig. 1. Quasi-static simulation of a neo-Hookean centaur with 98K tetrahedra and 26K vertices, subject to pin constraints. *Left*: Initial and rest shapes. *Middle*: Selected frames from the first ten seconds of simulation comparing our algorithm to competitors. *Top right*: Objective value vs. time. *Bottom right*: Norm of position error relative to optimal state vs. time.

Local-global solvers such as ADMM for elastic simulation and geometry optimization struggle to resolve large rotations such as bending and twisting modes, and large distortions in the presence of barrier energies. We propose two improvements to address these challenges. First, we introduce a novel local-global splitting based on the polar decomposition that separates the geometric nonlinearity of rotations from the material nonlinearity of the deformation energy. The resulting ADMM-based algorithm is a combination of an L-BFGS solve in the global step and proximal updates of element stretches in the local step. We also introduce a novel method for dynamic reweighting that is used to adjust element weights at runtime for improved convergence. With both improved rotation handling and element weighting, our algorithm is considerably faster than state-of-the-art approaches for quasi-static simulations. It is also much faster at making early progress in parameterization problems, making it valuable as an initializer to jump-start second-order algorithms.

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

Additional Key Words and Phrases: quasi-statics, parameterization, geometric nonlinearity, rotation-aware, reweighting, optimization, ADMM

Authors' addresses: George E. Brown, University of Minnesota, brow2327@umn.edu; Rahul Narain, Indian Institute of Technology Delhi, narain@csc.iitd.ac.in.

0730-0301/2021/8-ART82 $15.00
https://doi.org/10.1145/3450626.3459942

ACM Reference Format:
George E. Brown and Rahul Narain. 2021. WRAPD: Weighted Rotation-aware ADMM for Parameterization and Deformation. *ACM Trans. Graph.* 40, 4, Article 82 (August 2021), 14 pages. https://doi.org/10.1145/3450626.3459942

1 INTRODUCTION

From quasi-static and dynamic simulation of elastic bodies in physics-based animation to interactive shape manipulation and mesh parameterization in geometry processing, recent work has converged on minimization of deformation energies as a central task in computer graphics. The deformation energy model takes many different forms in different applications, from physically-based hyperelastic models such as the neo-Hookean model, to geomerically motivated distortion metrics such as the symmetric Dirichlet energy. These energies can be highly nonlinear with respect to stretching deformations, but have the key property that they are invariant to rigid transformations.

Many popular techniques for minimizing such energies use a local-global approach [Sorkine and Alexa 2007; Bouaziz et al. 2012; Overby et al. 2017; Liu et al. 2017; Peng et al. 2018; Zhang et al. 2019; Ouyang et al. 2020], in which chosen per-element quantities are introduced as additional optimization variables. Optimization is then performed in an alternating fashion, with a local step acting on the per-element local variables and a global step updating the vertex positions. We focus on ADMM-PD [Overby et al. 2017], a local-global algorithm that has been shown to be effective for nonlinear energies

Mixed Variational Finite Elements for Implicit Simulation of Deformables

TY TRUSTY, University of Toronto, Canada
DANNY M. KAUFMAN, Adobe Research, USA
DAVID I.W LEVIN, University of Toronto, Canada

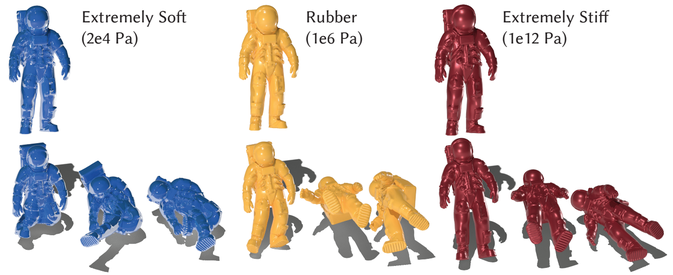


Fig. 1. Our new mixed finite element method (MFEM) can produce simulations of elastica with wildly different materials (including rigid) both accurately and quickly. Our key contribution is that our method is both capable of converging to an accurate solution, matching that of a Newton's method, as well as generate visually plausible results when stopped early. This makes it ideal for a plethora of engineering and graphics applications.

We propose and explore a new method for the implicit time integration of elastica. Key to our approach is the use of a mixed variational principle. In turn, its finite element discretization leads to an efficient and accurate sequential quadratic programming solver with a superset of the desirable properties of many previous integration strategies. This framework fits a range of elastic constitutive models and remains stable across a wide span of time step sizes and material parameters (including problems that are approximately rigid). Our method exhibits convergence on par with full Newton type solvers and also generates visually plausible results in just a few iterations comparable to recent fast simulation methods that do not converge. These properties make it suitable for both offline accurate simulation and performant applications with expressive physics. We demonstrate the efficacy of our approach on a number of simulated examples.

