物理仿真

北京大学 前沿计算研究中心

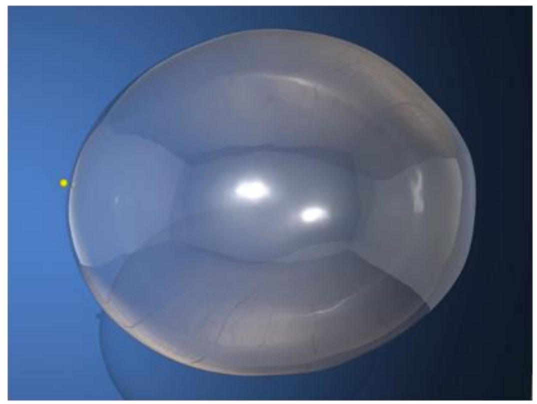
刘利斌

Ruan, Liangwang, et al. "Solid-fluid interaction with surface-tension-dominant contact." *ACM Transactions on Graphics (TOG)* 40.4 (2021): 1-12.

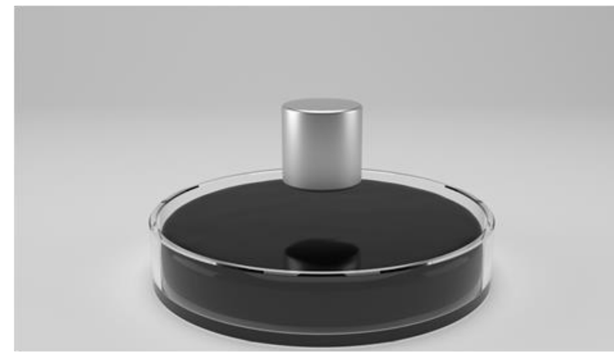
Zhu, Bo, et al. "A new grid structure for domain extension." *ACM Transactions on Graphics (TOG)* 32.4 (2013): 1-12.

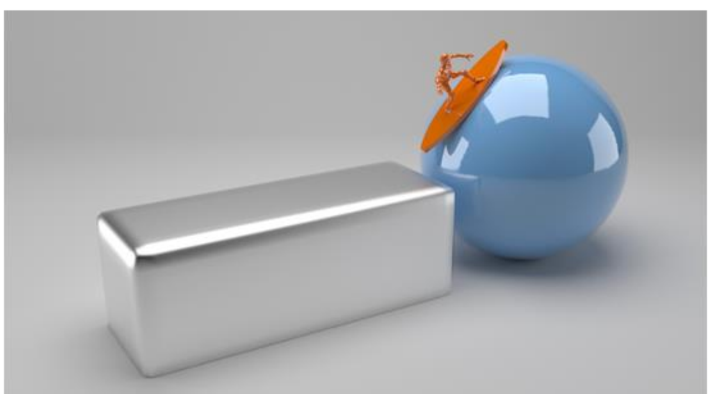
Wang, Huamin. "GPU-based simulation of cloth wrinkles at submillimeter levels." ACM Transactions on Graphics (TOG) 40.4 (2021): 1-14.

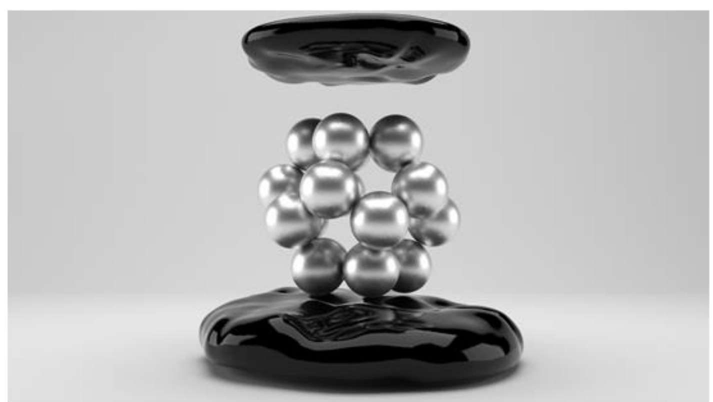
Stomakhin, Alexey, et al. "A material point method for snow simulation." *ACM Transactions on Graphics (TOG)* 32.4 (2013): 1-10.



Zhu, Bo, et al. "Codimensional surface tension flow on simplicial complexes." *ACM Transactions on Graphics (TOG)* 33.4 (2014): 1-11.







Sun, Yuchen, et al. "A material point method for nonlinearly magnetized materials." *ACM Transactions on Graphics (TOG)* 40.6 (2021): 1-13.

物理仿真

- 质点系统
  - 弹簧质点
- 软体仿真
  - 对象：
    - 一维：绳索
    - 二维：薄壳物体、衣服
    - 三维：体软体
  - 现象：
    - 弹性形变与非弹性形变
    - 撕裂、破碎、爆炸
- 刚体仿真
- 流体仿真
  - 理想流体：水、空气
  - 粘性流体、非牛顿流体
  - 拟流体以及相关现象
    - 沙、雪、烟
- 其他
  - 声音
  - 锈蚀、老化、燃烧
  - 电磁
- 多物理场耦合

Kinematic or Dynamic?

