Accelerated Complex-Step Finite Difference for Expedient Deformable Simulation

RAN LUO, The University of New Mexico,
WEIWEI XU, Zhejiang University
TIANJIA SHAO, University of Leeds
HONGYI XU, Google AI
YIN YANG, Clemson University

CLEMSON
SCHOOL OF COMPUTING

Deformable Objects

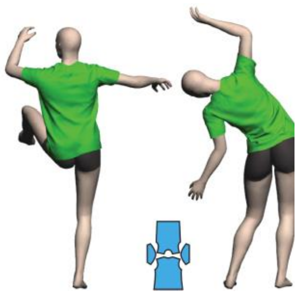
- High-quality deformable simulation is important
- Well-known problem but computational expensive

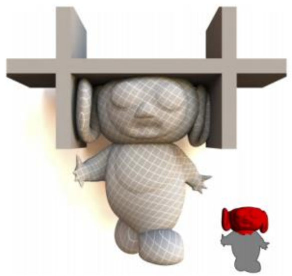
$$\mathbf{M}\left(\mathbf{u}_{n+1} - \mathbf{u}_n - \Delta t \dot{\mathbf{u}}_n\right) = \Delta t^2 \left(\mathbf{f}_{int}\left(\mathbf{u}_{n+1}\right) + \mathbf{f}_{ext}\right)$$

  - Nonlinearity: repeated evaluation of internal force and its gradient


[Zheng & James, 2012]


[Zhao & Barbič, 2013]


[Xu et al, 2014]


[Teng et al, 2015]

…

CLEMSON SCHOOL OF COMPUTING

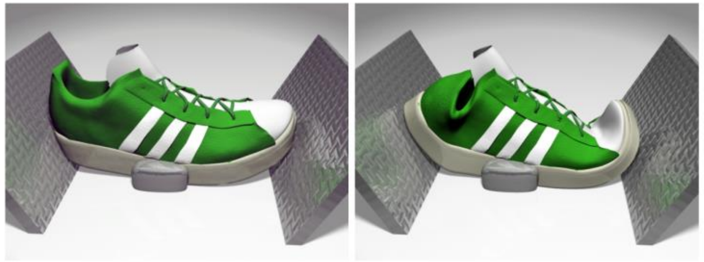
Nonlinearity

- A key challenge is the nonlinearity
  - Largely come from the strain energy
  - Define strain-stress relation just based on the strain energy

$$E_{NH} = \lambda(J-1)^2 + \mu(J^{-2/3} I_1 - 3)$$

$$\text{where} \quad J = |\mathbf{F}|, \quad I_1 = \mathrm{tr}(\mathbf{F}^\top \mathbf{F}), \quad \mathbf{F} = \mathbf{I} + \frac{\partial \mathbf{u}}{\partial \mathbf{x}}$$

- More complicated energies are not uncommon



[Martin et al, 2011]

[Xu et al, 2015]

Will a Numerical Derivative Work?

- The best-known method is finite difference

$$f(x_0+h)=f(x_0)+f'(x_0)\cdot h+\mathbf{O}(h^2)$$

$$\Downarrow$$

$$f'(x_0)\approx\frac{f(x_0+h)-f(x_0)}{h}+\mathbf{O}(h)$$

- Does not work in general due to the **subtractive cancellation**
  - In theory, the smaller perturbation is, the better approximation we obtain
  - In practice, smaller perturbation does not converge, explode at certain

Will a Numerical Derivative Work?

Newton method

Will a Numerical Derivative Work?

Forward difference

Rounding Error

- We have **limited** digits to represent a real number
- Precision depends on how many digits we could allocate

$$a = 1999.99, \qquad \tilde{a} = 1.999 \times 10^3$$

$$E_{round} = \frac{|a - \tilde{a}|}{|a|} = \frac{\left|1999.99 - 1.999 \times 10^3\right|}{|1999.99|} \approx 4.95 \times 10^{-4}$$

- Known as **machine epsilon** ($\sim 1.1 \times 10^{-16}$ for **`double`** precision)
  - However, it is **NOT** the evil of finite difference
  - True problem appears when we have a subtraction between two similar values

Subtractive Cancellation

- Another example

$$a = 1999.99,\ \tilde{a} = 1.999 \times 10^3 \text{ and } b = 1998.88,\ \tilde{b} = 1.998 \times 10^3$$

$$\begin{aligned} E_{subtraction} &= \frac{\left|(\tilde{a} - \tilde{b}) - (a - b)\right|}{|a - b|} \\ &= \frac{\left|(1999.99 - 1.998) \times 10^3 - (1999.99 - 1998.88)\right|}{\left|1999.99 - 1998.88\right|} \\ &= 0.1 \end{aligned}$$

- Subtraction eliminates the first three significant digits
  - Rounding eliminates the least important digit
- Bigger perturbation has bigger approximation error but smaller perturbation leads to subtractive cancellation

