CS**5643**

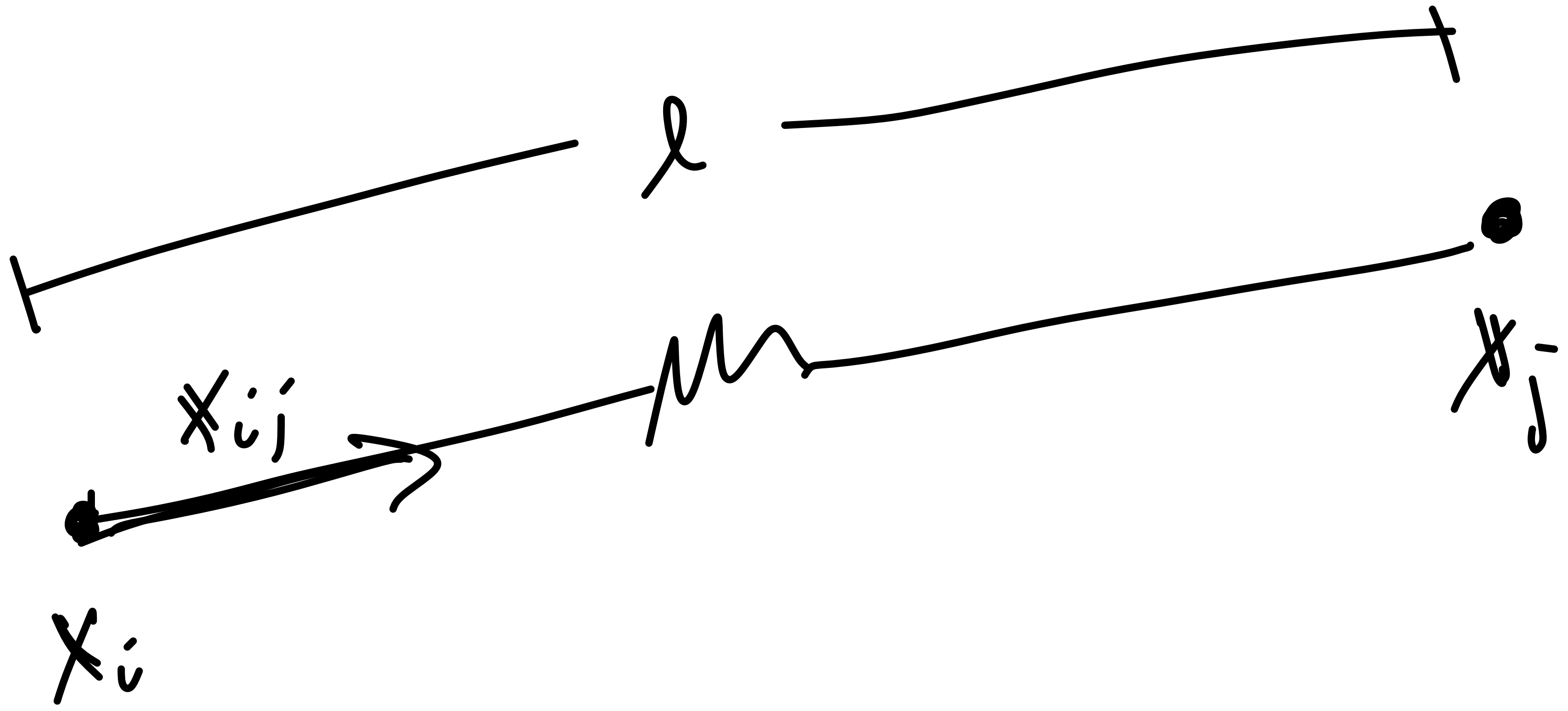
05 Mass-and-spring models

Steve Marschner
Cornell University
Spring 2023

Binary Spring

The most basic modeling tool for all kinds of deformable things

Spring defined by



- which particles i and j it connects
- its spring stiffness k_s
- its rest length l_0