Mass

$F = ma$

Position
Velocity
Acceleration
……

Force
Torque
……

Kinematics

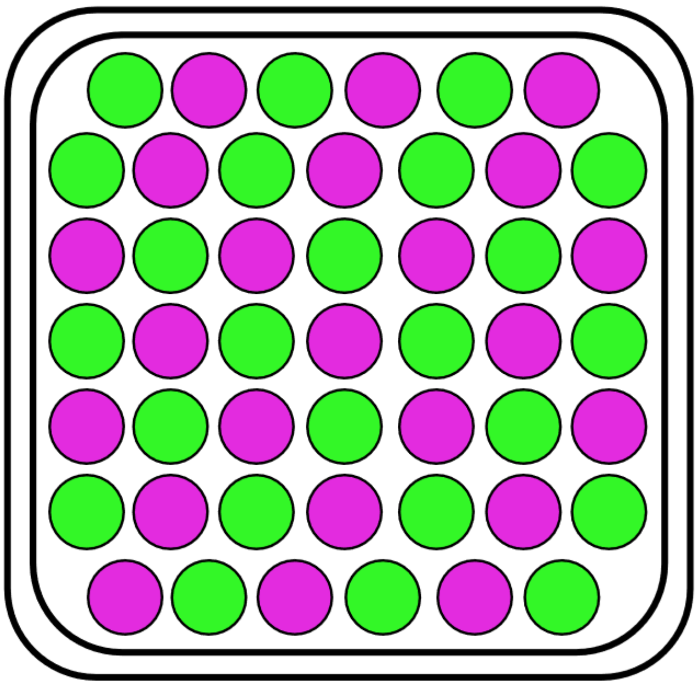
Dynamics

Homogeneous or Heterogeneous

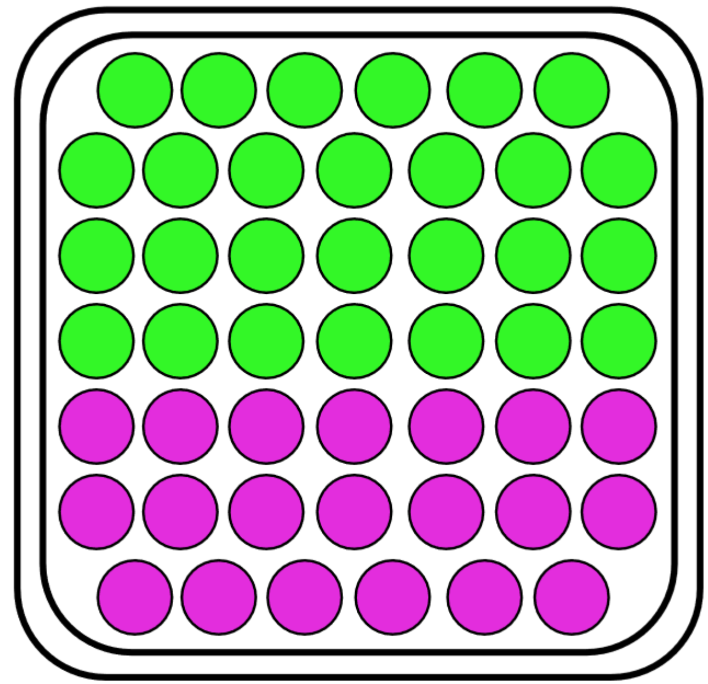
Homogeneous

Heterogeneous

Homogeneous or Heterogeneous



Homogeneous



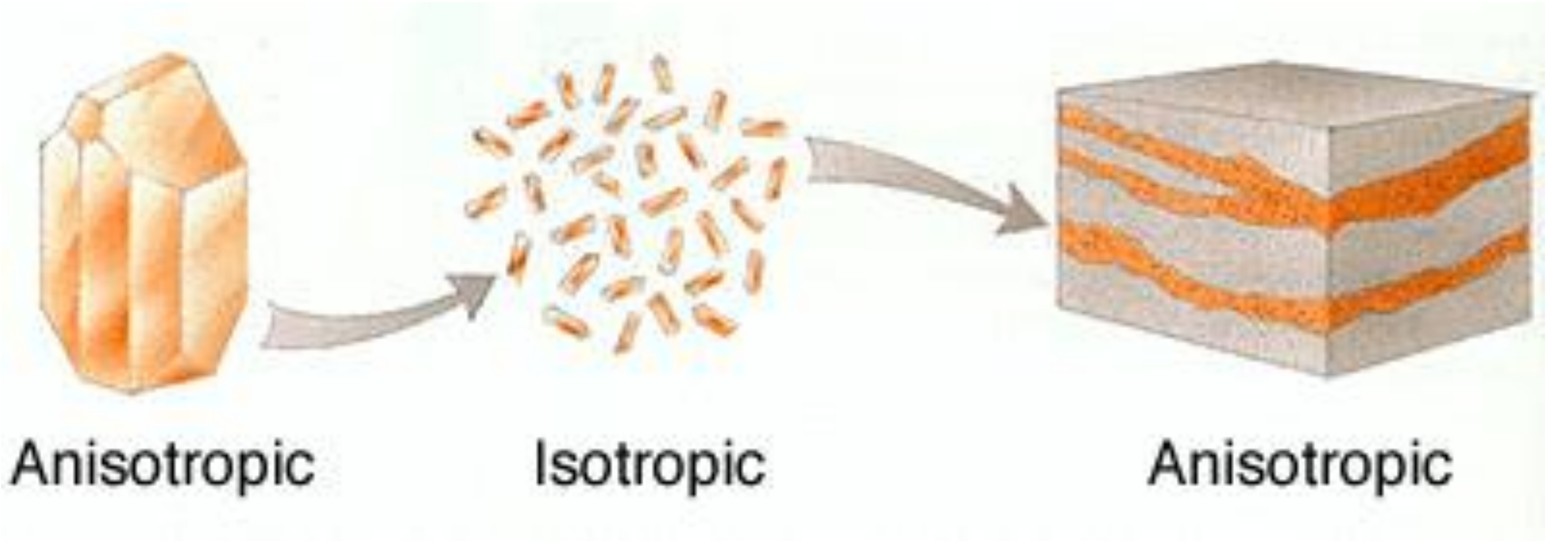
Heterogeneous

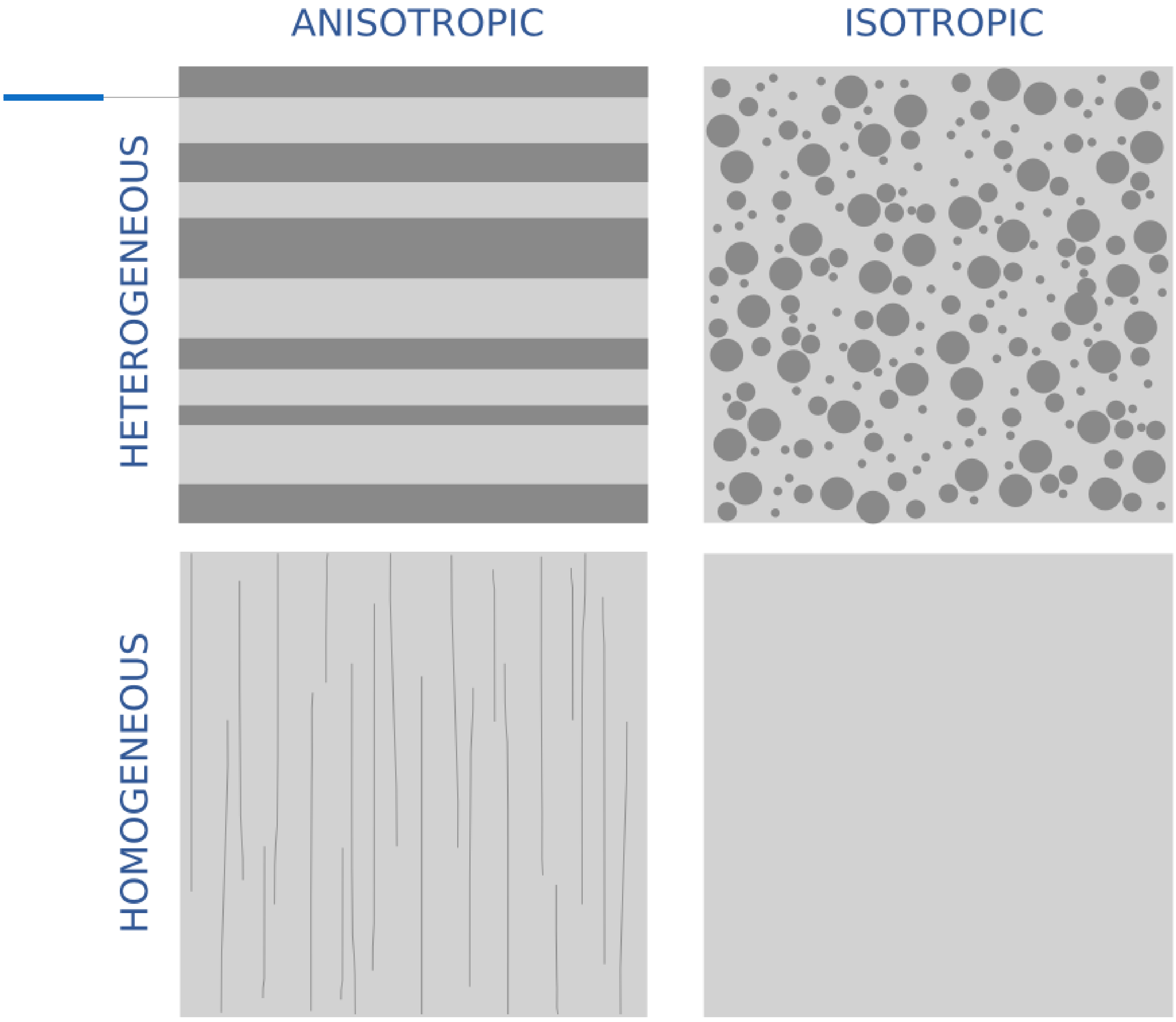
Isotropic or Anisotropic?

Isotropic

Anisotropic

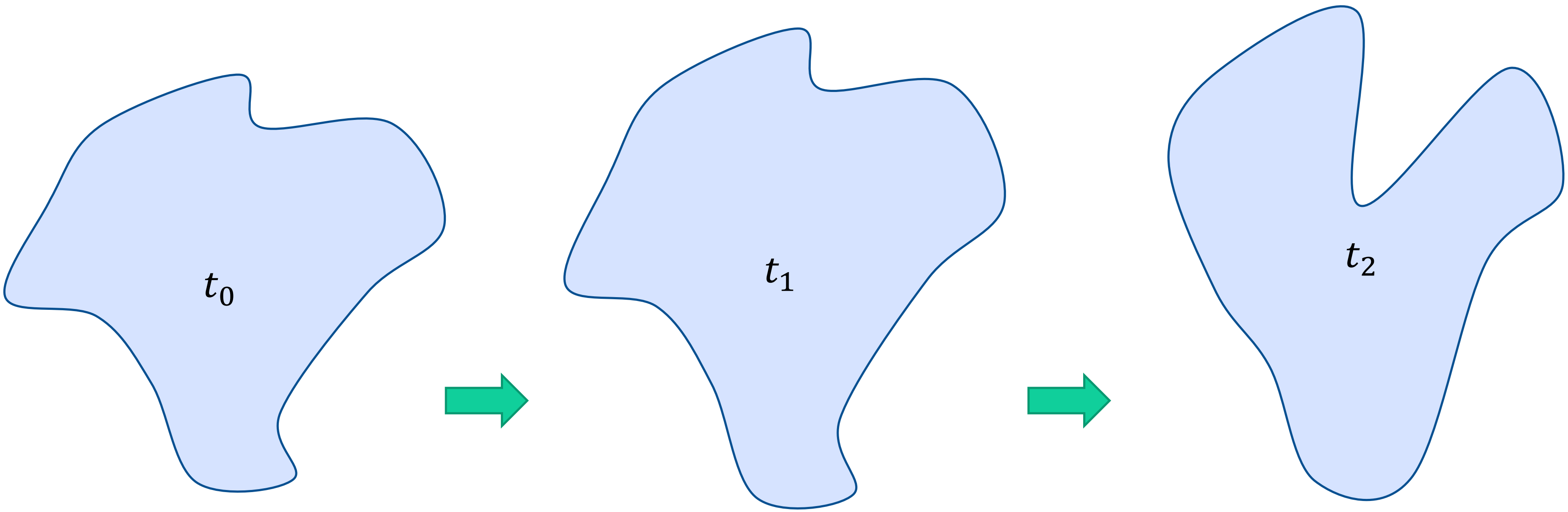
Isotropic or Anisotropic?