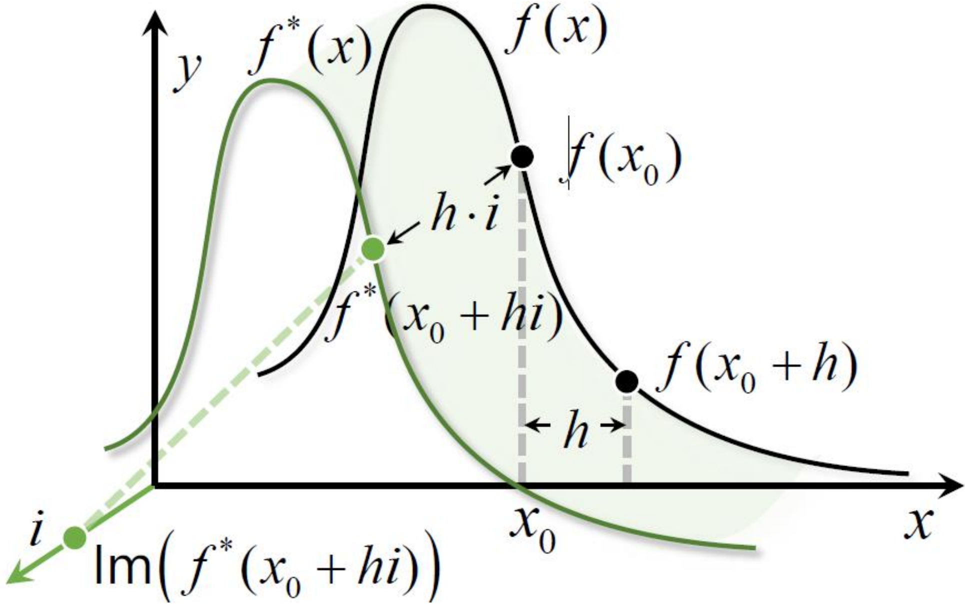
Complex Step Finite Difference (CSFD)

- Apply the perturbation with **complex** Tayler expansion

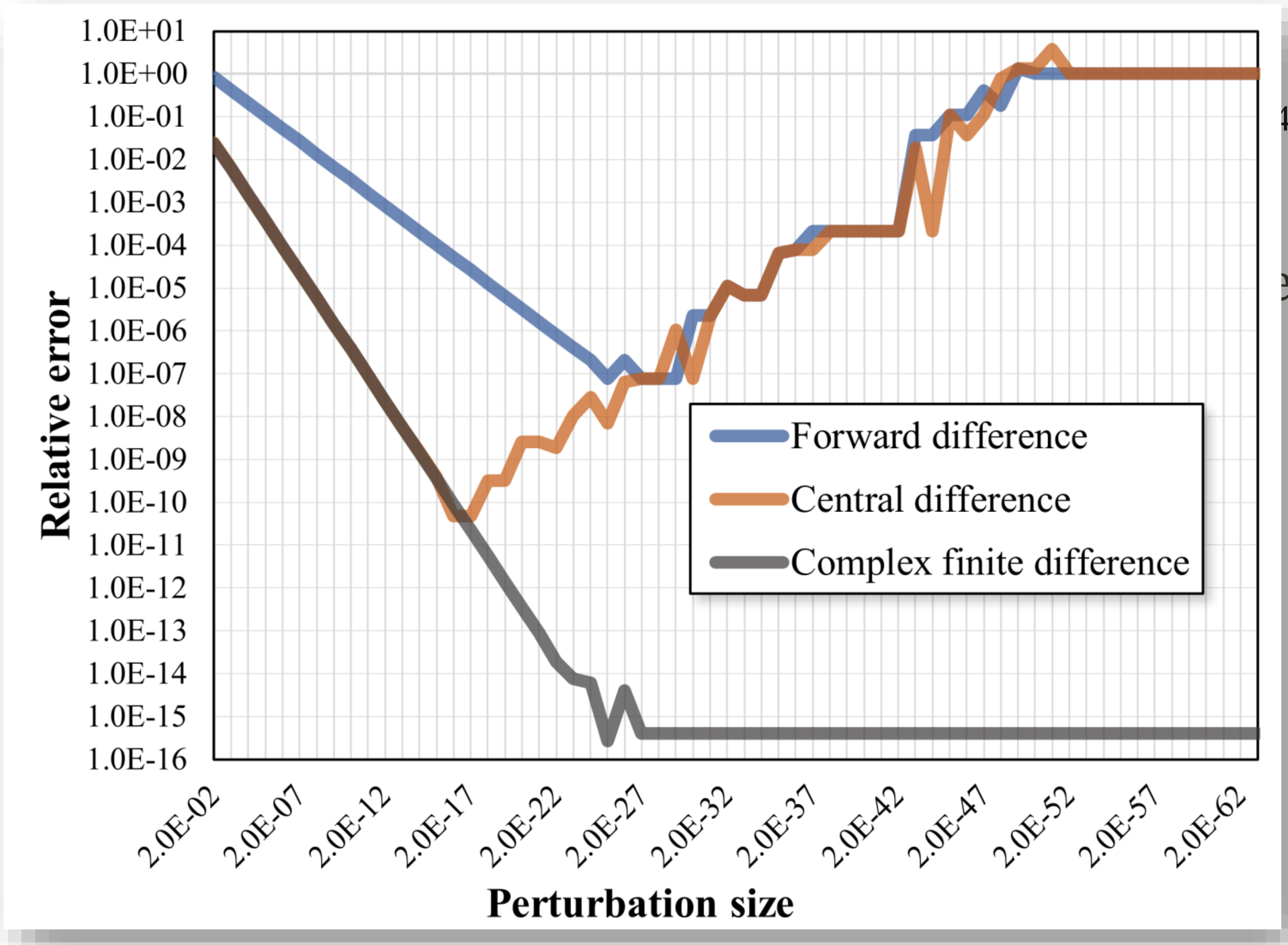
$$f^*(x_0 + hi) = f^*(x_0) + f^{*\prime}(x_0)\cdot(hi) + \mathbf{O}(h^2)$$

