COL781: Computer Graphics

32. Mass-Spring Systems

World of Goo

Assignment 4 notes

Late days increased to 6

Starter code provided to support deforming meshes (for cloth simulation):
https://git.iitd.ac.in/col781-2302/a4

Time integration recap

$$\frac{\mathrm{d}}{\mathrm{d}t} y(t) = \phi(y(t), t)$$

Forward Euler:

$$y_{n+1} = y_n + \phi(y_n, t_n)\,\Delta t$$

Backward Euler:

$$y_{n+1} \approx y_n + \phi(y_{n+1}, t_{n+1})\,\Delta t$$

For finite Δt, these obviously have some approximation error. How much?

Taylor series:

$$y(t) = y(0) + y'(0)\, t + \tfrac{1}{2}\, y''(0)\, t^2 + \cdots$$

In forward Euler, we approximate $y(\Delta t) \approx y(0) + y'(0)\, \Delta t$.

- Error per time step: **local truncation error** = $O(\Delta t^2)$

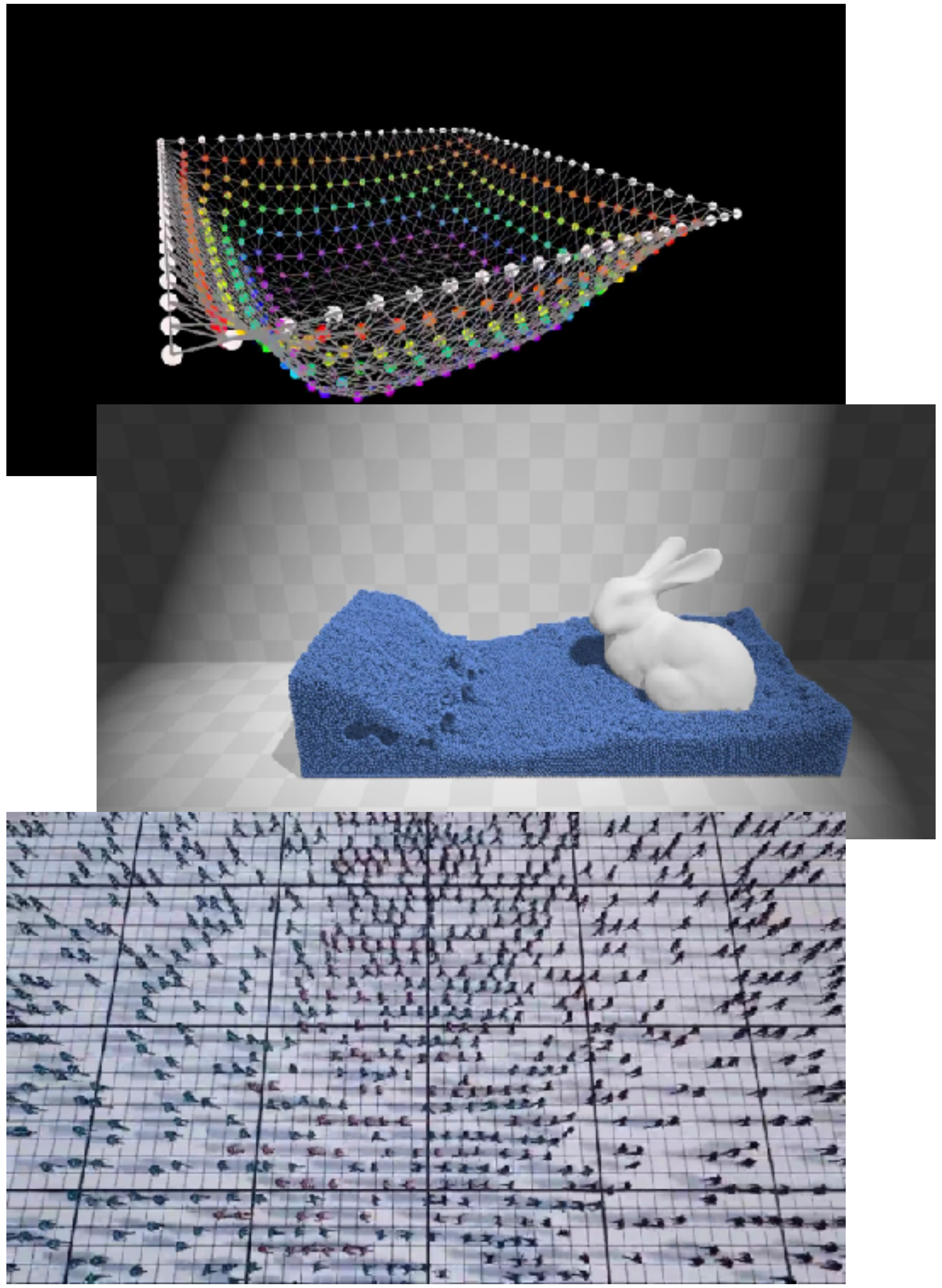
To reach time t, we will need $t/\Delta t$ steps

- Total error in $y(t)$: **global truncation error** $\approx t/\Delta t\; O(\Delta t^2) = O(\Delta t)$

So we say forward Euler is first-order accurate. Same is true for backward Euler!

Schemes with higher-order accuracy: trapezoid, midpoint (2nd order), RK4 (4th order), …

How to apply all this to systems of interacting particles?



Forward and semi-implicit Euler are easy:

- For each particle i, compute total force $\mathbf{f}_i^n$
- For each particle i, compute new state

$$\mathbf{v}_i^{n+1} = \mathbf{v}_i^n + m_i^{-1}\,\mathbf{f}_i^n\,\Delta t$$
$$\mathbf{x}_i^{n+1} = \mathbf{x}_i^n + \mathbf{v}_i\,\Delta t$$

A bit inconvenient to analyze mathematically:
Each $\mathbf{f}_i$ could depend on $\mathbf{x}_1$, $\mathbf{v}_1$, $\mathbf{x}_2$, $\mathbf{v}_2$, …

Not clear how to do backward Euler!

Simpler with generalized coordinates:

$$\mathbf{q} = \begin{bmatrix} \mathbf{x}_1 \\ \mathbf{x}_2 \\ \vdots \\ \mathbf{x}_n \end{bmatrix}, \quad \mathbf{v} = \begin{bmatrix} \mathbf{v}_1 \\ \mathbf{v}_2 \\ \vdots \\ \mathbf{v}_n \end{bmatrix}$$