ANISOTROPIC
ISOTROPIC
HETEROGENEOUS
HOMOGENEOUS

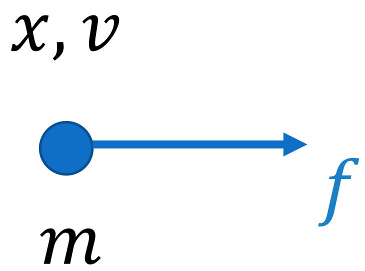
What is Simulation

$$X = X(t)$$



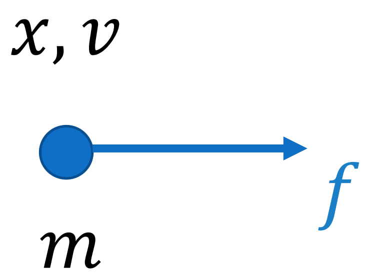
Dynamics of a Particle

$$x = x(t)$$



Dynamics of a Particle

$$x = x(t)$$



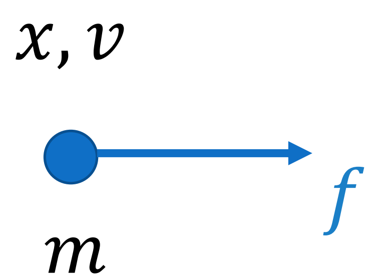
$$f = ma$$

$$a = \dot{v}$$

$$v = \dot{x}$$

Dynamics of a Particle

$$x = x(t)$$



$$f = ma$$

$$a = \dot{v}$$

$$v = \dot{x}$$

$$\Rightarrow$$

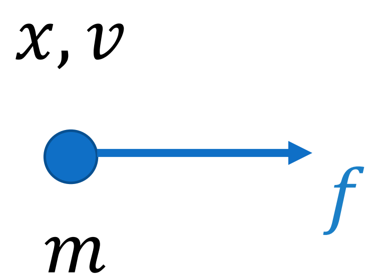
$$a = f/m$$

$$v = v_0 + \int_{t_0}^{t} a dt$$

$$\mathrm{x} = \mathrm{x}_0 + \int_{t_0}^{t} v dt$$

Dynamics of a Particle

$$x = x(t)$$



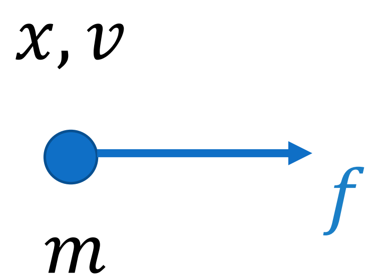
$$f = ma \qquad\qquad a = f/m$$

$$a = \dot{v} \qquad \Rightarrow \qquad v = v_0 + at$$

$$v = \dot{x} \qquad\qquad \mathrm{x} = x_0 + v_0 t + \frac{1}{2} at$$

Dynamics of a Particle

$$x = x(t)$$



$$f = ma$$

$$a = \dot{v}$$

$$v = \dot{x}$$

$$a = f(x, v, t)/m$$

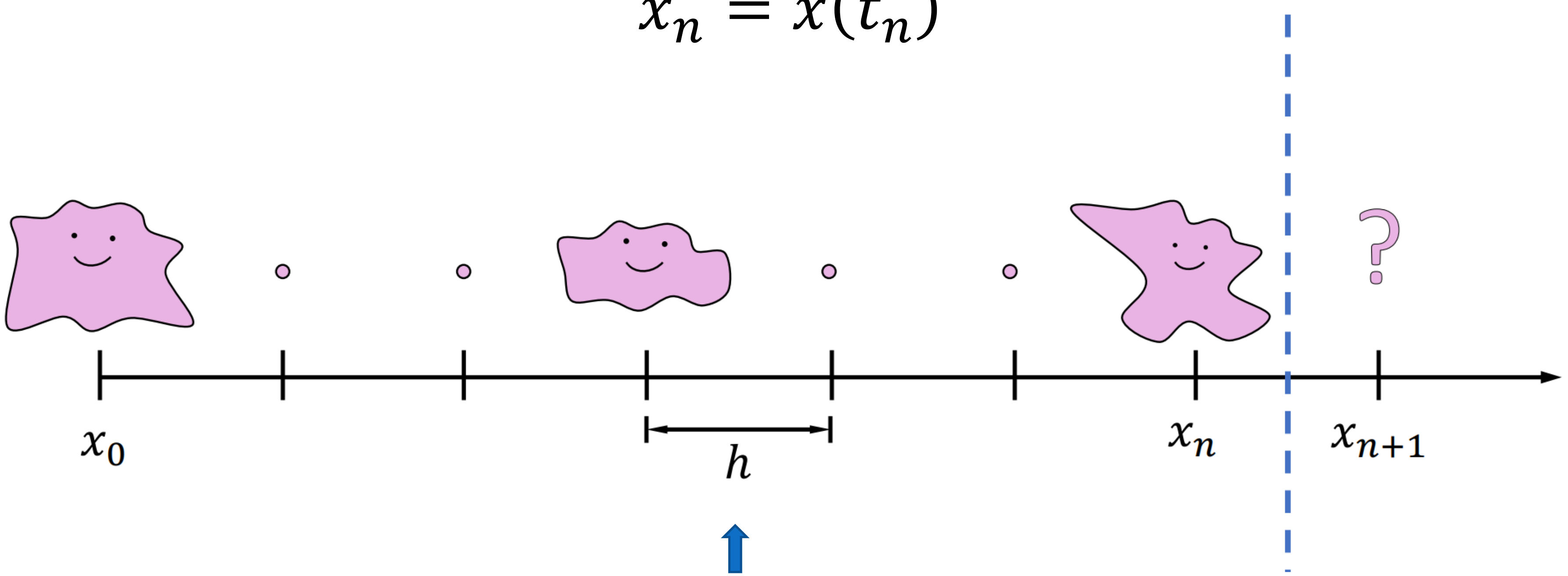
$$v = v_0 + \int_{t_0}^{t} a dt$$

$$\mathrm{x} = \mathrm{x}_0 + \int_{t_0}^{t} v dt$$

Temporal Discretization

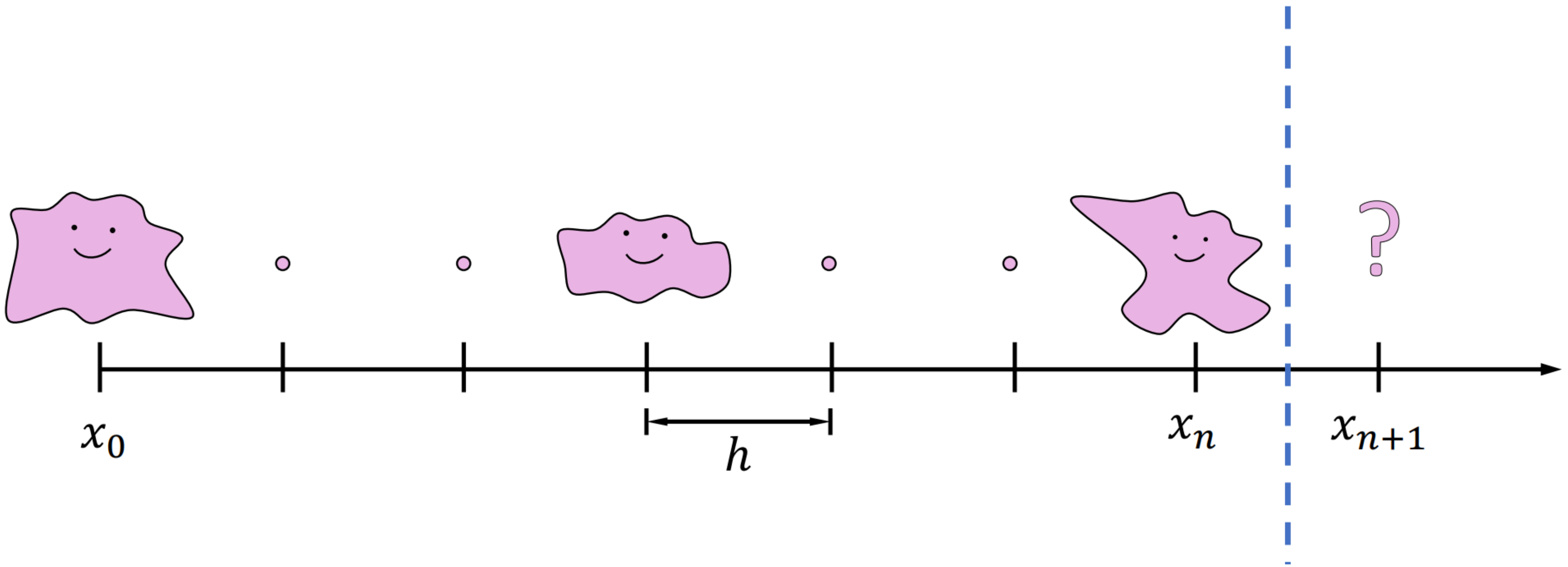
$$x = x(t)$$

$$x_n = x(t_n)$$



Simulation time step

Temporal Discretization



$$a = f(x, v, t)/m$$

$$v = v_0 + \int_{t_0}^{t} a\,dt$$

$$x = x_0 + \int_{t_0}^{t} v\,dt$$

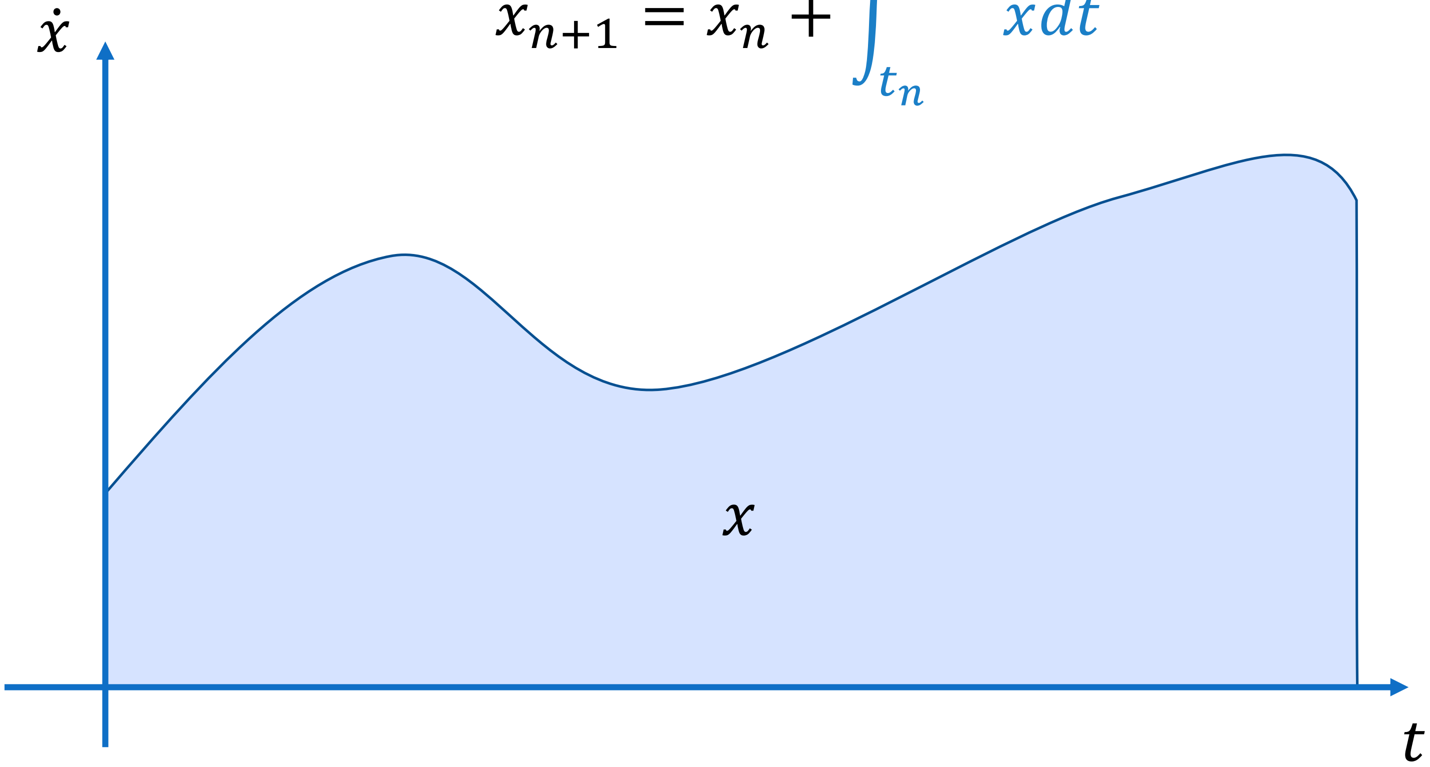
$$a = f(x, v, t)/m$$

$$v_{n+1} = v_n + \int_{t_n}^{t_{n+1}} a\,dt$$