$$\mathrm{Im}\Big(f^*(x_0 + hi)\Big) = \mathrm{Im}\Big(f^*(x_0) + f^{*\prime}(x_0)\cdot(hi)\Big) + \mathbf{O}(h^3)$$

$$f'(x_0) = \frac{\mathrm{Im}\Big(f^*(x_0 + hi)\Big)}{h} + \mathbf{O}(h^2)$$



Complex Step Finite Difference (CSFD)



40) to fully

order of

$$f(x) = \frac{e^x}{x^4 + x^2 + 1}$$

$$f'(x) = \frac{x^4 - 4x^3 + x^2 - 2x + 1}{x^4 + x^2 + 1} e^x$$

CLEMSON
SCHOOL OF COMPUTING

Too Good to Be True?

- Promoting the original real function to a complex function
  - Can be easily implemented especially given a good complex library
  - It could also be expensive (orders of magnitude in some cases)

$$f(x_0) = x_0^{\frac{1}{m}} \rightarrow f^*(x_0 + hi) = r^{\frac{1}{m}} \left(\cos \frac{\phi + 2\pi k}{m} + \sin \frac{\phi + 2\pi k}{m} i \right)$$

- CSFD can be accelerated
  - A full complex promotion is a waste (only imaginary part is needed)
  - Perturbation is a very small value, can be treated as an infinitesimal
  - Higher order *h* term can be discarded

An Example

- Full promotion

$$f(x_0) = x_0^{1/m} \rightarrow f^*(x_0 + hi) = r^{1/m}\left(\cos\frac{\phi + 2\pi k}{m} + \sin\frac{\phi + 2\pi k}{m} i\right)$$

- Imaginary only

$$\frac{\operatorname{Im}\left(f^*(x_0 + hi)\right)}{h} = \frac{1}{h}\left(r^{1/m} \cdot \sin\frac{\phi}{m}\right)$$

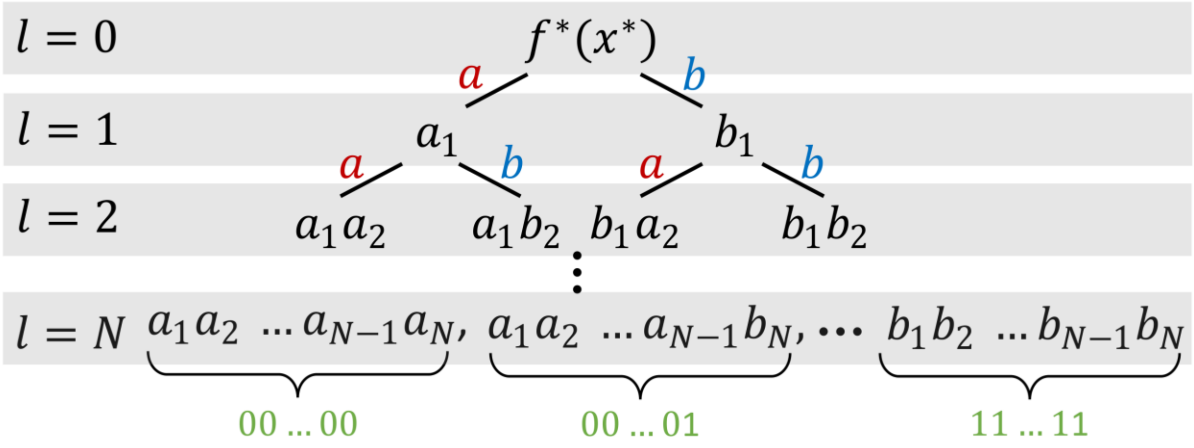
- By treating *h* as an infinitesimal

$$\frac{1}{h}\left(r^{1/m} \cdot \sin\frac{\phi}{m}\right) = \frac{1}{h}\left(r^{1/m} \cdot \frac{\phi}{m}\right) = \frac{1}{h}\left(r^{1/m} \cdot \frac{h}{rm}\right) = \frac{r^{1/m}}{rm}$$

Full promotion	13.1 sec
Imaginary only	9.49 sec
Infinitesimal trick	0.056 sec (233X)
Analytic	0.064 sec