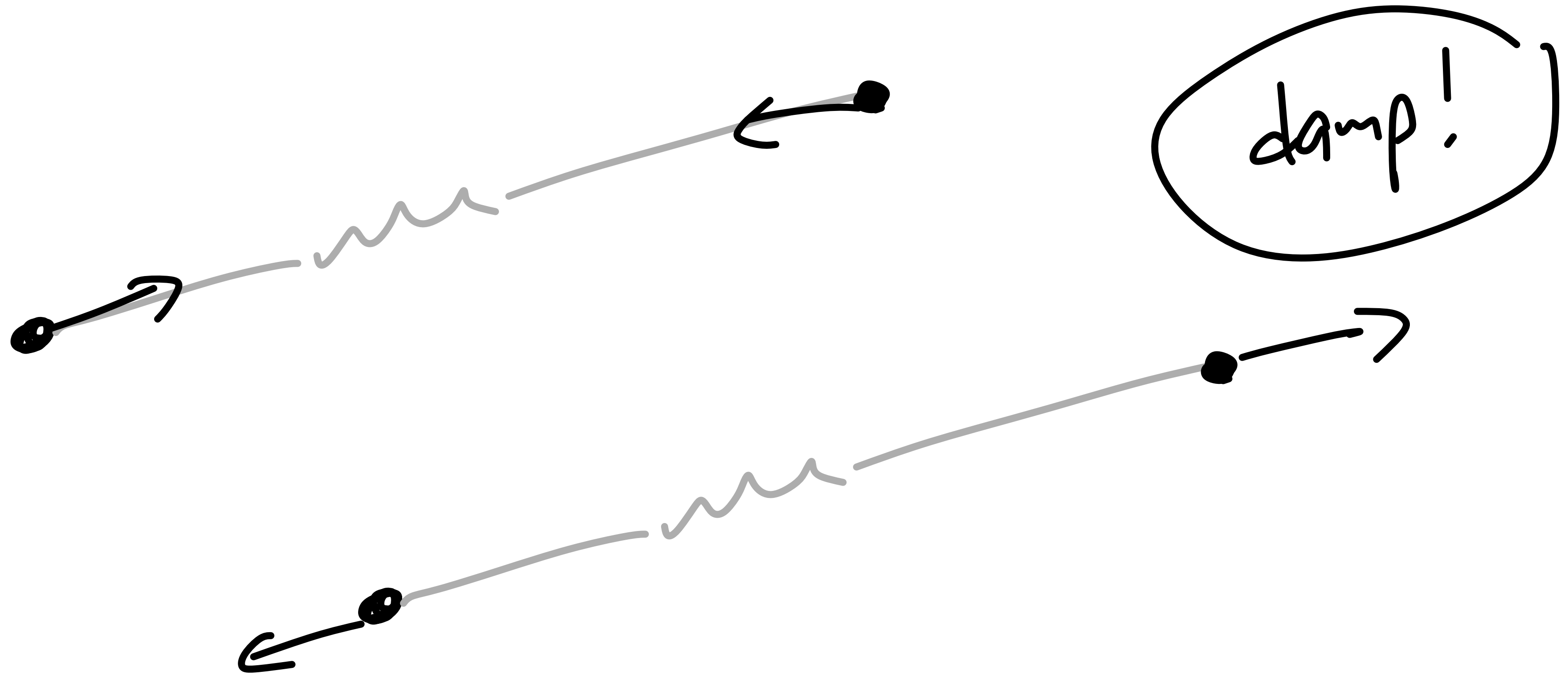
From Hooke's law we know force is proportional to displacement from rest

- $f = k_s(l - l_0)$

The force acts along the direction of the spring

- $\mathbf{f}_i = k_s(\|\mathbf{x}_{ij}\| - l_0)\hat{\mathbf{x}}_{ij}$ where $\mathbf{x}_{ij} = \mathbf{x}_j - \mathbf{x}_i$ (quick sanity check: pulls i towards j when stretched)
- $\mathbf{f}_j = k_s(\|\mathbf{x}_{ji}\| - l_0)\hat{\mathbf{x}}_{ji} = -\mathbf{f}_{ij}$