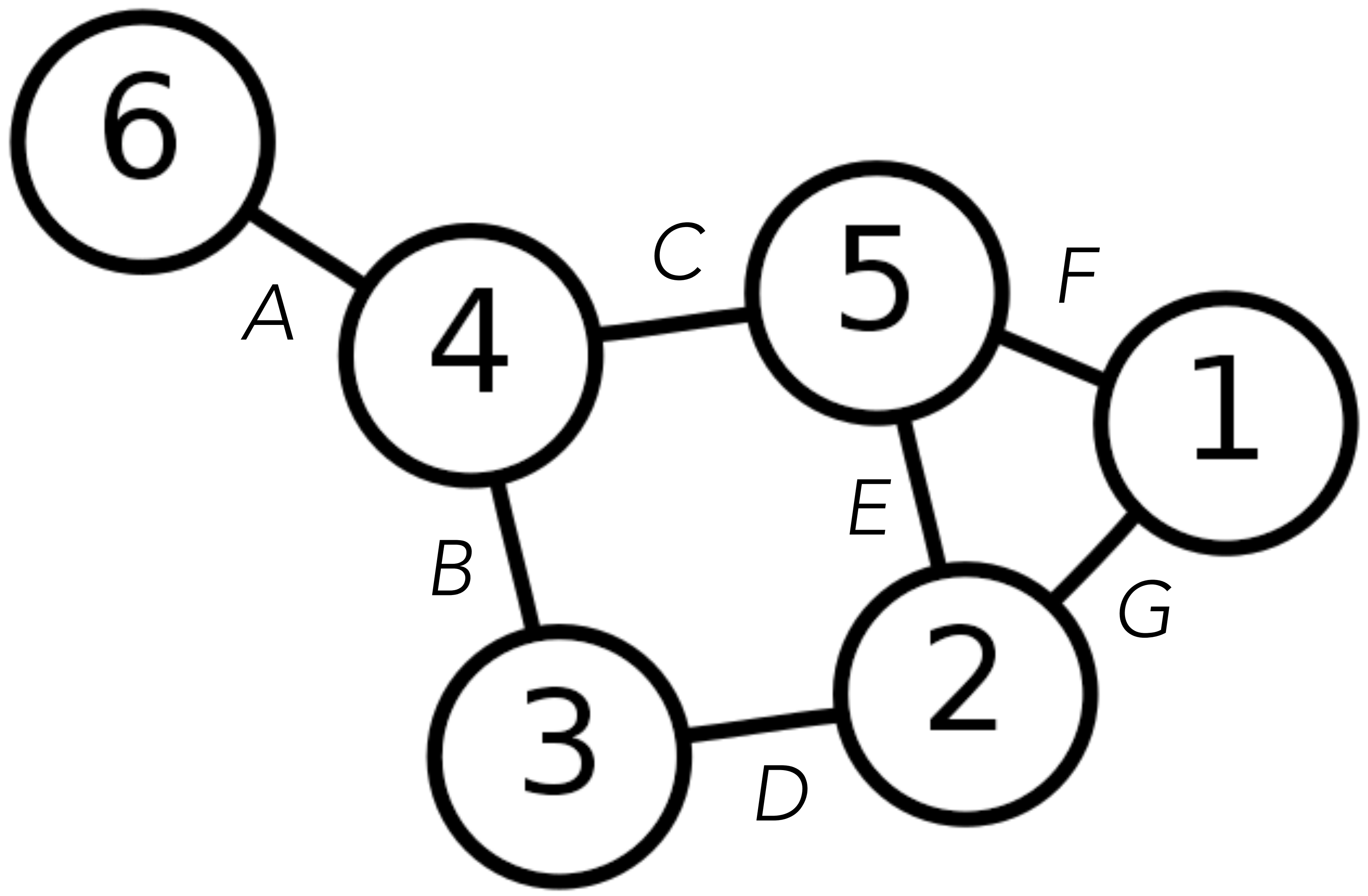
Then

$$\frac{\mathrm{d}^2\mathbf{q}(t)}{\mathrm{d}t^2} = \begin{bmatrix} m_1^{-1}\mathbf{f}_1(t, \mathbf{q}, \mathbf{v}) \\ m_2^{-1}\mathbf{f}_2(t, \mathbf{q}, \mathbf{v}) \\ \vdots \\ m_n^{-1}\mathbf{f}_n(t, \mathbf{q}, \mathbf{v}) \end{bmatrix} = \begin{bmatrix} m_1\mathbf{I} & & & \\ & m_2\mathbf{I} & & \\ & & \ddots & \\ & & & m_n\mathbf{I} \end{bmatrix}^{-1} \begin{bmatrix} \mathbf{f}_1(t, \mathbf{q}, \mathbf{v}) \\ \mathbf{f}_2(t, \mathbf{q}, \mathbf{v}) \\ \vdots \\ \mathbf{f}_n(t, \mathbf{q}, \mathbf{v}) \end{bmatrix}$$

Now we're solving for the evolution of a single (though $3n$-dimensional!) vector

Example: A small mass-spring system

$$\begin{bmatrix} \mathbf{f}_1(t, \mathbf{q}, \mathbf{v}) \\ \mathbf{f}_2(t, \mathbf{q}, \mathbf{v}) \\ \vdots \\ \mathbf{f}_6(t, \mathbf{q}, \mathbf{v}) \end{bmatrix} = \begin{bmatrix} \mathbf{f}_{1F} + \mathbf{f}_{1G} \\ \mathbf{f}_{2D} + \mathbf{f}_{2E} + \mathbf{f}_{2G} \\ \vdots \\ \mathbf{f}_{6A} \end{bmatrix}$$



Force due to spring *D*: $\begin{bmatrix} 0 \\ \mathbf{f}_{2D} \\ \mathbf{f}_{3D} \\ 0 \\ \vdots \end{bmatrix}$

(of course, $\mathbf{f}_{2D} = -\mathbf{f}_{3D}$)

Total force on system = $\sum$ force due to each spring

Per-particle formulation:

$$\frac{\mathrm{d}^2\mathbf{x}_i(t)}{\mathrm{d}t^2} = m_i^{-1}\,\mathbf{f}_i\,(t, \ldots) \quad \forall i = 1, 2, \ldots$$

$$\Downarrow$$

$$\mathbf{v}_i^{n+1} = \mathbf{v}_i^n + m_i^{-1}\,\mathbf{f}_i\,(t, \ldots)\,\Delta t \quad \forall i = 1, 2, \ldots$$
$$\mathbf{x}_i^n = \mathbf{x}_i^n + \mathbf{v}_i\,\Delta t \quad \forall i = 1, 2, \ldots$$

Careful not to update $\mathbf{x}_1$, $\mathbf{v}_1$ before computing $\mathbf{f}_2$, in case it depends on them

Generalized coordinates:

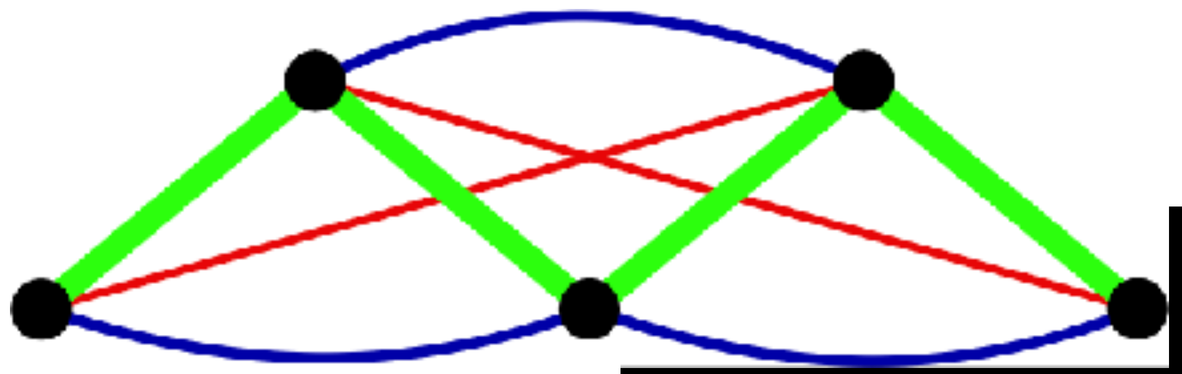
$$\frac{\mathrm{d}^2\mathbf{q}(t)}{\mathrm{d}t^2} = \mathbf{M}^{-1}\,\mathbf{f}(t, \mathbf{q}, \mathbf{v})$$