Composite Functions

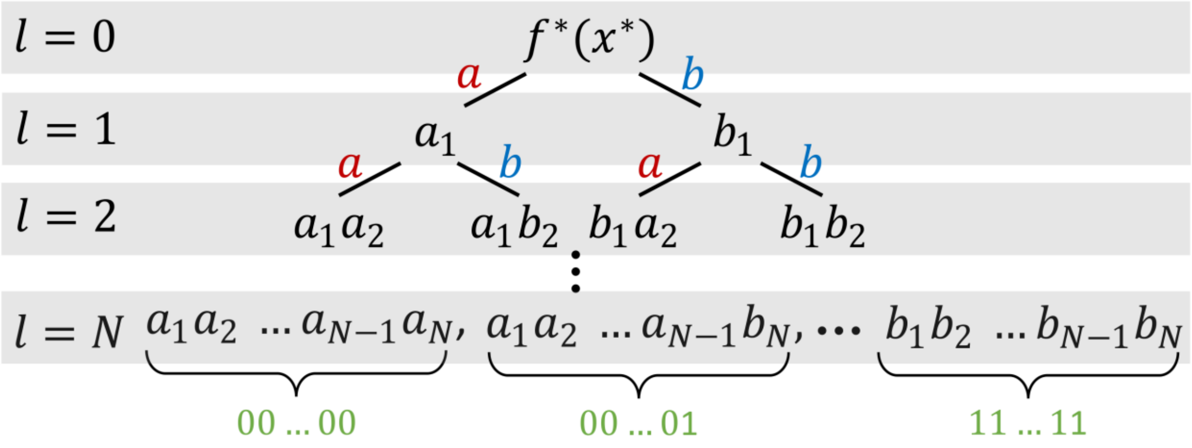
- Functions in real-world applications are complicated involving
  - A chain of binary operators $f(x) = f_1(x) \circ f_2(x) \circ \cdots \circ f_k(x)$
  - Nested operators $f(x) = f_1(f_2(f_3(\cdots)))$
- The key idea: isolate the propagation of imaginary perturbation
- Let $f_1(x) = a_1 + b_1$ etc., we have $f_1(x) \cdot f_2(x) = (a_1 + b_1)a_2 + (a_1 + b_1)b_2$
  - Each time a new f_k comes, number of addends doubles with an a_k and a b_k appended



Complicated Functions (cont.)

- The path to leaf determines the shape of an addend
- An a_k is the real part value and b_k is the perturbation (imaginary)
  - We do not need any leaves with more than two b_k
  - We could pre-compute product of all a_k

$$\boxed{A = a_1 \cdot a_2 \cdot a_3 \quad a_1 a_2 b_3 = \frac{A}{a_3} b_2}$$



Higher-order Derivative & Tensor

- Multi-complex number, recursively defined

$$\mathbb{C}^n = \left\{ z_1 + z_2 i_n, \quad z_1, z_2 \in \mathbb{C}^{n-1} \right\}$$

- We can have MC Tayler expansion, and MSCFD becomes

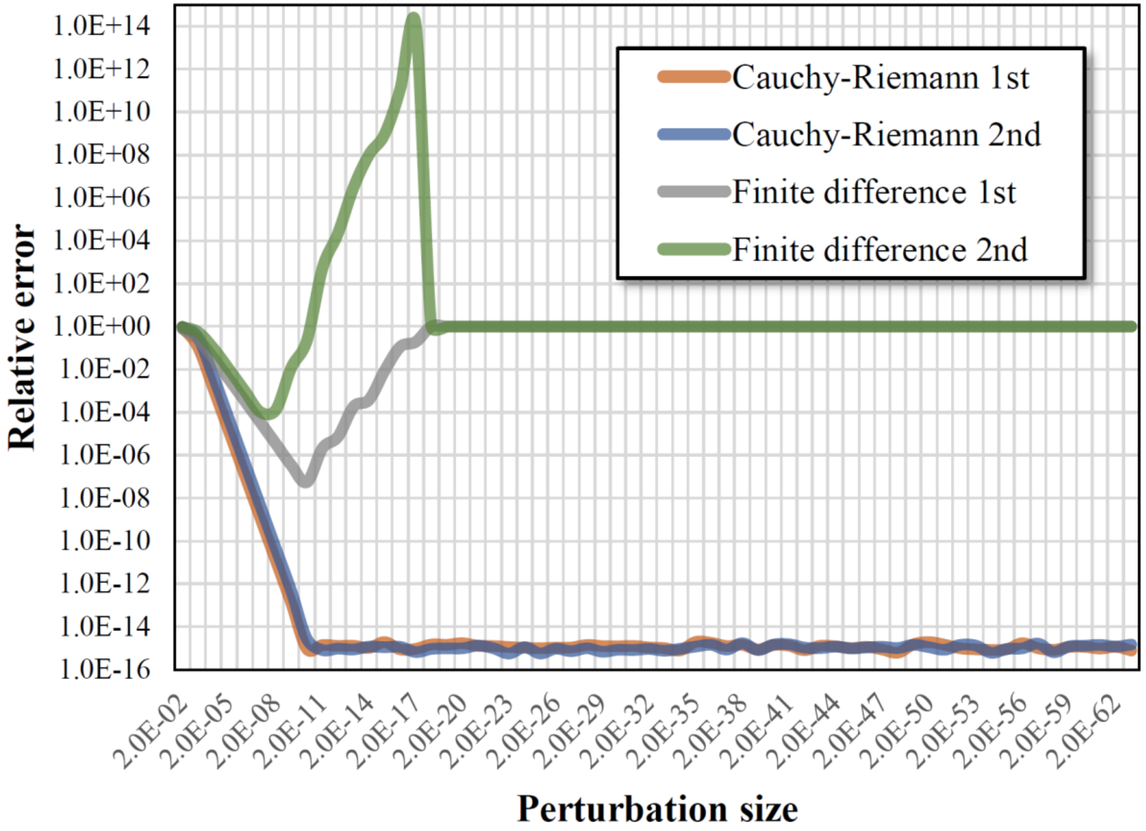
$$f^{(n)}(x) = \frac{\mathrm{Im}\left(f^{\star}(x_0 + hi_1 + hi_2 + \cdots + hi_n) \right)}{h^n} + \mathbf{O}(h^2)$$