Adding damping



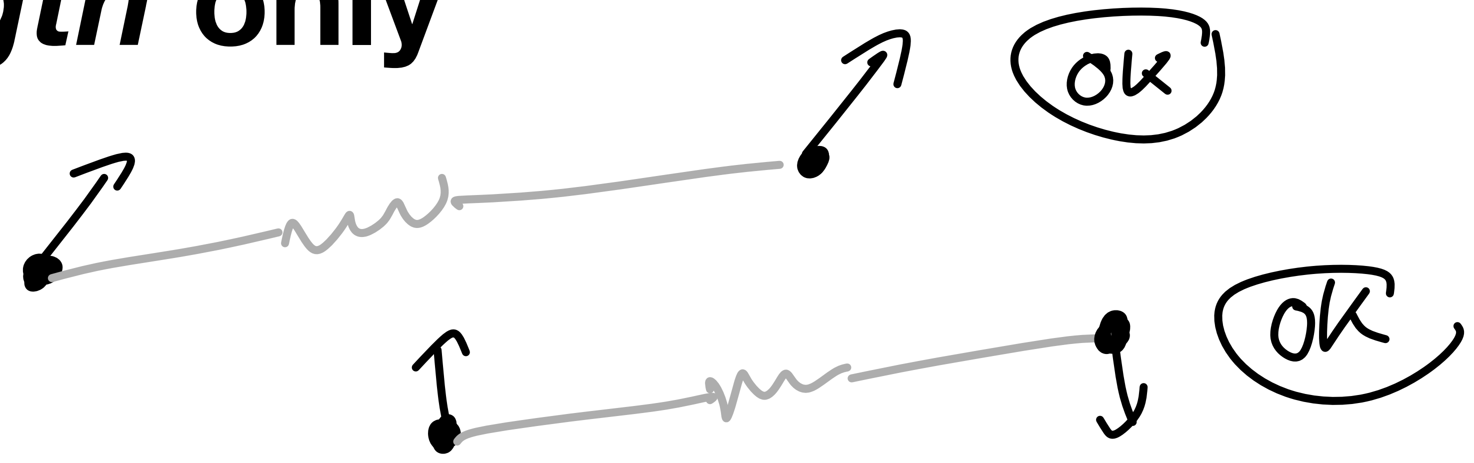
Springs are usually too springy!

- a system of springs will oscillate forever…

Damping will dissipate energy

- but using the drag forces $\mathbf{f}_d = -k_d \mathbf{v}$ we use for free particles can slow things inappropriately
- this force opposes all motion; we only want to oppose the spring's movement