$$x_{n+1} = x_n + \int_{t_n}^{t_{n+1}} \mathrm{v}\,dt$$

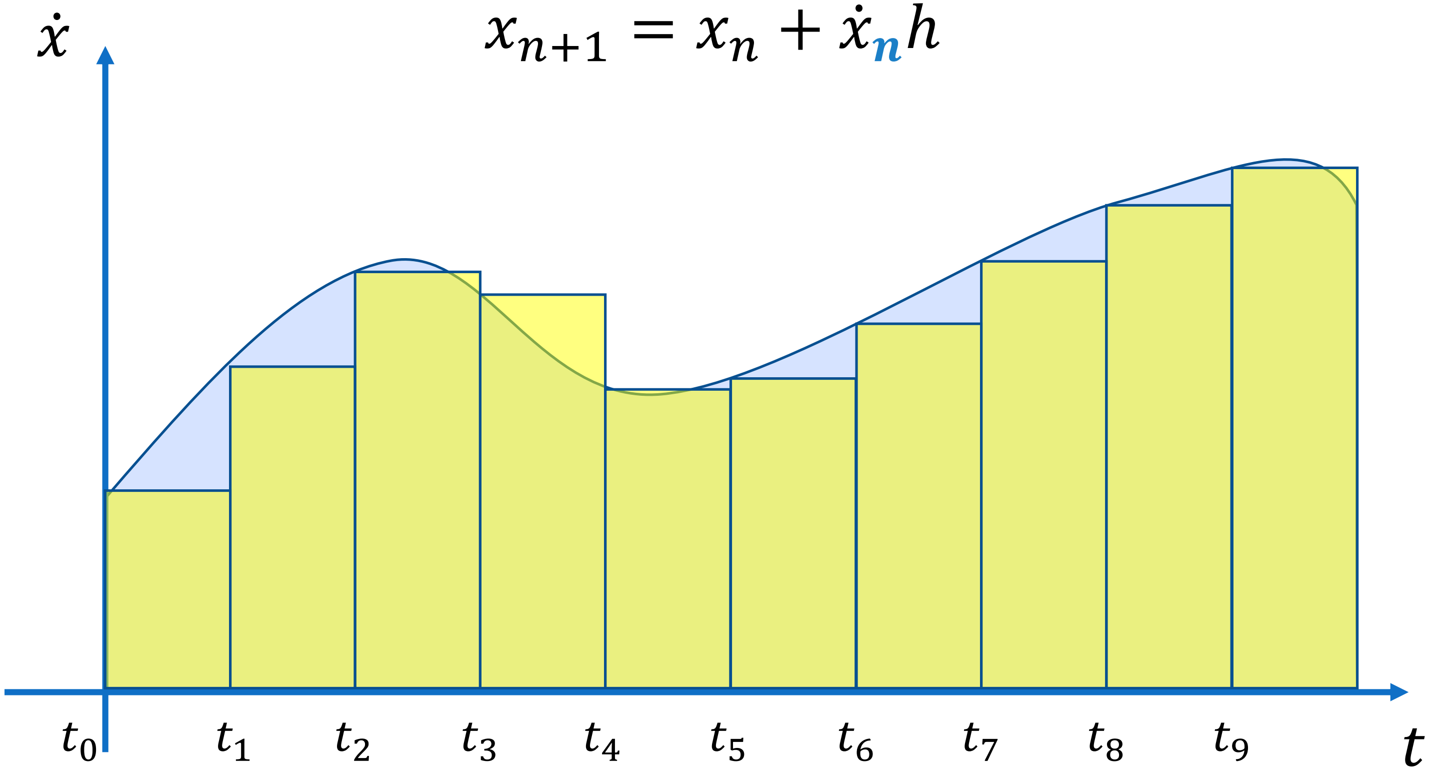
Numerical Integration

$$x_{n+1} = x_n + \int_{t_n}^{t_{n+1}} \dot{x}\,dt$$



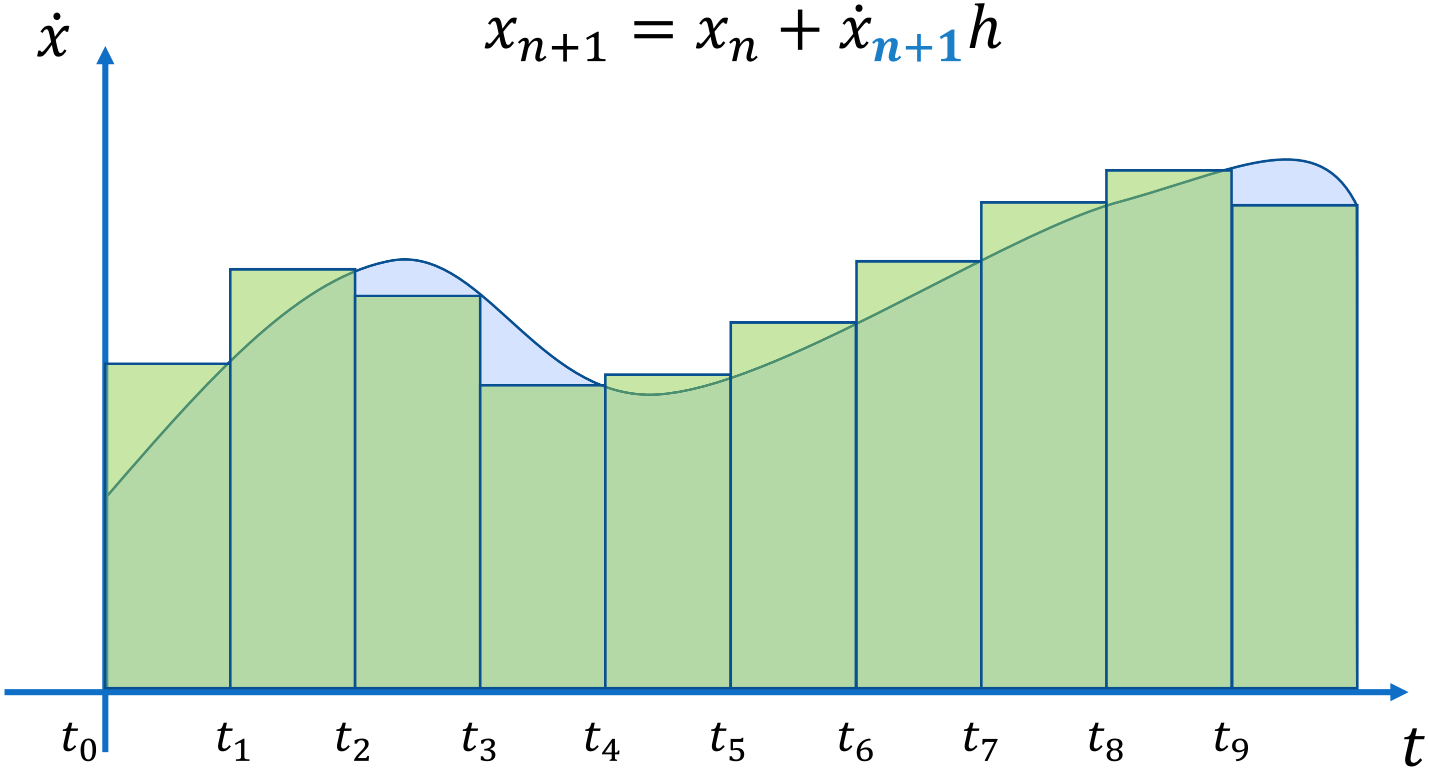
Numerical Integration

$$x_{n+1} = x_n + \dot{x}_n h$$



Numerical Integration

$$x_{n+1} = x_n + \dot{x}_{\boldsymbol{n+1}} h$$



Numerical Integration

- Explicit/Forward Euler Integration

$$v_{n+1} = v_n + a_{\boldsymbol{n}} h$$
$$x_{n+1} = x_n + v_{\boldsymbol{n}} h$$

- Implicit/Backward Euler Integration

$$v_{n+1} = v_n + a_{\boldsymbol{n+1}} h$$
$$x_{n+1} = x_n + v_{\boldsymbol{n+1}} h$$

Numerical Integration

- Explicit/Forward Euler Integration

$$v_{n+1} = v_n + a_{\boldsymbol{n}} h$$
$$x_{n+1} = x_n + v_{\boldsymbol{n}} h$$

- Implicit/Backward Euler Integration

$$v_{n+1} = v_n + a_{\boldsymbol{n+1}} h$$
$$x_{n+1} = x_n + v_{\boldsymbol{n+1}} h$$

Requires information from the future

Numerical Integration

- Explicit/Forward Euler Integration

$$v_{n+1} = v_n + a_{\boldsymbol{n}} h$$
$$x_{n+1} = x_n + v_{\boldsymbol{n}} h$$

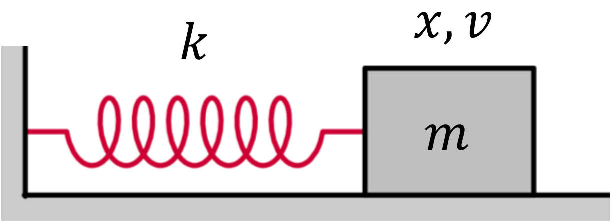
- Implicit/Backward Euler Integration

$$v_{n+1} = v_n + a_{\boldsymbol{n+1}} h$$
$$x_{n+1} = x_n + v_{\boldsymbol{n+1}} h$$

- Symplectic / Semi-implicit Euler Integration

$$v_{n+1} = v_n + a_{\boldsymbol{n}} h$$
$$x_{n+1} = x_n + v_{\boldsymbol{n+1}} h$$

Mass on a Spring



$$f = -kx$$

Explicit Euler Integration

$$v_{n+1} = v_n - \frac{kx_n}{m}h$$