- Cauchy-Riemann formulation extends CSFD/MSCFD to tensor functions

$$z^n = z_1^{n-1} + z_2^{n-1} i_n = \begin{bmatrix} z_1^{n-1} & -z_2^{n-1} \\ z_2^{n-1} & z_1^{n-1} \end{bmatrix}$$

CLEMSON SCHOOL OF COMPUTING

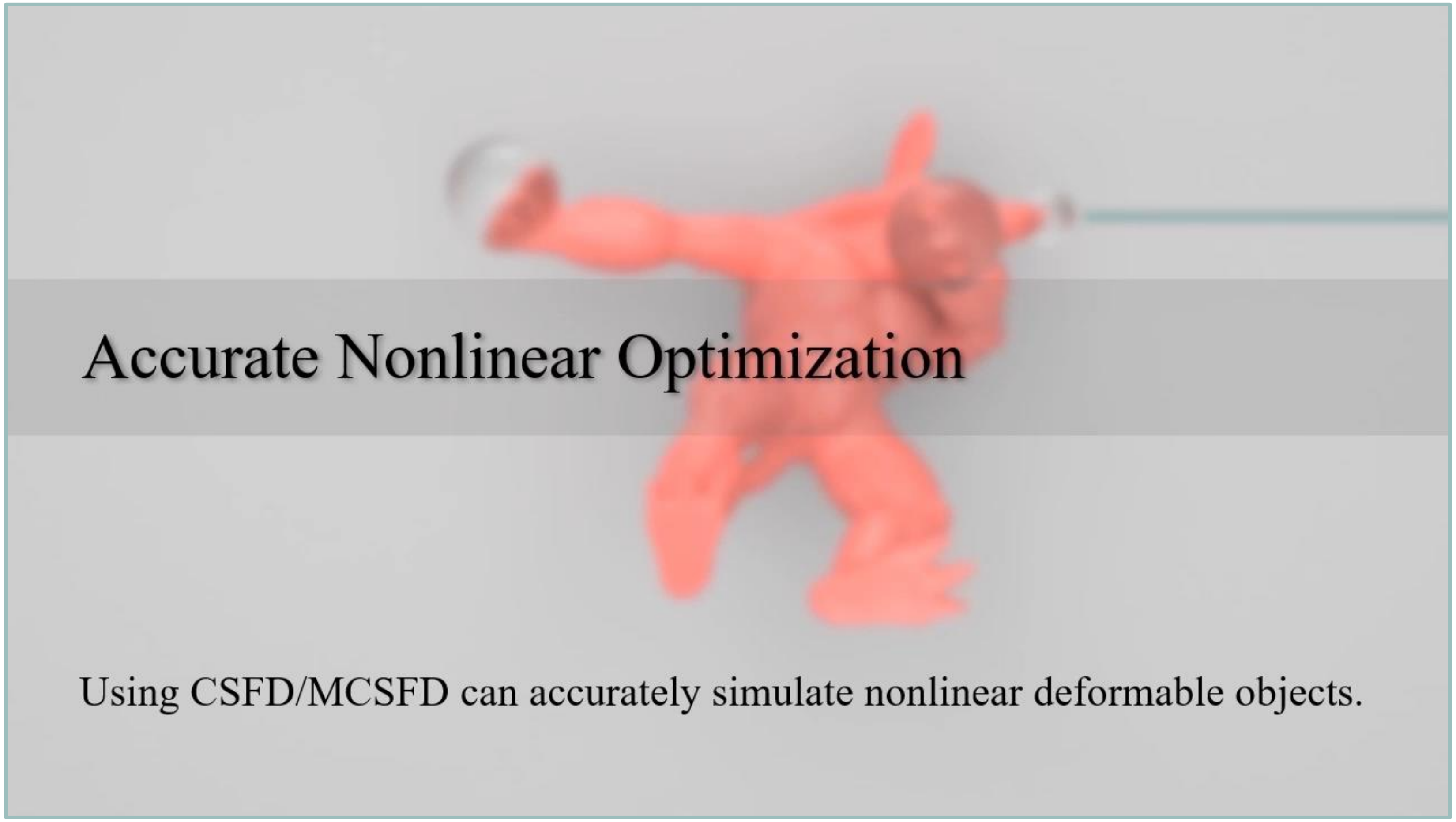
Derivative of Matrix Inverse



$$\frac{\partial f}{\partial X_{2,2}} = -\mathbf{X}^{-1}\frac{\partial \mathbf{X}}{\partial X_{2,2}}\mathbf{X}^{-1}$$

$$\frac{\partial^2 f}{\partial X_{2,2}^2} = -2\mathbf{X}^{-1}\frac{\partial \mathbf{X}}{\partial X_{2,2}}\mathbf{X}^{-1}\frac{\partial \mathbf{X}}{\partial X_{2,2}}\mathbf{X}^{-1}$$