Spring damping force opposes *changes in spring length* only

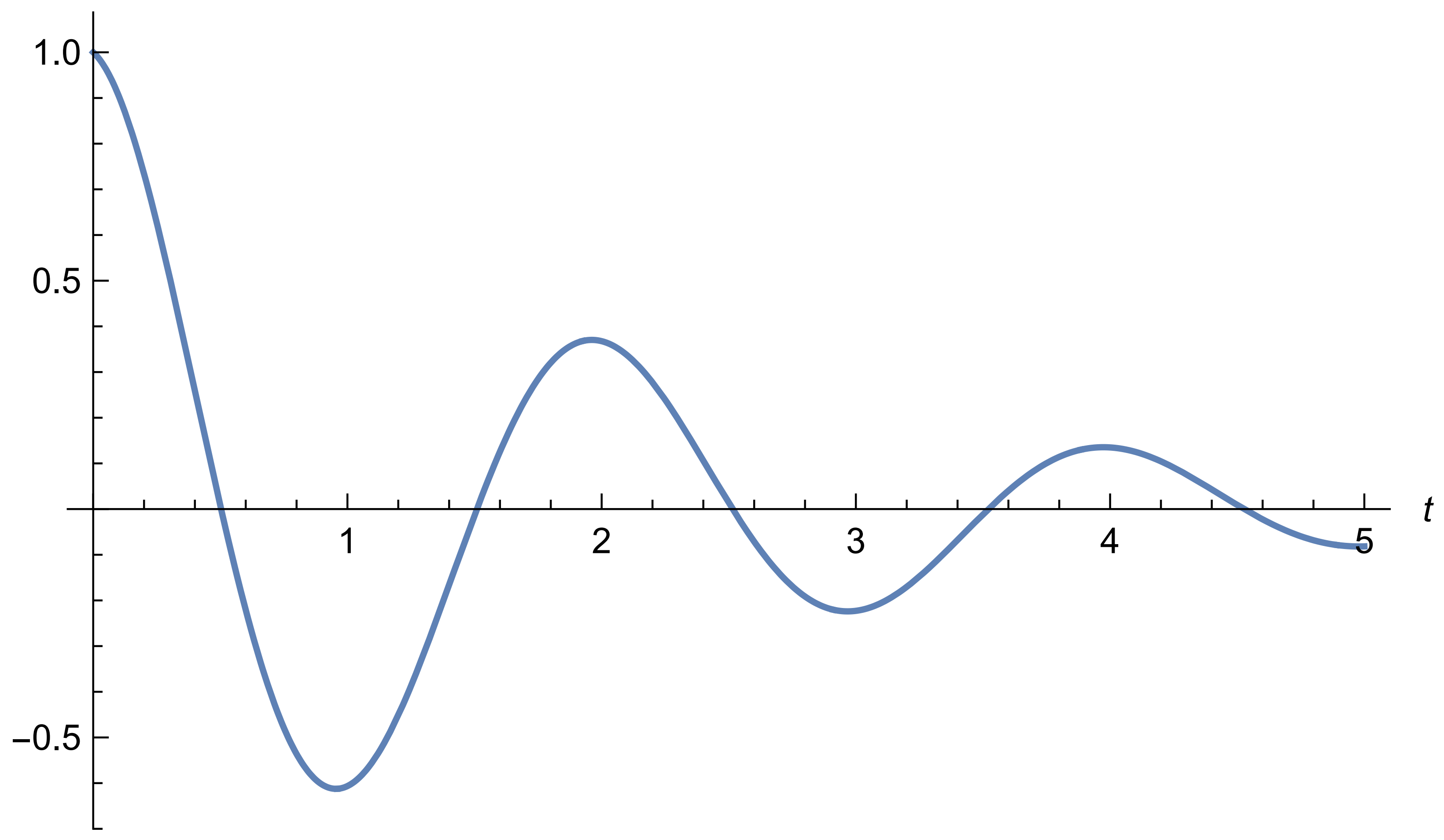


- only opposes *relative* motion
- only opposes motion *in the direction of the spring*
- $\mathbf{f}_i = k_d(\mathbf{v}_{ij} \cdot \hat{\mathbf{x}}_{ij})\hat{\mathbf{x}}_{ij}$ where $\mathbf{v}_{ij} = \mathbf{v}_j - \mathbf{v}_i$ (sanity check: pulls i towards j when elongating)

Phenomena of damped springs

Oscillations of undamped springs

- a 1D damped spring obeys $m\ddot{x} + k_d\dot{x} + k_s x = 0$
- when k_d is negligible the solution is $x(t) = C_1 \cos(\omega t + C_2)$ where $\omega^2 = \dfrac{k_s}{m}$
  - stiffer spring → faster oscillation; higher mass → slower oscillation
- when k_s is negligible the solution is $x(t) = c_0 + c_1 T(1 - e^{t/T})$ where $T = \dfrac{m}{k_d}$
  - higher mass → takes longer to stop; higher damping → takes less time to stop
- in general case we get $x(t) = Ce^{-t/T}\cos(\omega t + C_2)$ where $T = \dfrac{2m}{k_d}$ and $\omega = \sqrt{k_s/m - (1/T)^2}$
  - decaying oscillation combining the two above behaviors
  - damping also slows the oscillation; when we reach $\omega = 0$ the system is "critically damped"



Modeling rods with springs

A rod is a long, slender, flexible object (essentially 1D)

- can stretch or bend and elastically resist both

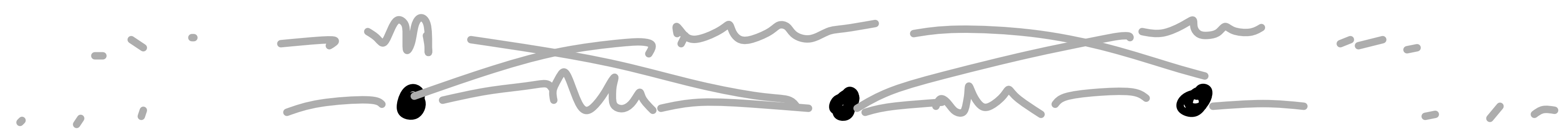
Basic plan: a chain of masses and springs

- N springs of length L/N and spring constant … Nk_s (why?)