$$\Downarrow$$

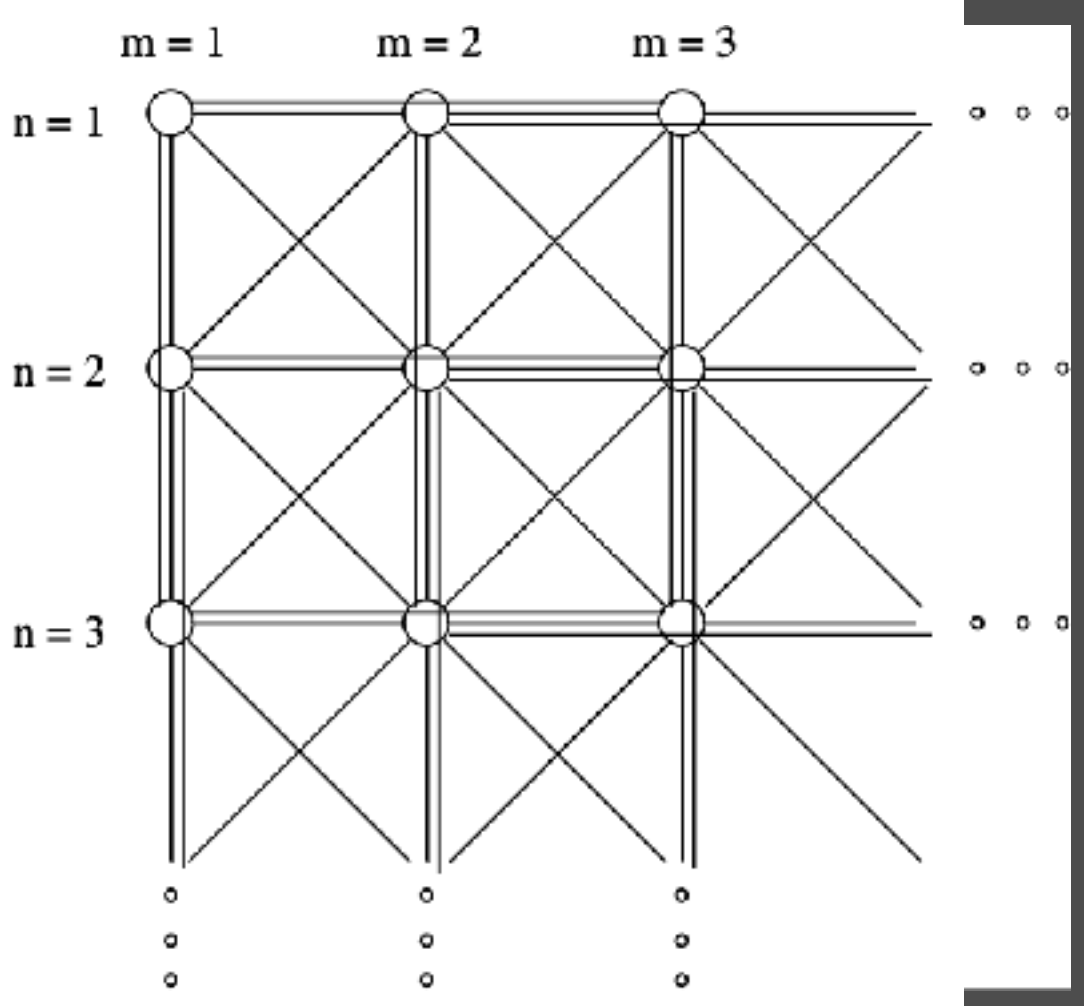
$$\mathbf{v}^{n+1} = \mathbf{v}^n + \mathbf{M}^{-1}\,\mathbf{f}(t, \mathbf{q}, \mathbf{v})\,\Delta t$$
$$\mathbf{q}^{n+1} = \mathbf{q}^n + \mathbf{v}\,\Delta t$$

Simple! And generalizes to other things (e.g. rigid bodies) with few changes

Mass-spring systems



Selle et al. 2008



Choi & Ko 2002

Recall springs in 1 dimension from physics classes.

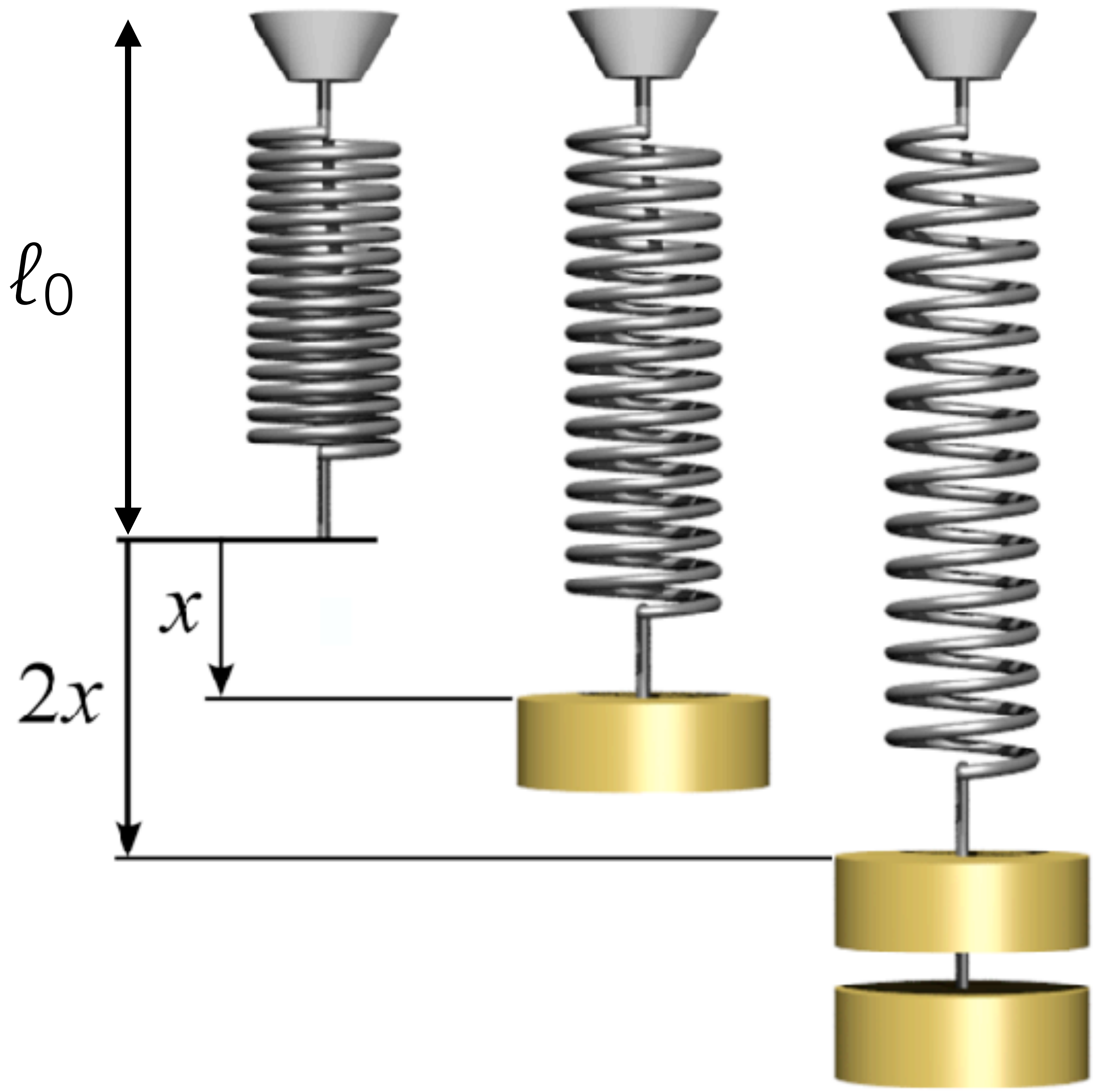
Hooke's law: force is proportional to displacement

$$F = -k\,x = -k\,(\ell - \ell_0)$$

Potential energy:

$$U = \tfrac{1}{2}\,k\,(\ell - \ell_0)^2$$

In fact $F = -\mathrm{d}U/\mathrm{d}\ell$



In 3D, suppose a spring connects particles *i* and *j*. What should be the force $\mathbf{f}_{ij}$ on *i* due to *j*?

Let's first define the potential:

$$U = \tfrac{1}{2}\, k\, (\|\mathbf{x}_i - \mathbf{x}_j\| - \ell_0)^2$$

Then $\mathbf{f}_{ij} = -\partial U/\partial \mathbf{x}_i \Rightarrow$

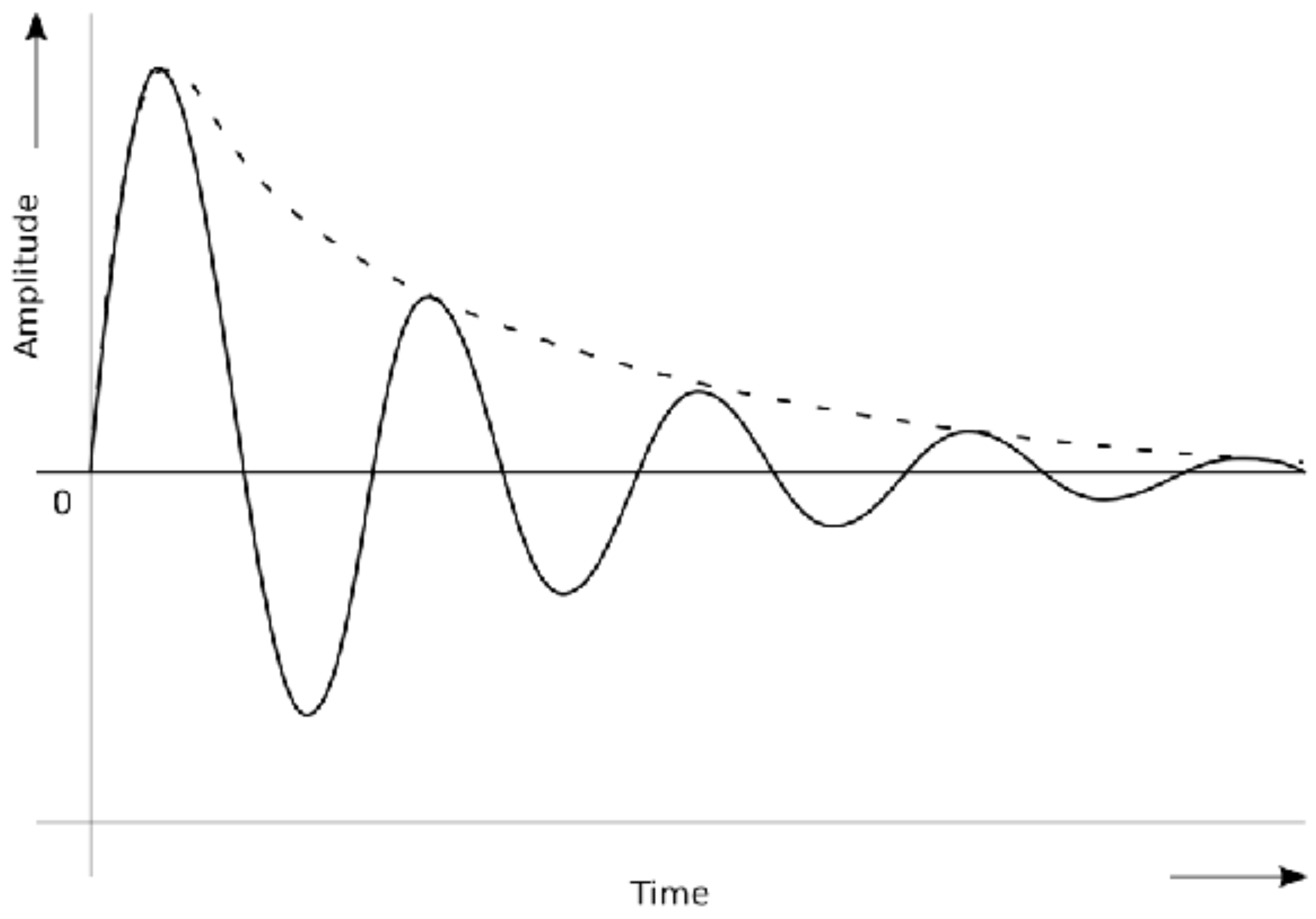
$$\mathbf{f}_{ij} = -k\, (\|\mathbf{x}_i - \mathbf{x}_j\| - \ell_0)\, \frac{\mathbf{x}_i - \mathbf{x}_j}{\|\mathbf{x}_i - \mathbf{x}_j\|}$$

$$= -k\, (\|\mathbf{x}_{ij}\| - \ell_0)\, \hat{\mathbf{x}}_{ij}$$

Similarly $\mathbf{f}_{ji} = -\partial U/\partial \mathbf{x}_j$ (but it's also just $-\mathbf{f}_{ij}$)

Exercise: Derive this expression from $-\partial U/\partial \mathbf{x}_i$. Optional: Look up multivariable calculus identities, chain rule, etc. so you don't have to differentiate componentwise.

Problem: Real springs dissipate energy and don't keep oscillating forever!



Bad idea: Just add a force that opposes all velocities

$$\mathbf{f}_i = -k_d \, \mathbf{v}_i$$

Sometimes called "**ether drag**"

- Particles look like they're suspended in a viscous medium
- Should a rusty spring fall slower than a clean spring?

Good idea: Only oppose relative velocities along the spring

$$\mathbf{f}_{ij} = -k_d \, (\mathbf{v}_{ij} \cdot \hat{\mathbf{x}}_{ij}) \, \hat{\mathbf{x}}_{ij}$$

Force due to a spring, finally:

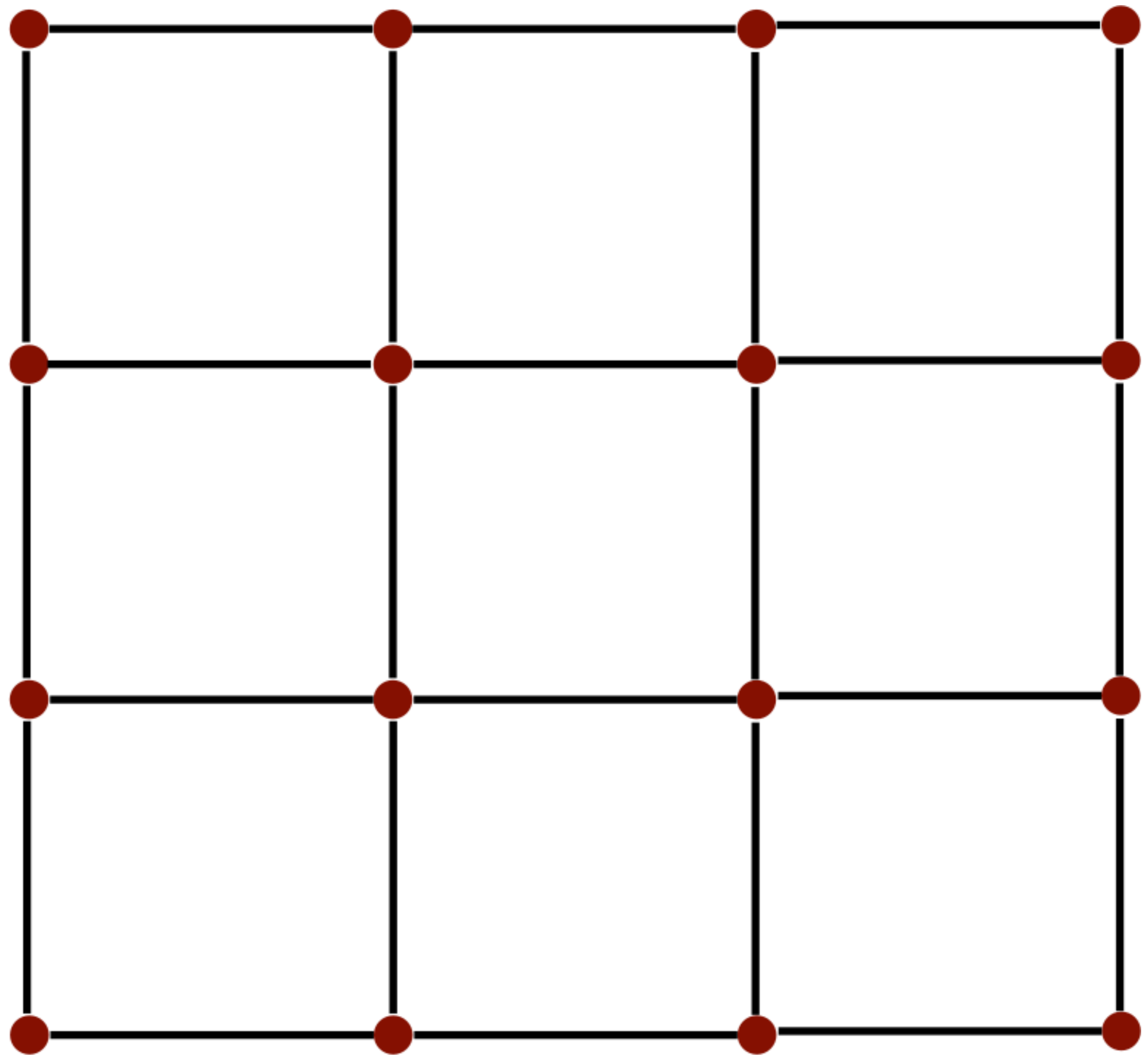
$$\mathbf{f}_{ij} = -k_s \left(\|\mathbf{x}_{ij}\| - \ell_0\right) \hat{\mathbf{x}}_{ij} - k_d \left(\mathbf{v}_{ij} \cdot \hat{\mathbf{x}}_{ij}\right) \hat{\mathbf{x}}_{ij}$$