$$x_{n+1} = x_n + v_n h$$

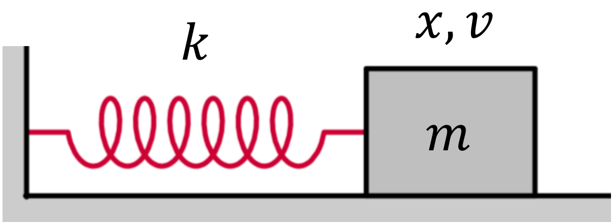
Semi-implicit Euler Integration

$$v_{n+1} = v_n - \frac{kx_n}{m}h$$
$$x_{n+1} = x_n + v_{n+1} h$$

Implicit Euler Integration

$$v_{n+1} = v_n - \frac{kx_{n+1}}{m}h$$
$$x_{n+1} = x_n + v_{n+1} h$$

Mass on a Spring



$$f = -kx$$

$$\hat{k} = k/m$$

Explicit Euler Integration

$$\begin{bmatrix} v_{n+1} \\ x_{n+1} \end{bmatrix} = \begin{bmatrix} 1 & -\hat{k}h \\ h & 1 \end{bmatrix} \begin{bmatrix} v_n \\ x_n \end{bmatrix}$$

$$\det A = 1 + \hat{k}h^2$$

Semi-implicit Euler Integration

$$\begin{bmatrix} v_{n+1} \\ x_{n+1} \end{bmatrix} = \begin{bmatrix} 1 & -\hat{k}h \\ h & 1 - \hat{k}h^2 \end{bmatrix} \begin{bmatrix} v_n \\ x_n \end{bmatrix}$$

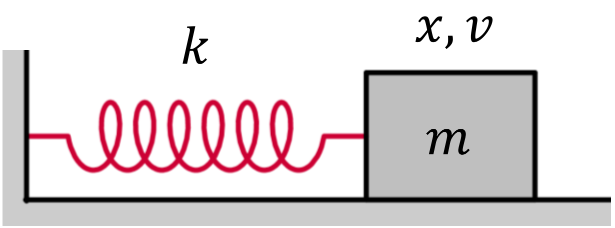
$$\det A = 1$$

Implicit Euler Integration

$$\begin{bmatrix} v_{n+1} \\ x_{n+1} \end{bmatrix} = \frac{1}{1 + \hat{k}h^2} \begin{bmatrix} 1 & -\hat{k}h \\ h & 1 \end{bmatrix} \begin{bmatrix} v_n \\ x_n \end{bmatrix}$$

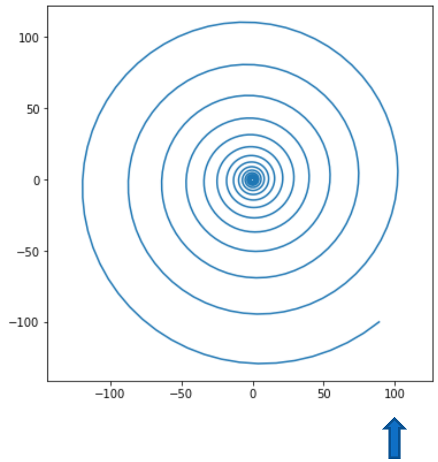
$$\det A = \frac{1}{1 + \hat{k}h^2}$$

Mass on a Spring

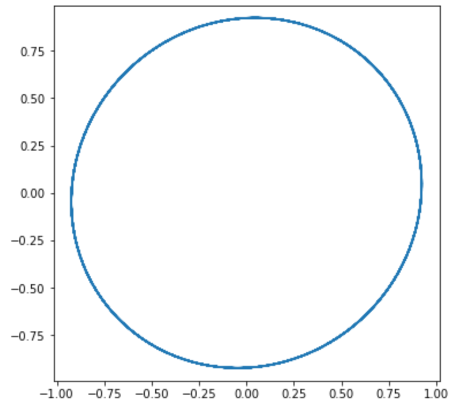


$$f = -kx$$

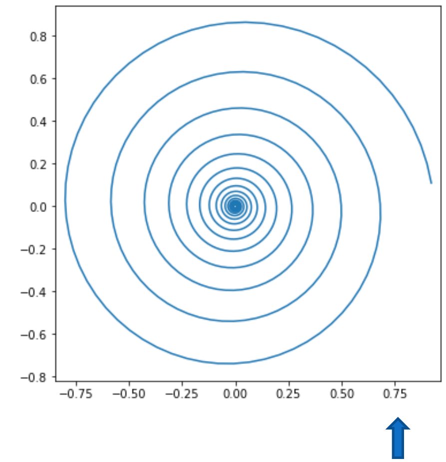
Explicit Euler Integration



Semi-implicit Euler Integration



Implicit Euler Integration



Numerical Integration

- Explicit/Forward Euler
  Symplectic/Semi-implicit Euler
  - Fast, no need to solve equations
  - Can be unstable under large time step