Highly Accurate in Simulation

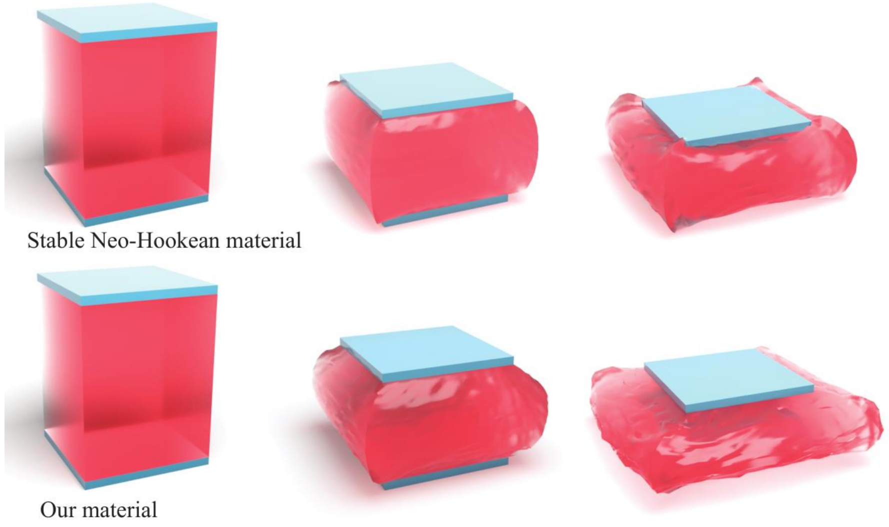


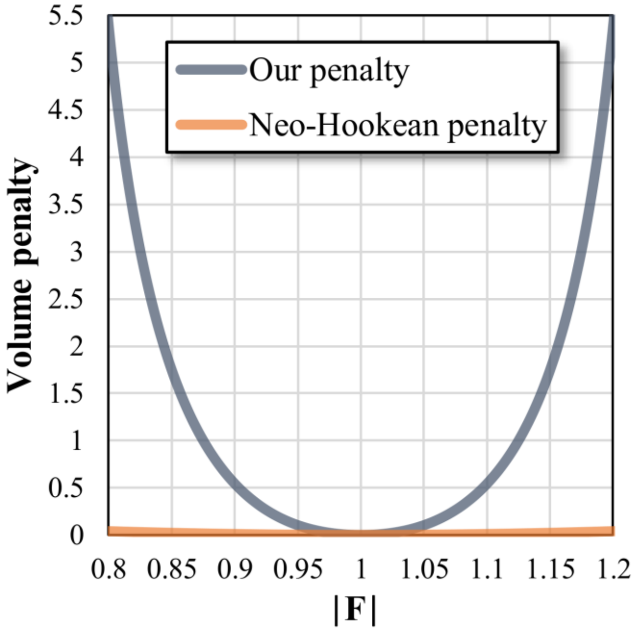
CLEMSON
SCHOOL OF COMPUTING

Intuitive Hyperelastic Simulation

- Hyperelastic models with minor implementation efforts
- CSFD/MSCFD further enables us to design customized energy