where

- Spring constant $k_s \geq 0$
- Damping constant $k_d \geq 0$

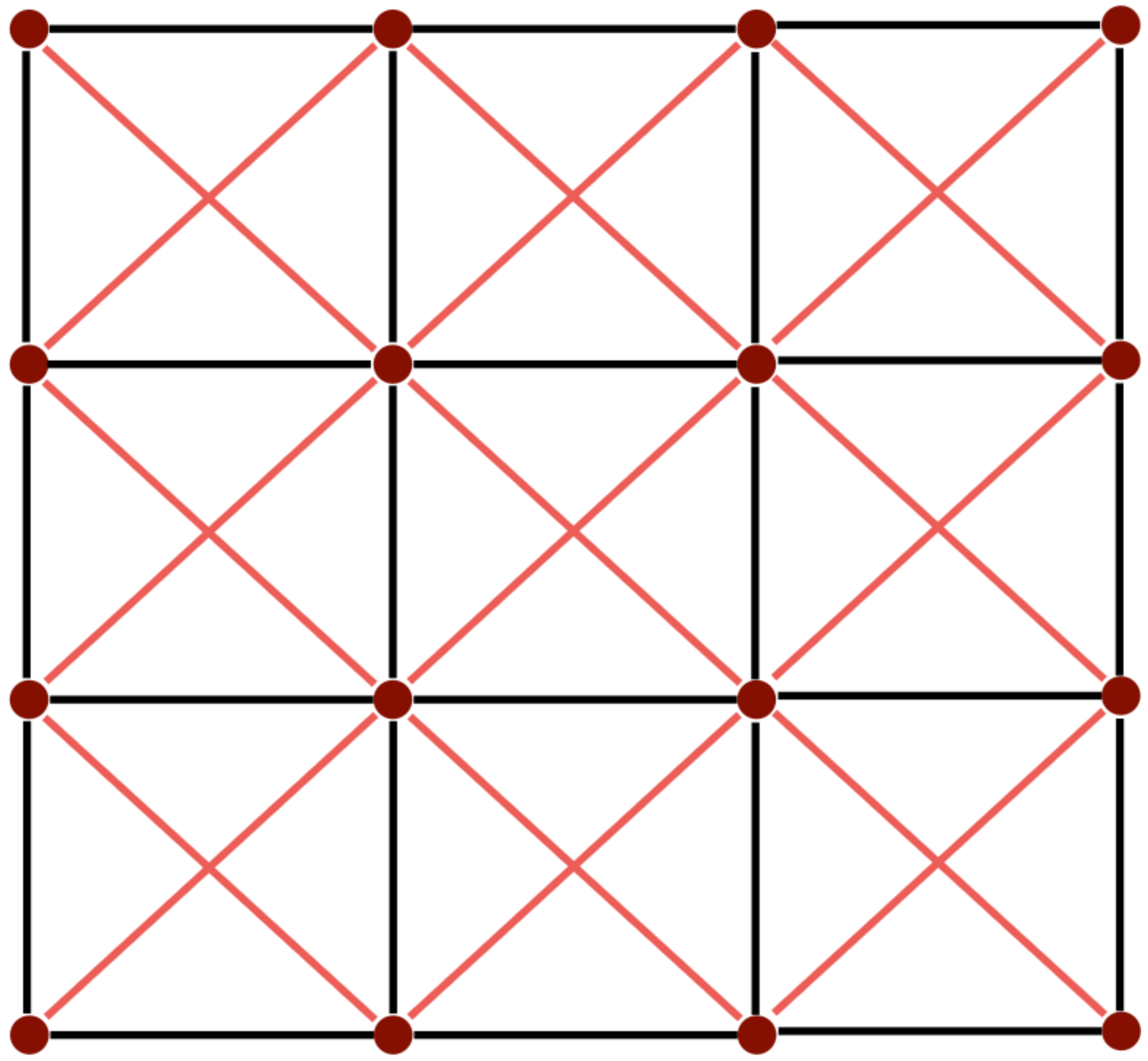
How to model a rectangular sheet of cloth?

- Structural springs



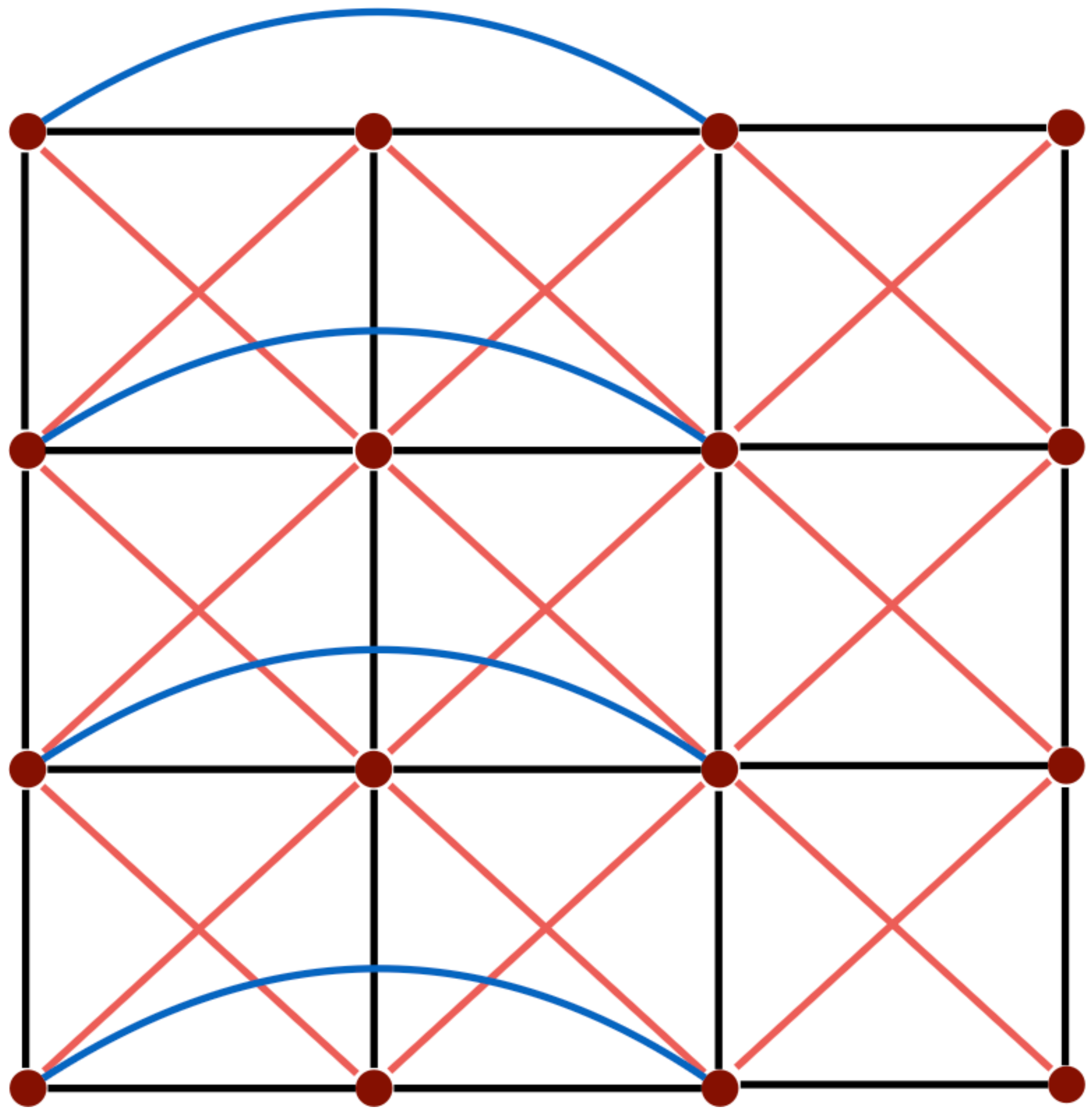
How to model a rectangular sheet of cloth?