- Implicit/Backward Euler
  - Rock stable (unconditionally)
  - Slow, need to solve a large problem

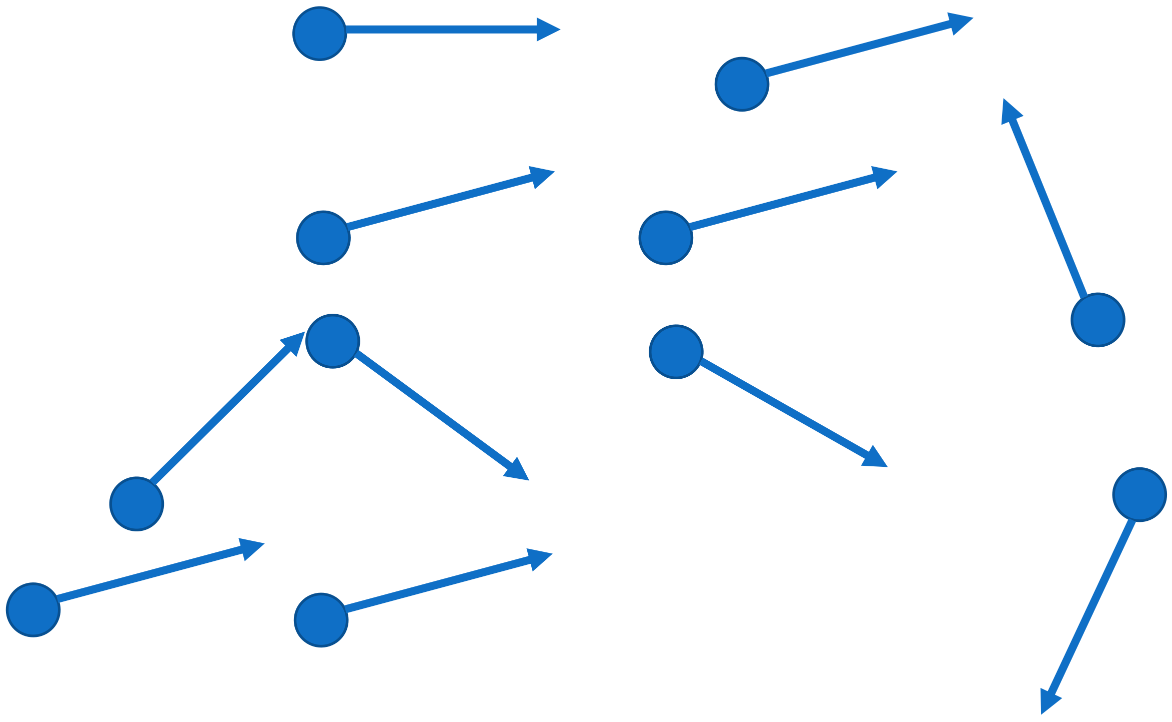
More Advanced Integration

- Runge–Kutta methods
- Variational integration

Particle Systems System

Particle Systems

- A set of (identical) simulated particles $\{x_i\}$



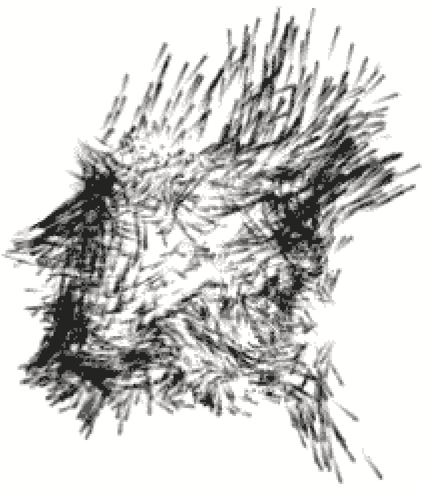
Particle Systems

- Simulation Loop
  - Clear forces
    - Prevent force accumulation
  - Calculate forces
    - Sum all forces into accumulators
  - Update
    - Loop over particles, update x_i and v_i using the corresponding integrator

$$
\boxed{\begin{array}{c} x \\ v \\ f \\ m \end{array}}
$$

Particle Systems

- Forces
  - Constant
    - Gravity
  - Position/time dependent
    - Force field
  - Velocity-dependent
    - Damping, dragging
  - Others
    - Contacts, bouncing
    - Spring



Realtime?

- A few related concepts
  - Wall clock / real world time T
  - Simulation clock t
    - Advance $\boldsymbol{h}$ seconds every simulation step
  - $t \geq T$ → realtime simualtion
- Synchronization between the two worlds
  - Sleep when necessary

Example: Particle System in Unity

Everything to know about the PARTICLE SYSTEM
https://www.youtube.com/watch?v=FEA1wTMJAR0

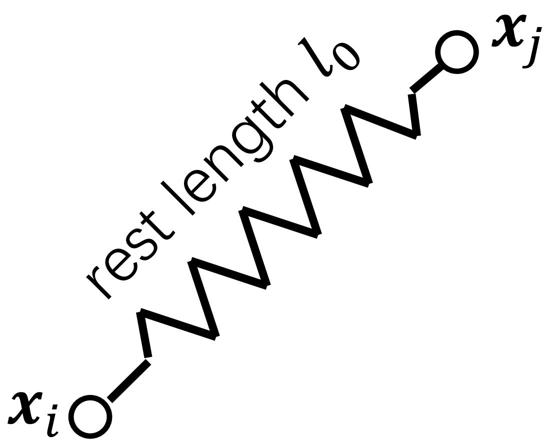
Any Questions?

Mass-Spring System

A few slides are edited from GAMES-103

Huamin Wang. 2021. ***GPU-based simulation of cloth wrinkles at submillimeter levels***. *ACM Trans. Graph.* 40, 4 (July 2021)

Mass Spring System

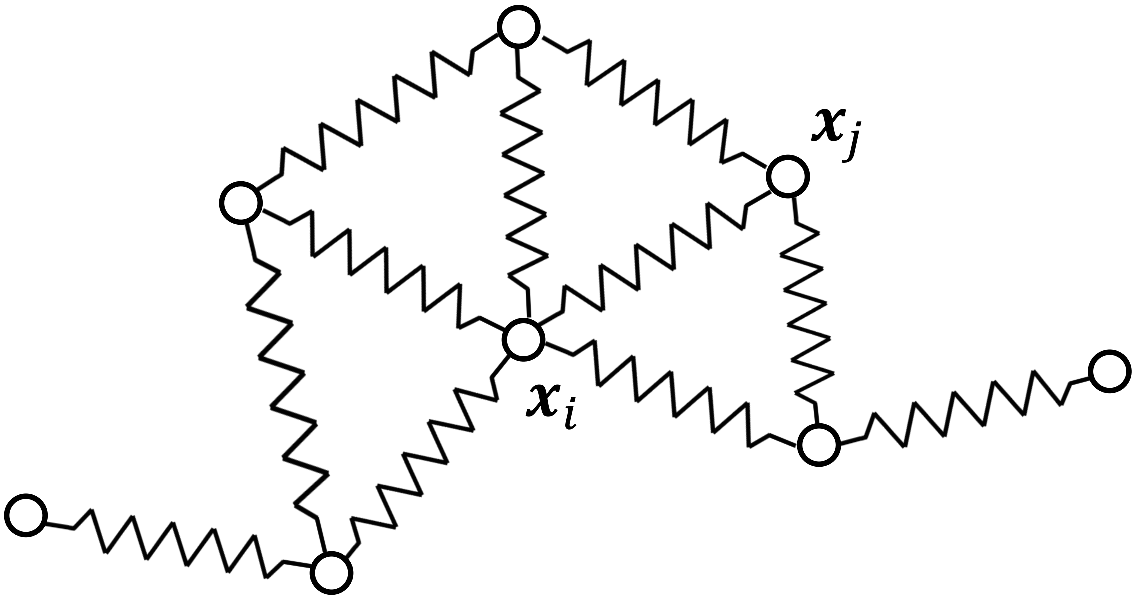


$$\boxed{\boldsymbol{f}_{ij} = -\boldsymbol{f}_{ji}}$$

$$f_{ij} = -k\left(\|x_i - x_j\| - l_0\right)\frac{x_i - x_j}{\|x_i - x_j\|}$$

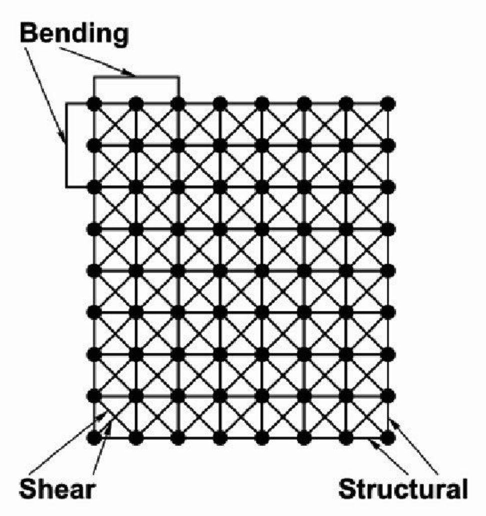
$$f_{ji} = -k\left(\|x_j - x_i\| - l_0\right)\frac{x_j - x_i}{\|x_j - x_i\|}$$