- this handles stretching but doesn't oppose bending—too "floppy" and chain-like

Simple way to resist bending: bending springs

- springs that skip one particle
- $N - 1$ springs of length $2L/N$ and spring constant Nk_s' where k_s' is usually quite a bit less than k_s



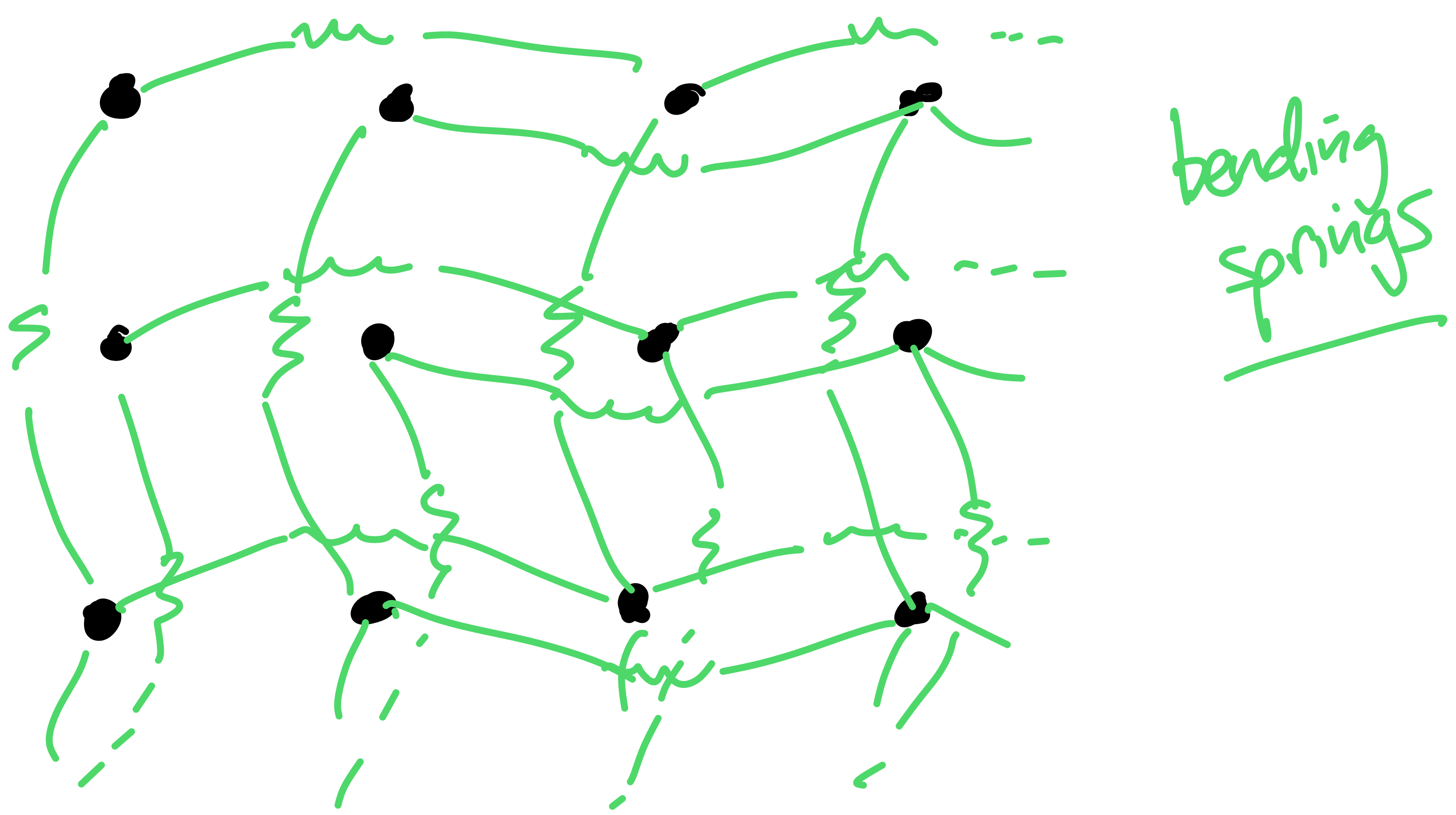
Modeling cloth with springs