- Structural springs
- Shear springs



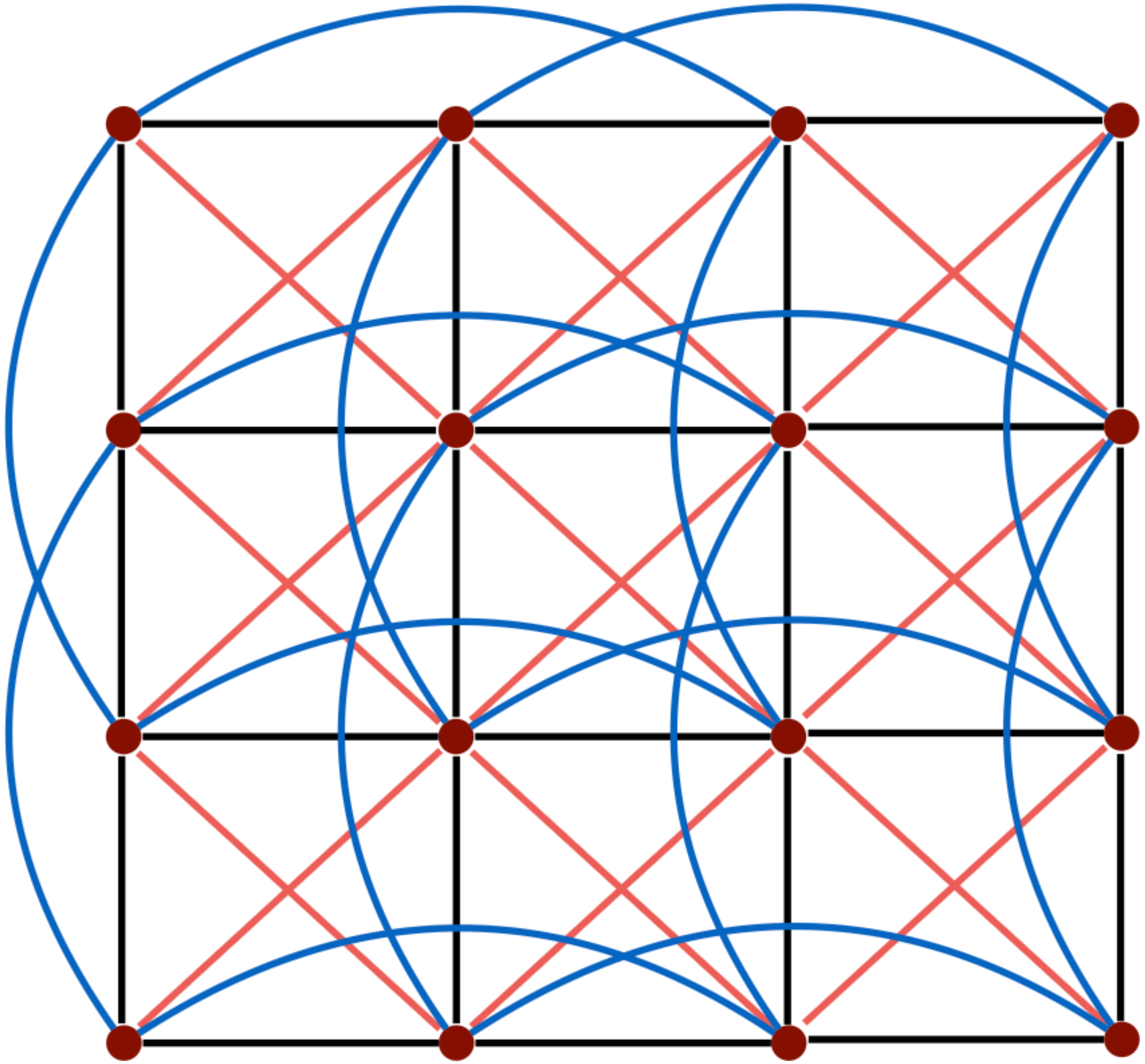
How to model a rectangular sheet of cloth?

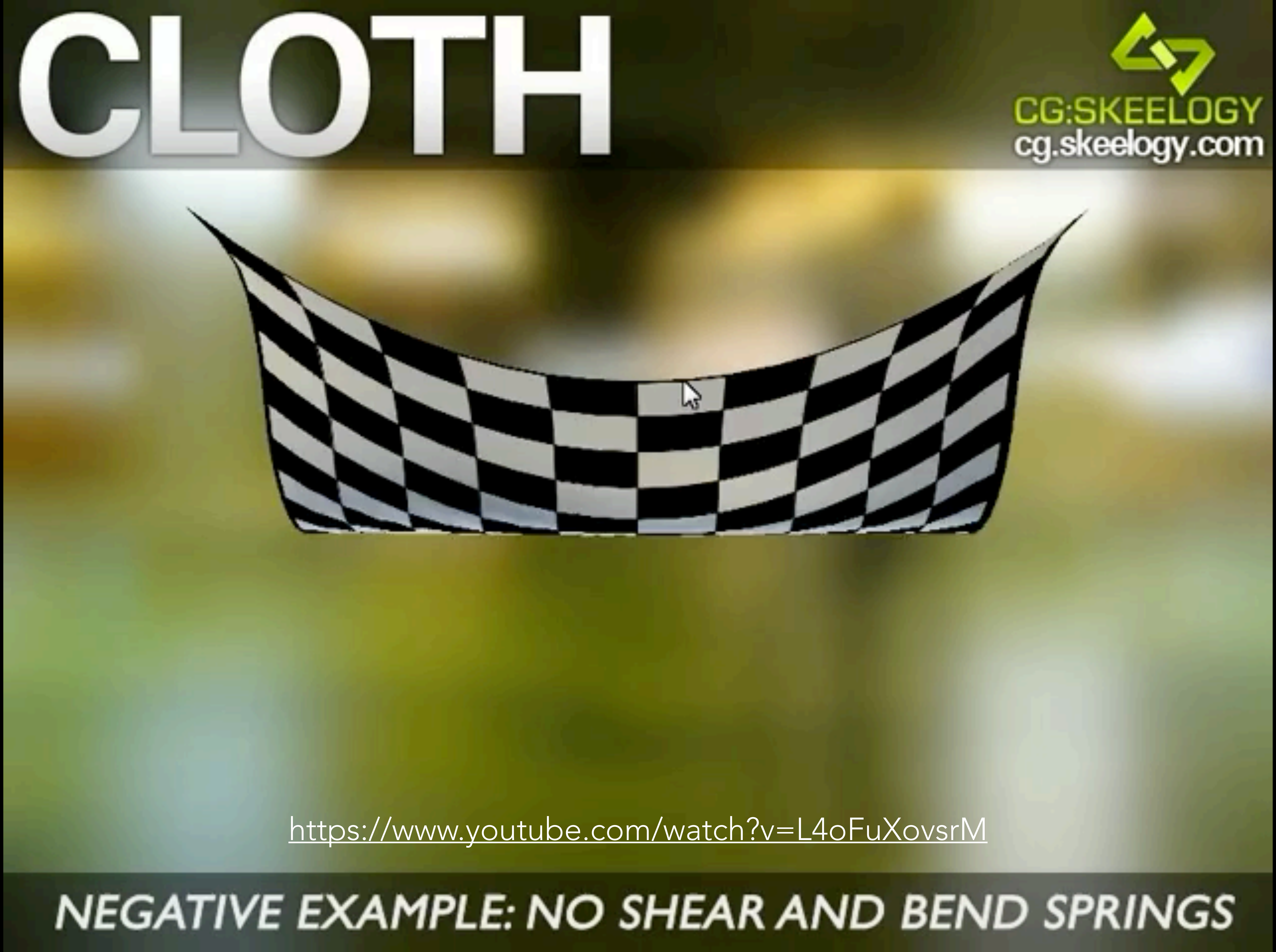
- Structural springs
- Shear springs
- Bending springs



How to model a rectangular sheet of cloth?