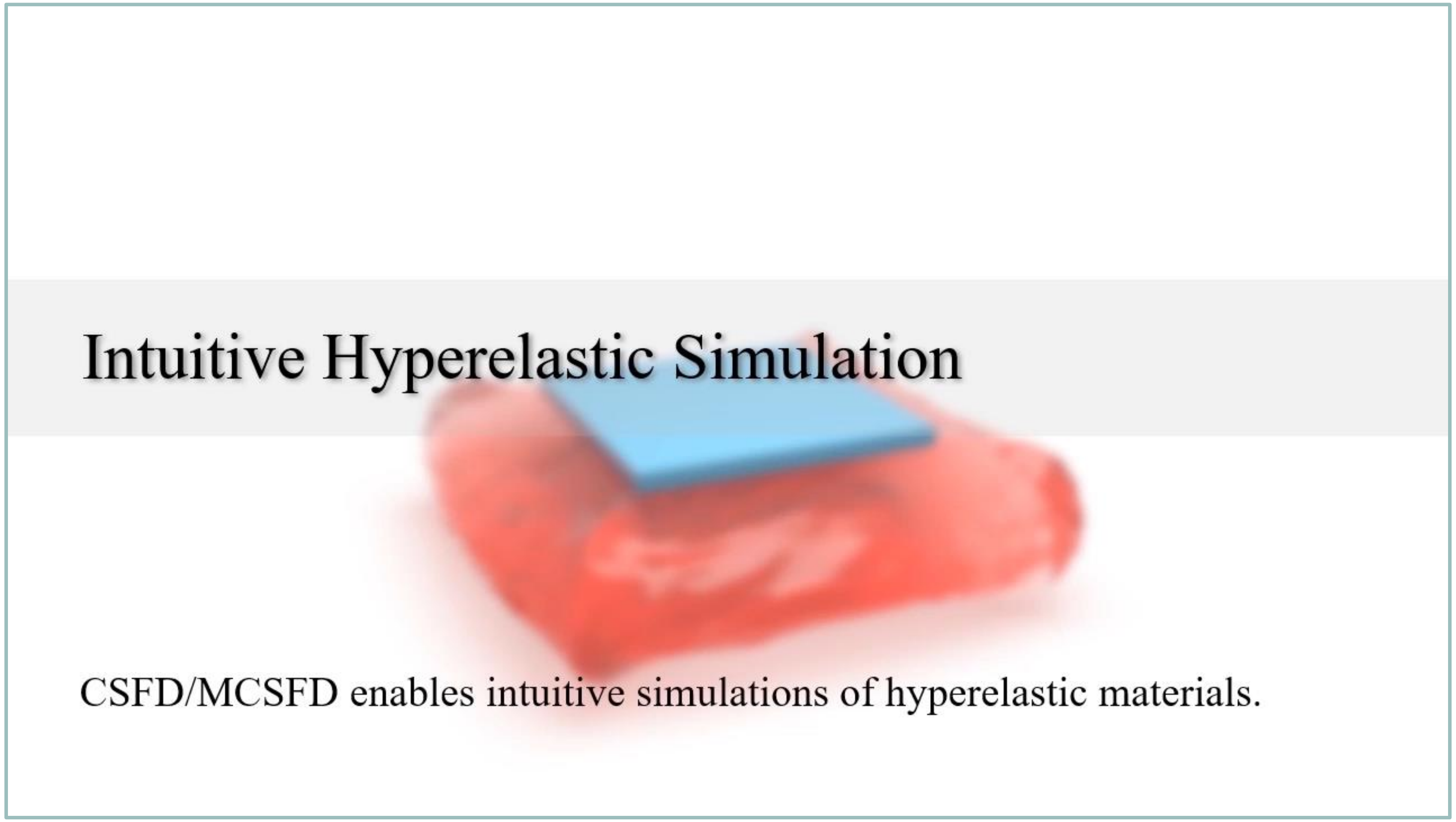
$$E_{volume} = \mu(J^{-2/3} I_C - 3) + \frac{\lambda}{2}\log\left(1-4(J-1)^2\right)$$