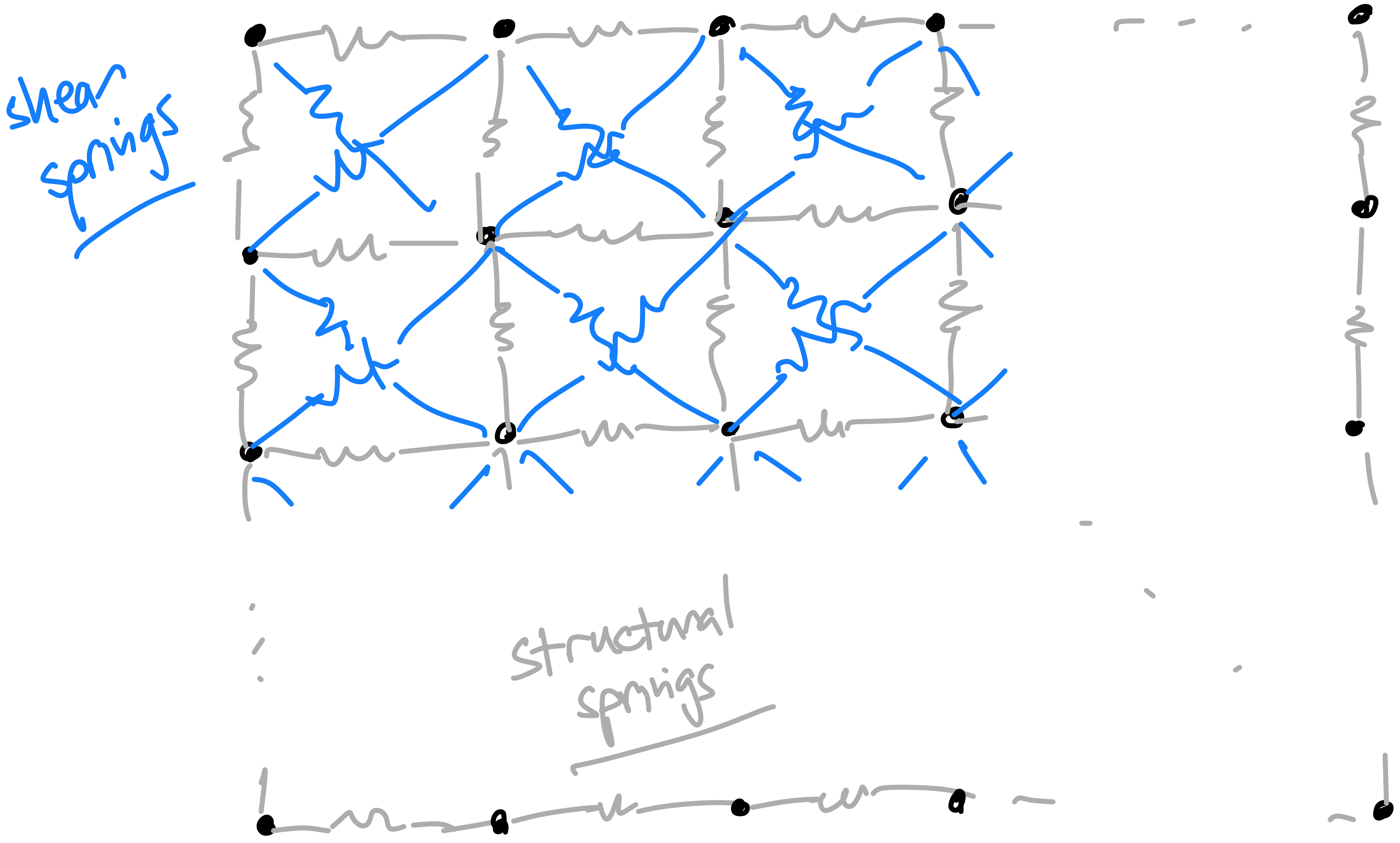
Very similar idea to rods, but in a 2D grid

- structural springs along the axes (stiff for woven cloth, softer for knitted)
- bending springs skipping one particle (weak to allow lots of bending)
- shear springs along the diagonals (weaker than structural, to allow shearing)

You'll do this in the assignment!