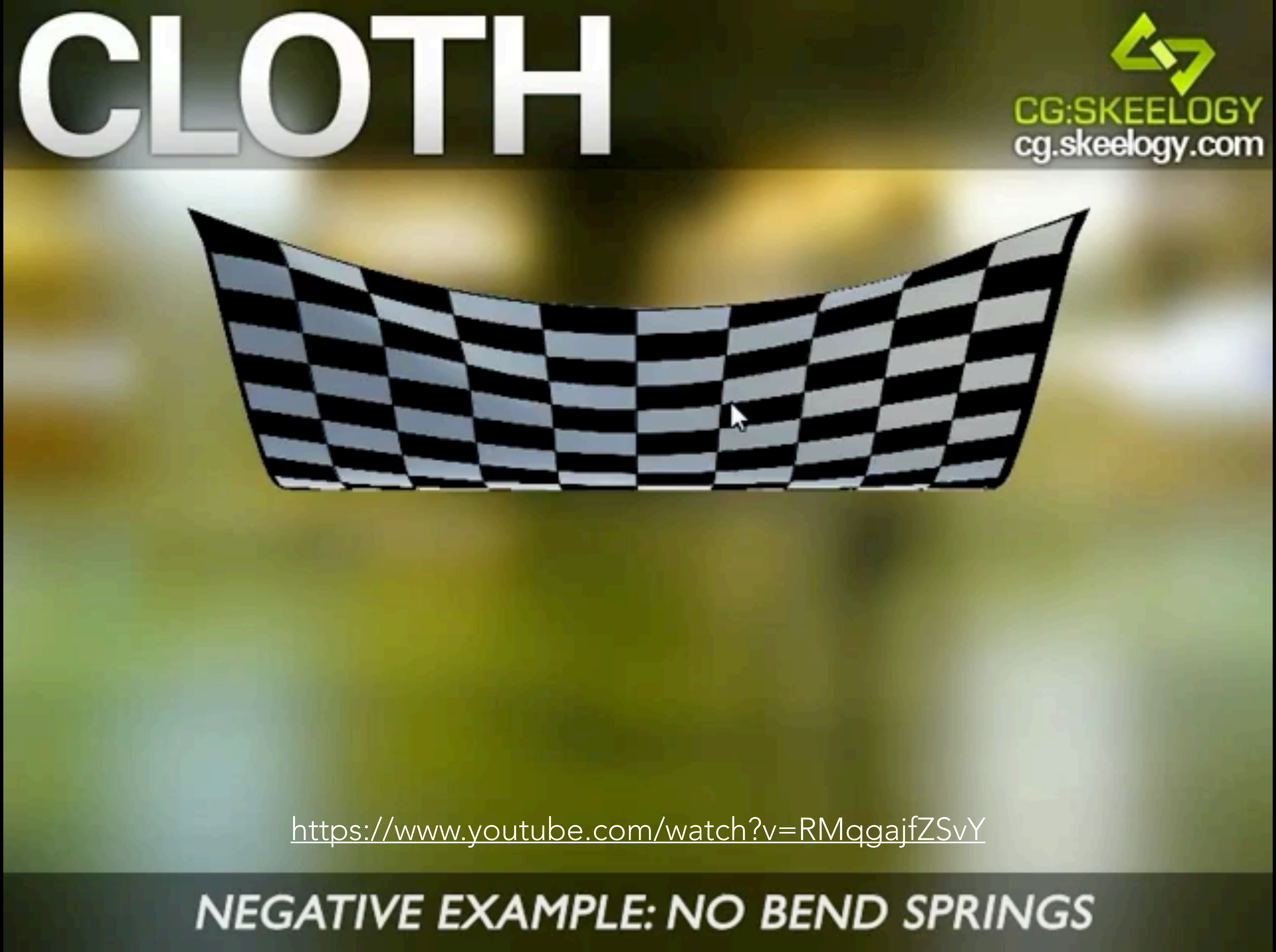
- Structural springs
- Shear springs
- Bending springs





https://www.youtube.com/watch?v=L4oFuXovsrM

Skeel Lee

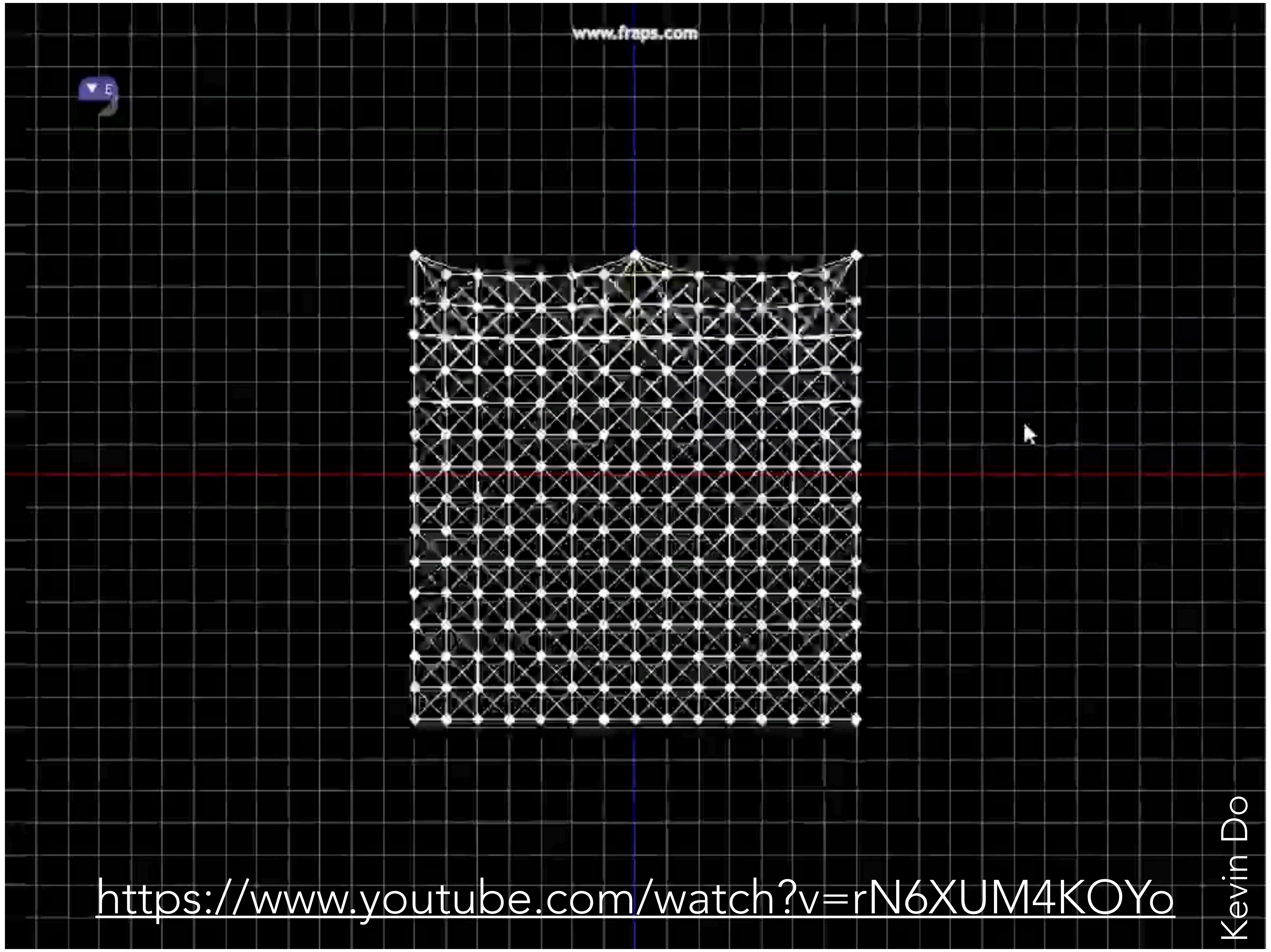


https://www.youtube.com/watch?v=RMqgajfZSvY

https://www.youtube.com/watch?v=JncQOCg9FS0

Here's what instability looks like
for a mass-spring system:

To avoid this, let's talk about how to
do backward Euler time stepping.