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

Additional Key Words and Phrases: physics-based animation, physics simulation

ACM Reference Format:
Ty Trusty, Danny M. Kaufman, and David I.W Levin. 2022. Mixed Variational Finite Elements for Implicit Simulation of Deformables. In *SIGGRAPH Asia 2022 Conference Papers (SA '22 Conference Papers), December 6–9, 2022, Daegu, Republic of Korea.* ACM, New York, NY, USA, 8 pages. https://doi.org/10.1145/3550469.3555418

SA '22 Conference Papers, December 6–9, 2022, Daegu, Republic of Korea
2022. ACM ISBN 978-1-4503-9470-3/22/12…$15.00
https://doi.org/10.1145/3550469.3555418

1 INTRODUCTION

In this paper we explore the use of a mixed variational principle to build an efficient and general-purpose simulation algorithm for the physics-based animation of elastica.

Standard approaches for the implicit time integration of continua discretize with finite differences in time and finite elements in space. Recent methods often leverage the observation that, for these implicit time integration choices, each individual time step solve can then be cast as a minimization problem. In turn, the applied strategy for solving these optimization problems then leads to a wide range of well-known simulation algorithms [Li et al. 2019]. For example, a "standard" finite element approach involves minimizing an implicit integration energy via Newton's method while solving the bottleneck of inner linear-systems solves either via direct or iterative methods. Extended Position-Based Dynamics replaces standard direct or iterative solvers with iterations (e.g., GS, Jacobi, and/or SOR) acting on the dual variables (constraint forces) while Projective Dynamics and its more recent generalizations apply various forms of ADMM-type solvers to split, augmented Lagrangian forms.

Despite their common variational origin, implicit solvers for elastica exhibit a wide range of features and limitations, and so tradeoffs.

Subspace Mixed Finite Elements for Real-Time Heterogeneous Elastodynamics

Ty Trusty*
University of Toronto
Canada

Otman Benchekroun*
University of Toronto
Canada

Eitan Grinspun
University of Toronto
Canada

Danny M. Kaufman
University of Toronto and Adobe Research
U.S.A.

David I.W. Levin
University of Toronto and NVIDIA
Canada

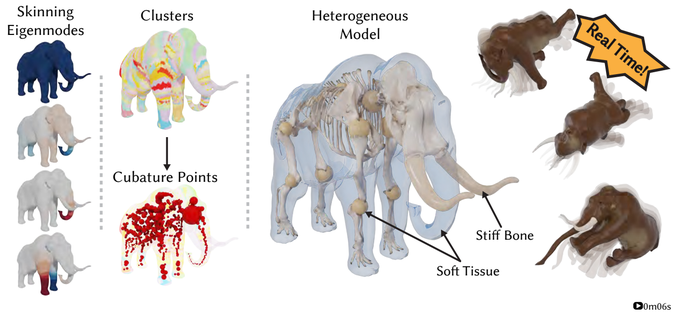


Figure 1: We propose a reduced space mixed finite element method (MFEM) built on a Skinning Eigenmode subspace and material-aware cubature scheme. Our solver is well-suited for simulating scenes with large material and geometric heterogeneities in real-time. This mammoth geometry is composed of 98,175 vertices and 531,565 tetrahedral elements and with a heterogenous composition of widely varying materials of muscles ($E = 5 \times 10^5$ Pa), joints ($E = 1 \times 10^5$ Pa), and bone ($E = 1 \times 10^{10}$ Pa). The resulting simulation runs at 120 frames per second (FPS).

ABSTRACT

Real-time elastodynamic solvers are well-suited for the rapid simulation of homogeneous elastic materials, with high-rates generally enabled by aggressive early termination of timestep solves. Unfortunately, the introduction of strong domain heterogeneities can make these solvers slow to converge. Stopping the solve short creates visible damping artifacts and rotational errors. To address these challenges we develop a reduced mixed finite element solver that preserves rich rotational motion, even at low-iteration regimes. Specifically, this solver augments time-step solve optimizations with auxillary stretch degrees of freedom at mesh elements, and maintains consistency with the primary positional degrees of freedoms at mesh nodes via explicit constraints. We make use of a Skinning Eigenmode subspace for our positional degrees of freedom. We accelerate integration of non-linear elastic energies with a cubature approximation, placing stretch degrees of freedom at cubature points. Across a wide range of examples we demonstrate that this subspace is particularly well suited for heterogeneous material simulation. Our resulting method is a subspace mixed finite element method completely decoupled from the resolution of the mesh that is well-suited for real-time simulation of heterogeneous domains.

*Indicates joint first authors.

Authors' addresses: Ty Trusty*, University of Toronto, Canada; Otman Benchekroun*, University of Toronto, Canada ; Eitan Grinspun, University of Toronto, Canada; Danny M. Kaufman, University of Toronto and Adobe Research, U.S.A.; David I.W. Levin, University of Toronto and NVIDIA, Canada.