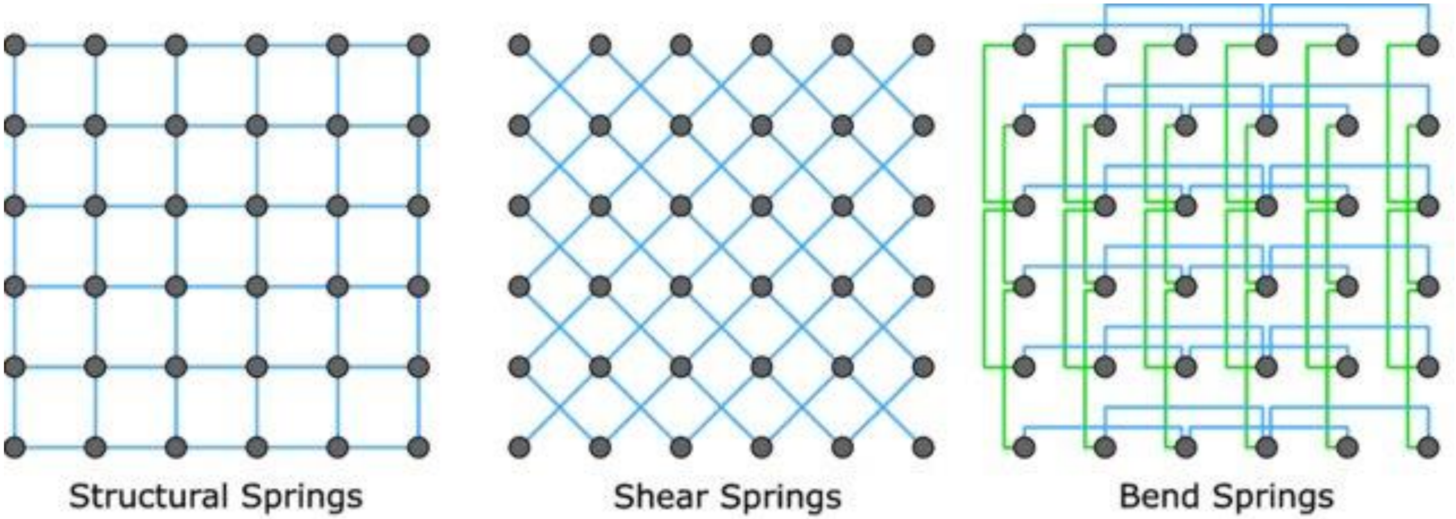
Mass Spring System



$$f_i = \sum_{j \in N(i)} f_{ij}$$

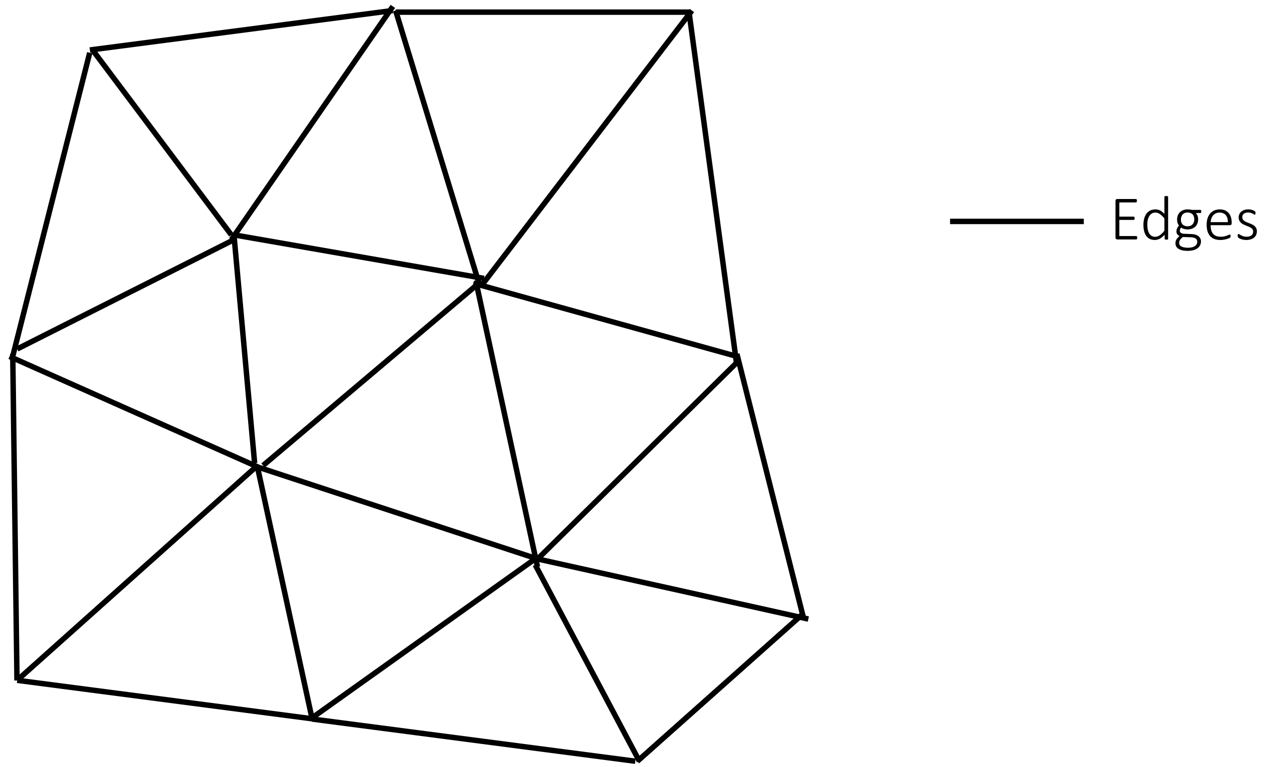
Structured Network





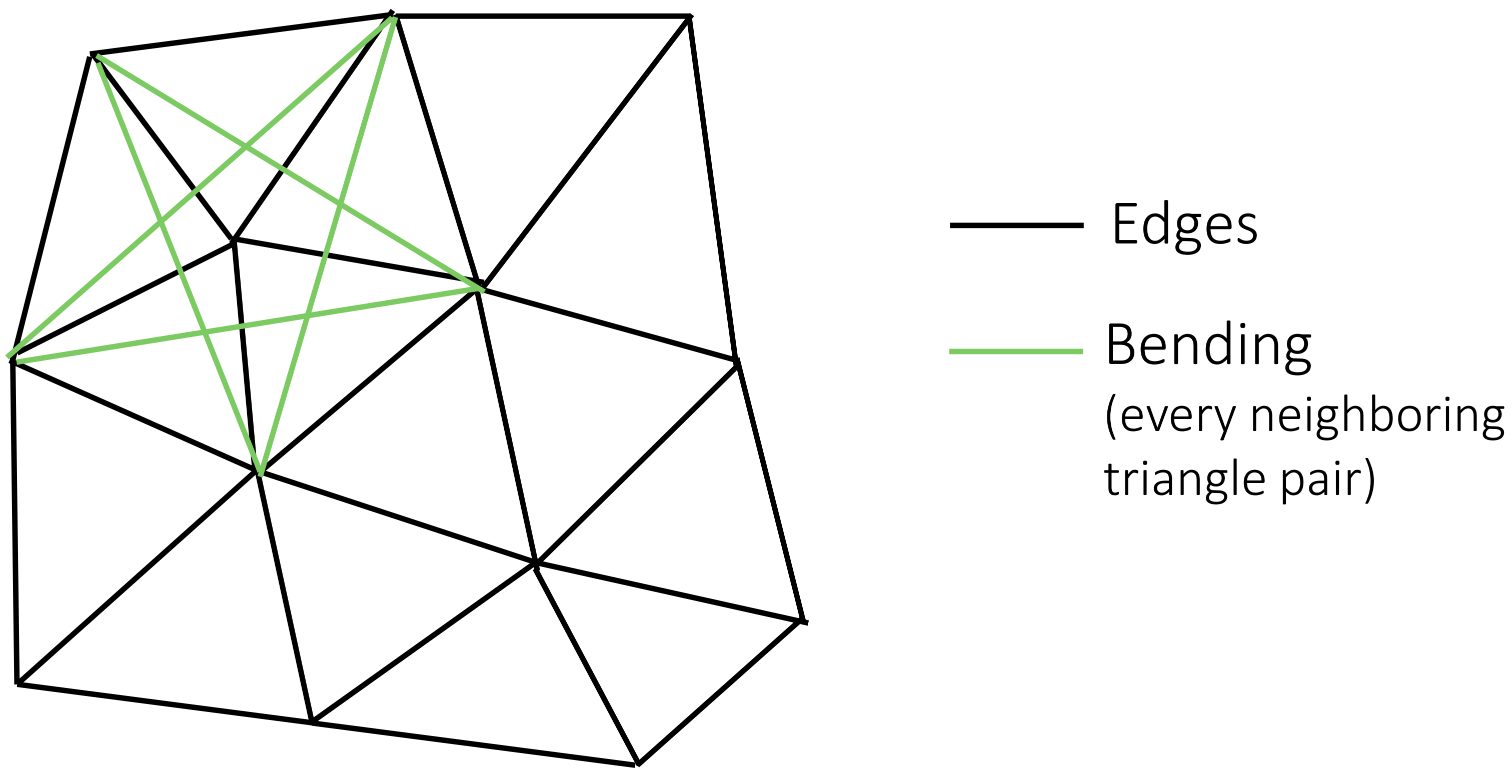
Structured Network

- For a triangle mesh



Structured Network

- For a triangle mesh



Mass Spring System

- Simulation Loop
  - Clear forces
    - Prevent force accumulation
  - Calculate forces
    - For every edge ij
      - Compute f_{ij}
      - $f_i \mathrel{+}= f_{ij}, \qquad f_j \mathrel{-}= f_{ij}$
  - Update
    - Loop over particles, update x_i and v_i using the corresponding integrator