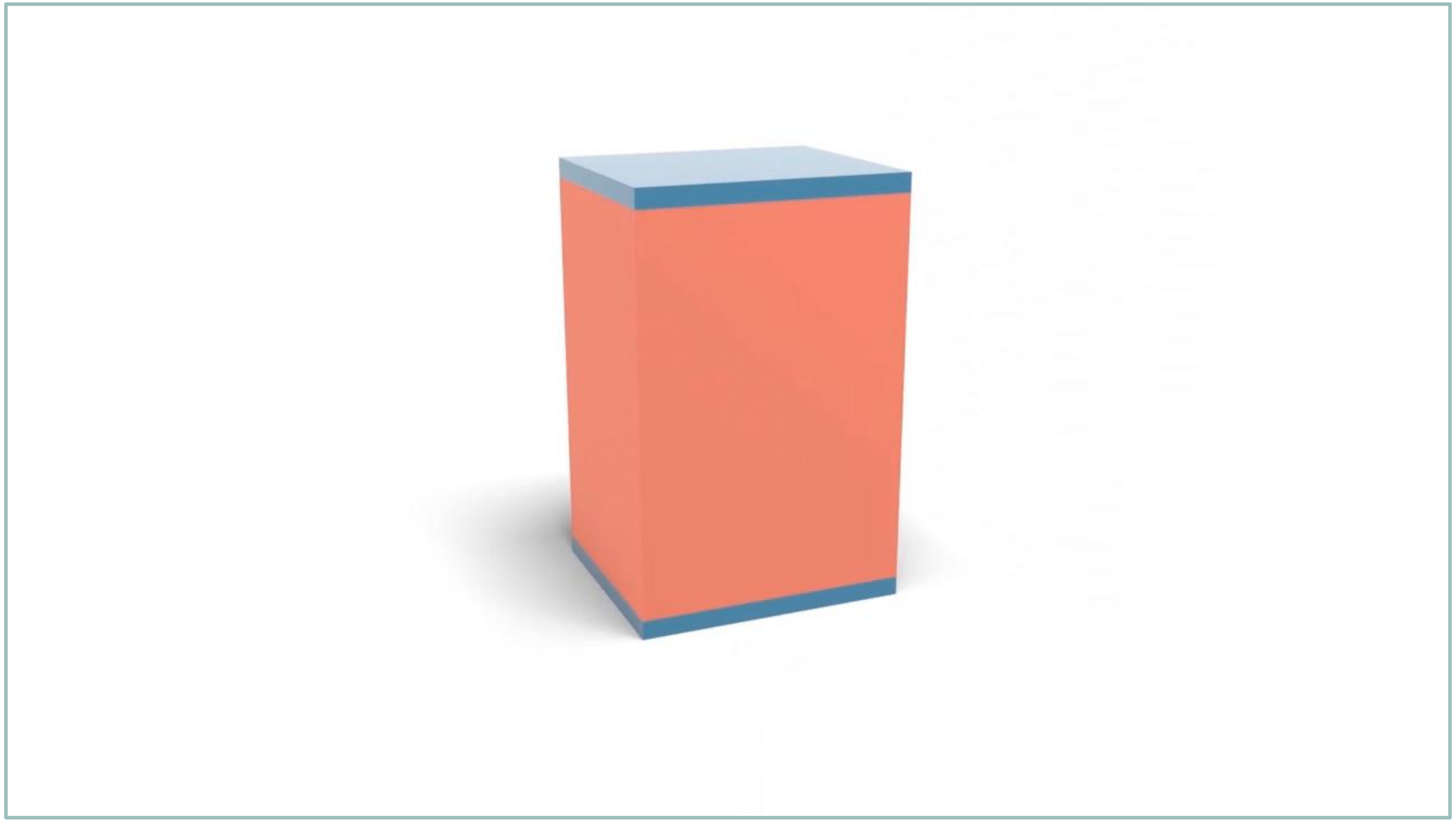


CLEMSON SCHOOL OF COMPUTING

Intuitive Hyperelastic Simulation

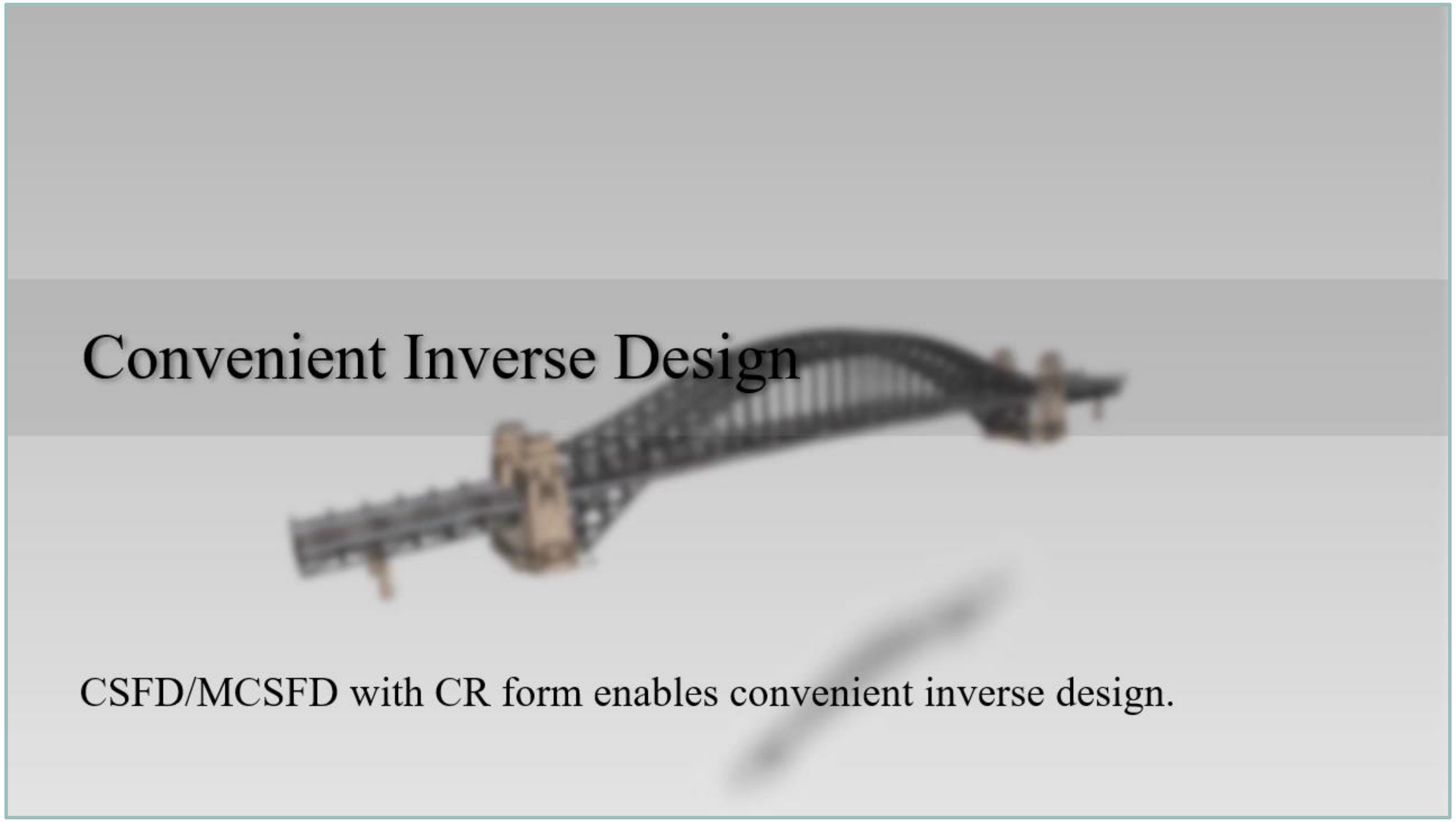


More Customized Energy



Model Derivative for All

Convenient Inverse Design



Thank You

https://github.com/lrquad/MCSFD

We are hiring ☺

CLEMSON
SCHOOL OF COMPUTING