KEYWORDS

Mixed FEM, Heterogeneous Materials

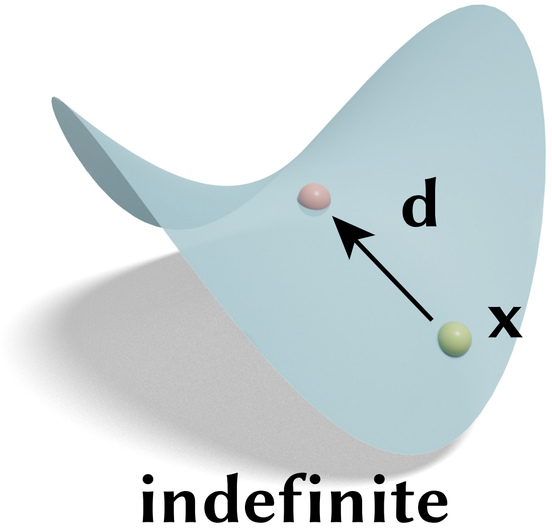
Rotation-aware Newton's Method

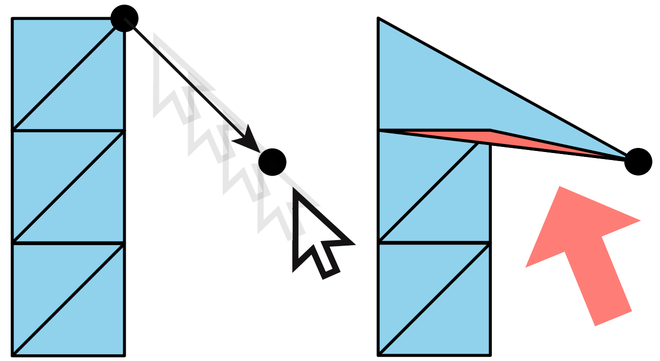
Rotation-aware Subspace

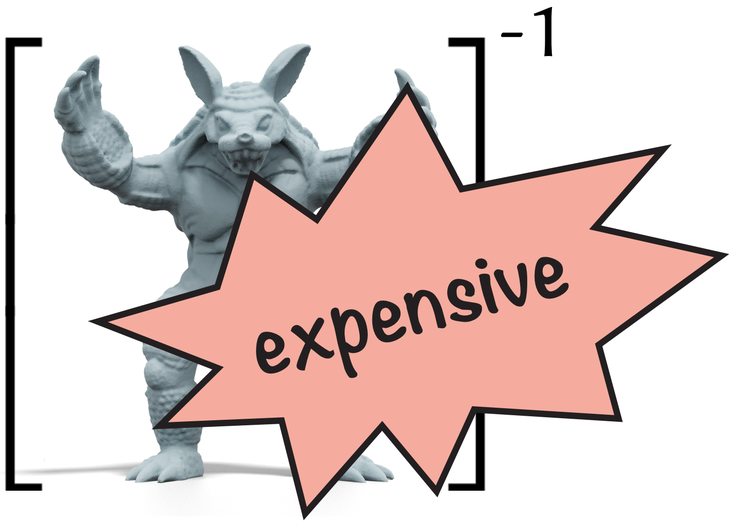
Recap

$$\mathbf{d} = -\mathbf{H}^{-1}\mathbf{g}$$

- Nonconvexity ✓
- Infeasible points ✓
- Large-scale ✓

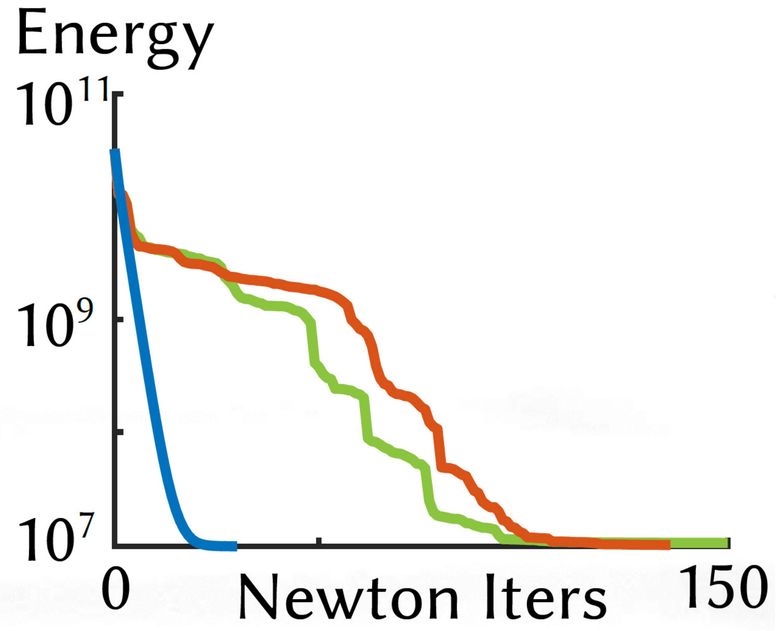






SCA 2024

Thank you!



Introduction to Optimization for Simulation

Honglin Chen

honglin.chen@columbia.edu

https://www.cs.columbia.edu/~honglinchen/