- it's only cloth if it's in 3D,
  so can't really demo this ;)





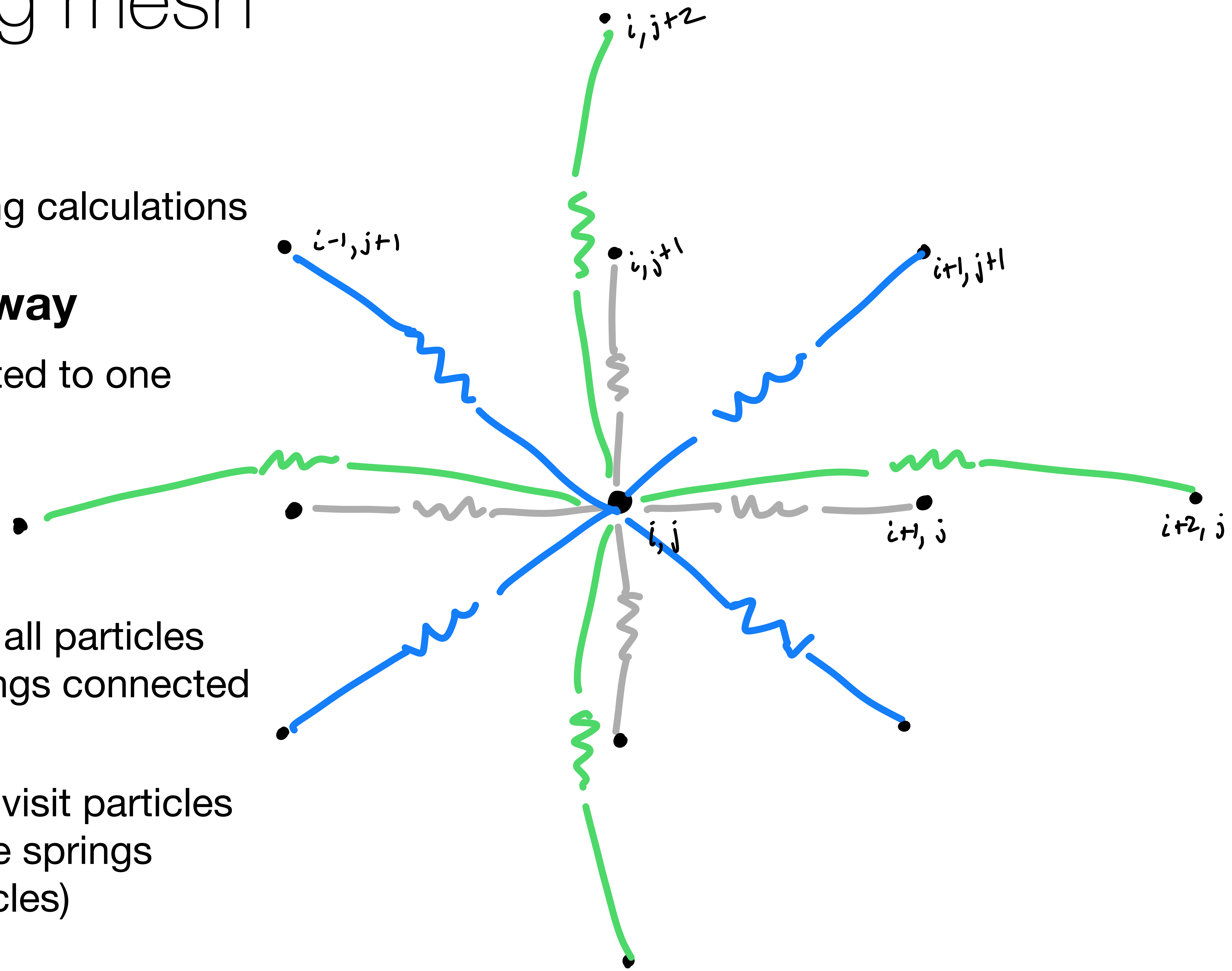
Indexing in a spring mesh

index particles in 2D

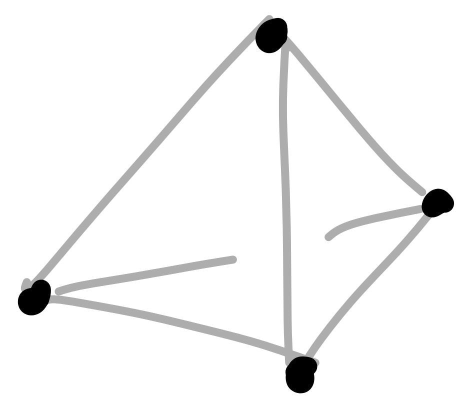
- avoids row-column indexing calculations

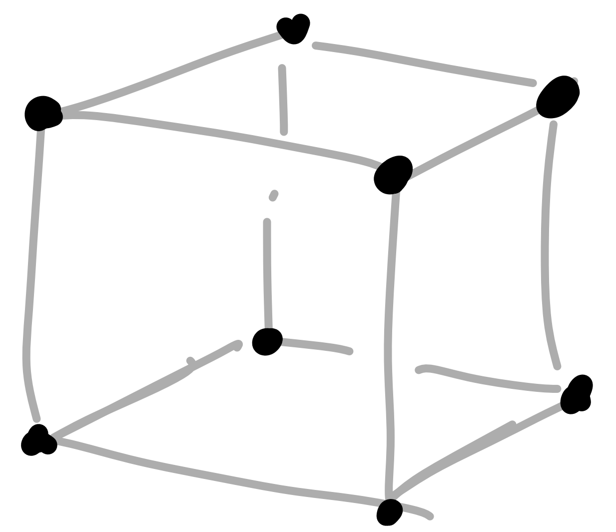
index springs the same way

- think of all springs connected to one particle
- compute forces by visiting all particles and considering all springs connected to that particle
- (to enumerate the springs, visit particles and considering only the springs connected to later particles)



Modeling deformable solids with springs





2D: looks a lot like cloth

- don’t need bending springs
- shear springs should probably be stronger than for cloth
- a triangular mesh only requires springs along the edges
- in a 2D space, a 2D object can resist compression

3D: need enough springs to prevent collapsing

- for a cube mesh, various strategies are possible — bracing diagonals of faces, or bracing across the diagonals of the cube
- a tetrahedral mesh is naturally stable with just a spring along each edge

Deriving forces from energies

Binary springs are simple and are a lot of fun to play with but they eventually start to become limited

- bending and shear springs contribute also to stretching stiffness
- difficult to achieve behavior matching particular measurements or material models
- bending springs are not very good at resisting slight bending (bending stiffness = 0 when straight!)
- difficult or impossible to express things like volume or area preservation

The spring force belongs to a useful class

- it is a conservative force, meaning it takes the same amount of work to get from one configuration to another regardless of the path
- this means it is the derivative (gradient in this case) of a potential … and the potential is literally the potential energy stored in the spring!

Working out our spring force from the energy

Start with the spring energy

- $E_{ij}(\mathbf{x}) = \frac{1}{2}k_s(\|\mathbf{x}_i - \mathbf{x}_j\| - l_0)^2$ (this is the contribution of one spring to the total system energy)

Force is minus the gradient of energy

- $\mathbf{f}_i(\mathbf{x}) = \dfrac{\partial E}{\partial \mathbf{x}_i}(\mathbf{x})$ (remember $\mathbf{x}$ is a big vector of all the positions; this partial derivative is zero for all the particles that are not connected to this particular spring)

Take the computation one step at a time:

- derivative of $\mathbf{x}_i - \mathbf{x}_j$
- derivative of $\|\mathbf{v}\|$ wrt. $\mathbf{v}$
- derivative of E_{ij} wrt $\|\mathbf{v}\|$