Recall: backward Euler gives us a system of equations in the unknown next state $(\mathbf{q}_{n+1}, \mathbf{v}_{n+1})$

$$\mathbf{q}_{n+1} = \mathbf{q}_n + \mathbf{v}_{n+1}\,\Delta t$$
$$\mathbf{v}_{n+1} = \mathbf{v}_n + \mathbf{M}^{-1}\,\mathbf{f}(\mathbf{q}_{n+1}, \mathbf{v}_{n+1})\,\Delta t$$

How do we solve this system of equations? Newton's method!

Pick a guess $(\tilde{\mathbf{q}}, \tilde{\mathbf{v}})$. A natural choice is to start with $\tilde{\mathbf{q}} = \mathbf{q}_n$, $\tilde{\mathbf{v}} = \mathbf{v}_n$.

1. Linearize the problem:

$$(\tilde{\mathbf{q}}+\Delta\mathbf{q}) = \mathbf{q}_n + (\tilde{\mathbf{v}}+\Delta\mathbf{v})\,\Delta t$$

$$(\tilde{\mathbf{v}}+\Delta\mathbf{v}) = \mathbf{v}_n + \mathbf{M}^{-1}\,\mathbf{f}(\tilde{\mathbf{q}}+\Delta\mathbf{q}, \tilde{\mathbf{v}}+\Delta\mathbf{v})\,\Delta t$$
$$\approx \mathbf{v}_n + \mathbf{M}^{-1}\left(\mathbf{f}(\tilde{\mathbf{q}}, \tilde{\mathbf{v}}) + \frac{\partial \mathbf{f}}{\partial \mathbf{q}}(\tilde{\mathbf{q}}, \tilde{\mathbf{v}})\,\Delta\mathbf{q} + \frac{\partial \mathbf{f}}{\partial \mathbf{v}}(\tilde{\mathbf{q}}, \tilde{\mathbf{v}})\,\Delta\mathbf{v}\right)\Delta t$$

Here we need the **force Jacobians** $\frac{\partial \mathbf{f}}{\partial \mathbf{q}}$ and $\frac{\partial \mathbf{f}}{\partial \mathbf{v}}$: how does $\mathbf{f}$ change with $\mathbf{q}$ and $\mathbf{v}$?

1. Linearize the problem:

$$(\tilde{\mathbf{q}}+\Delta\mathbf{q}) = \mathbf{q}_n + (\tilde{\mathbf{v}}+\Delta\mathbf{v})\,\Delta t$$

$$(\tilde{\mathbf{v}}+\Delta\mathbf{v}) = \mathbf{v}_n + \mathbf{M}^{-1}\,(\mathbf{f}(\tilde{\mathbf{q}}, \tilde{\mathbf{v}}) + \mathbf{J}_\mathbf{q}(\tilde{\mathbf{q}}, \tilde{\mathbf{v}})\,\Delta\mathbf{q} + \mathbf{J}_\mathbf{v}(\tilde{\mathbf{q}}, \tilde{\mathbf{v}})\,\Delta\mathbf{v})\,\Delta t$$

2. Now the system is linear in $(\Delta\mathbf{q}, \Delta\mathbf{v})$. Plug into any linear solver.

Faster solves: substitute $\Delta\mathbf{q}$ in terms of $\Delta\mathbf{v}$, then rearrange to get something of the form

$$(\mathbf{M} - \mathbf{J}_\mathbf{v}\,\Delta t - \mathbf{J}_\mathbf{q}\,\Delta t^2)\,\Delta\mathbf{v} = \cdots$$

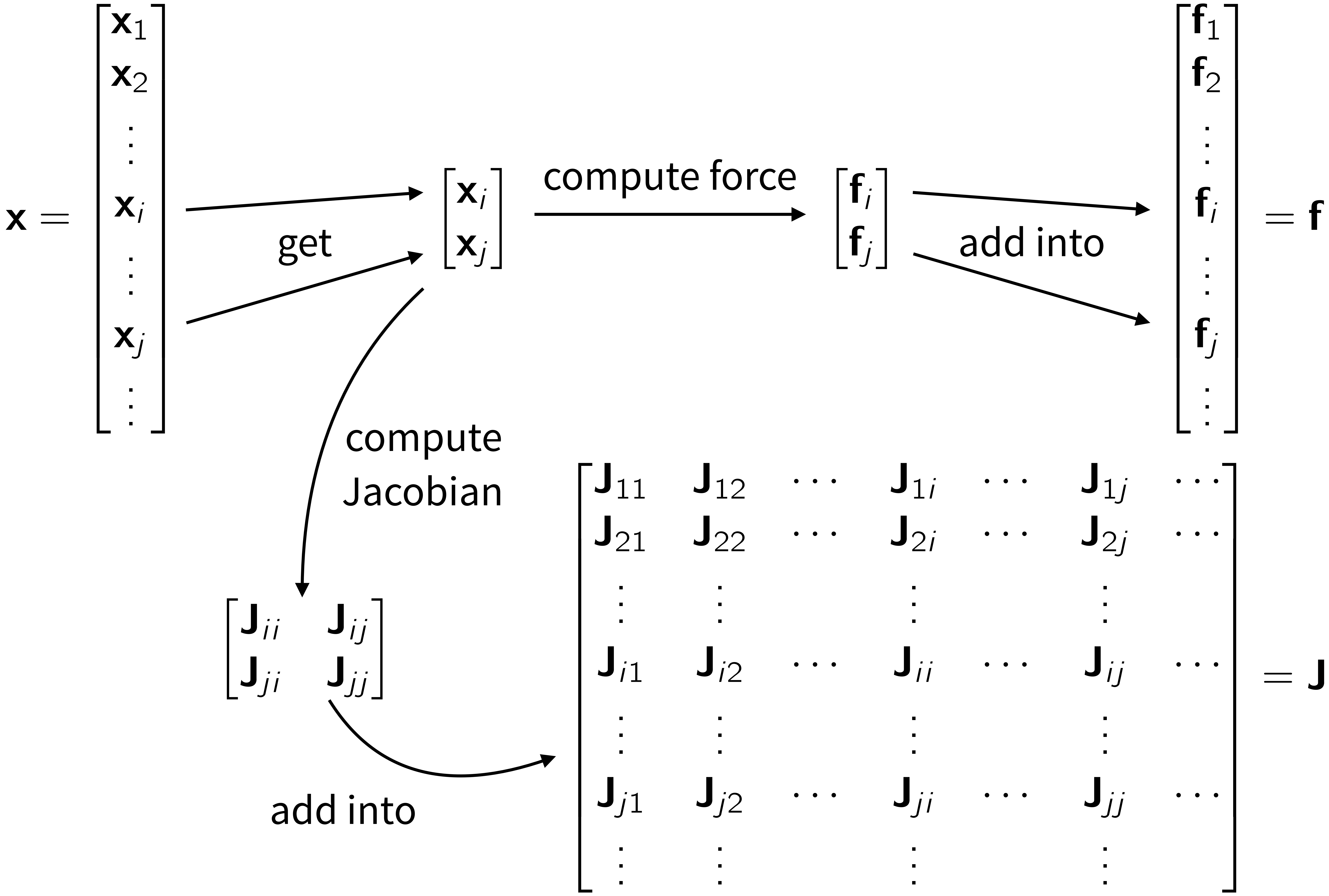
Left-hand side is almost always symmetric, often positive definite

Example: force Jacobians for springs

$$\mathbf{f}_{ij} = -k_s \left(\|\mathbf{x}_{ij}\| - \ell_0\right) \hat{\mathbf{x}}_{ij} - k_d \left(\mathbf{v}_{ij} \cdot \hat{\mathbf{x}}_{ij}\right) \hat{\mathbf{x}}_{ij}$$

$$\frac{\partial \mathbf{f}_{ij}}{\partial \mathbf{x}_i} = ?$$

$$\frac{\partial \mathbf{f}_{ij}}{\partial \mathbf{v}_i} = ?$$


x =
get
compute force
add into
= f
compute
Jacobian
add into
= J

Homework problem

Suppose you try to implement backward Euler:

$$(\mathbf{M} - \mathbf{J_v}\,\Delta t - \mathbf{J_q}\,\Delta t^2)\,\Delta\mathbf{v} = \cdots$$

But you're too lazy to derive the force Jacobians, and you replace them with zeroes instead.

What kind of time integration scheme do you get? Does it reduce to a known one?