$$x$$
$$v$$
$$f$$
$$M$$

Update

- (Semi-) Explicit Euler Integration
  - Need small time step

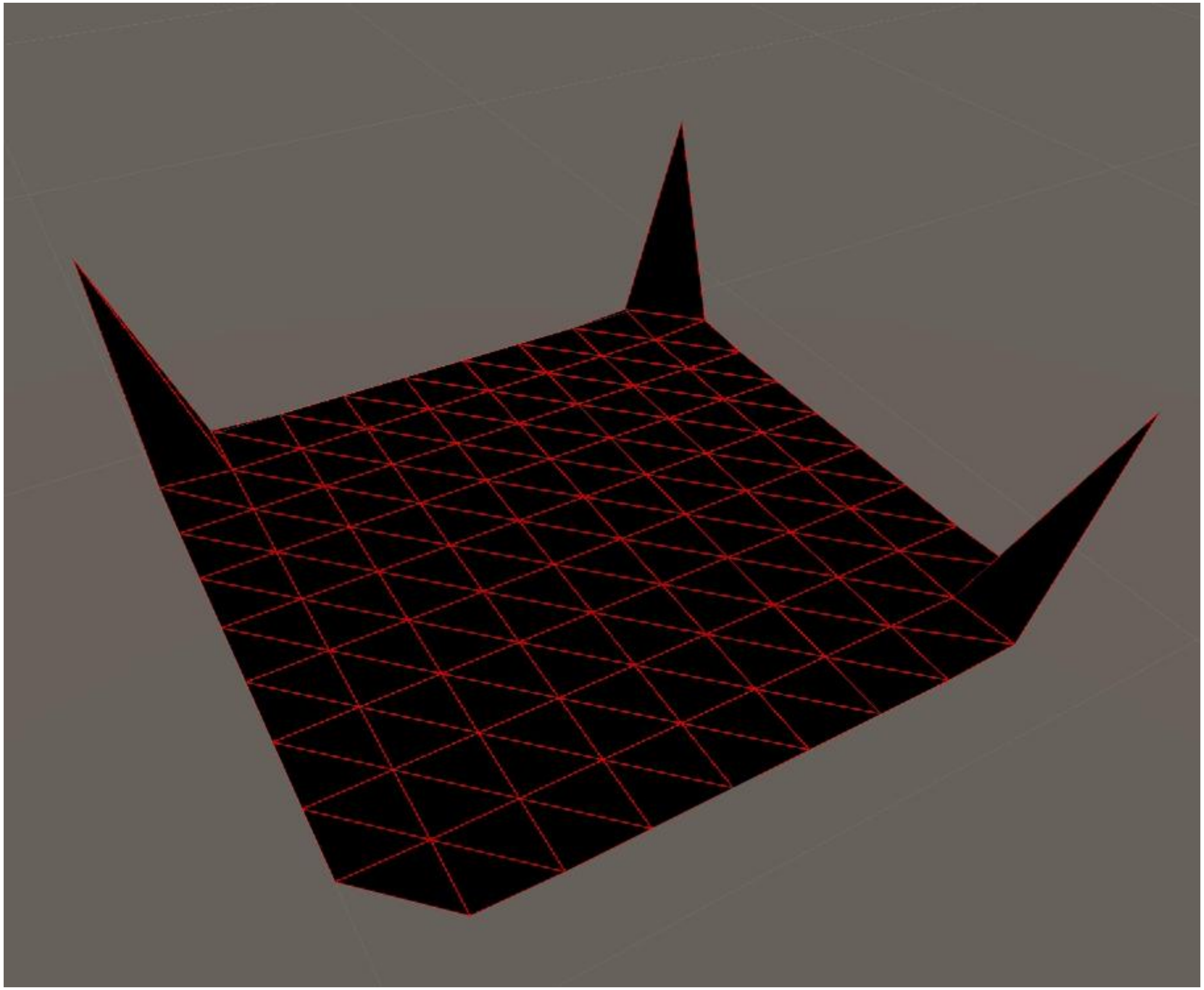
Explicit Euler Integration

$$v_{n+1} = v_n + hM^{-1}f_n h$$
$$x_{n+1} = x_n + v_{\boldsymbol{n}} h$$

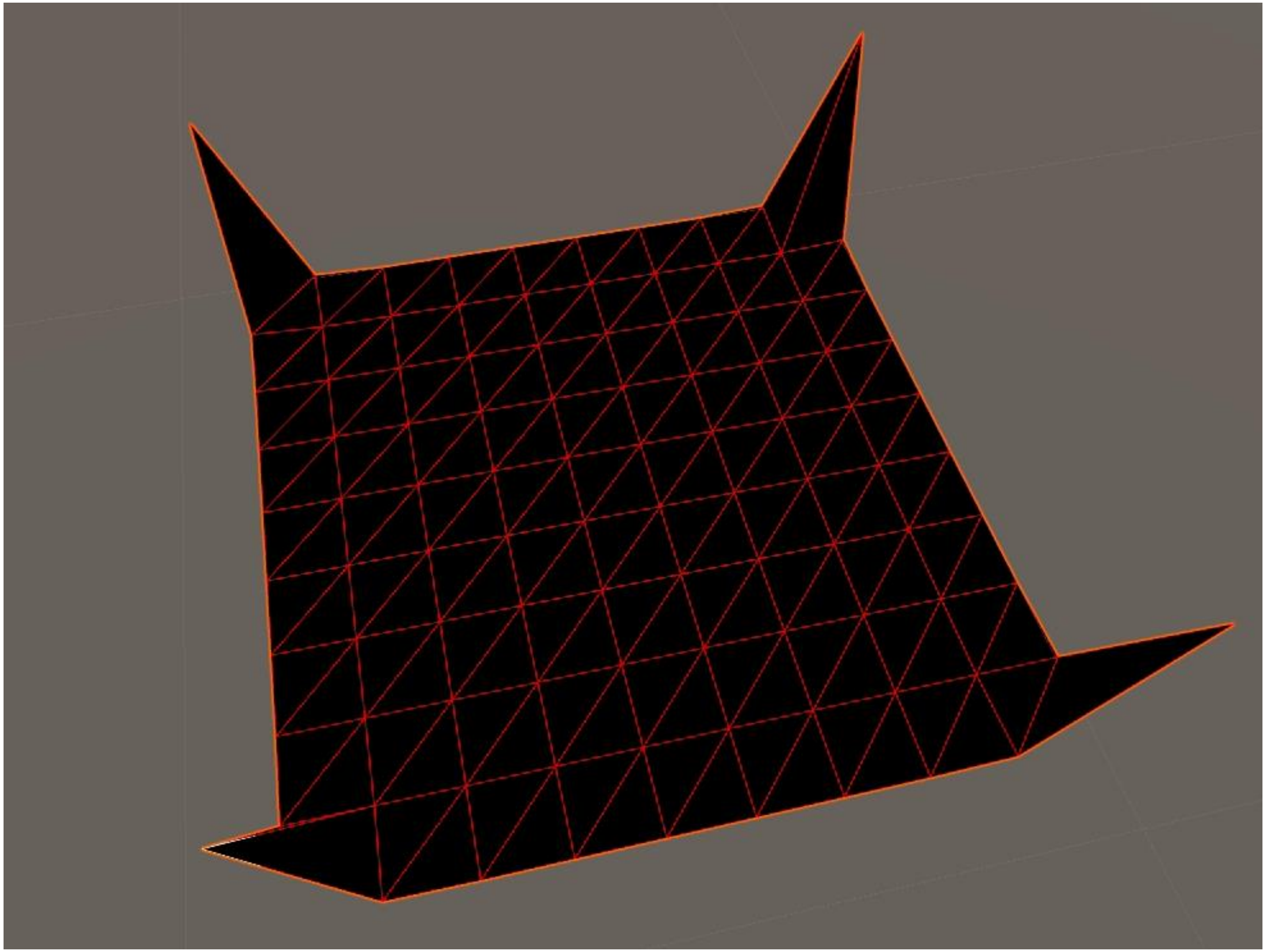
Semi-implicit Euler Integration

$$v_{n+1} = v_n + hM^{-1}f_n$$
$$x_{n+1} = x_n + v_{\boldsymbol{n+1}} h$$

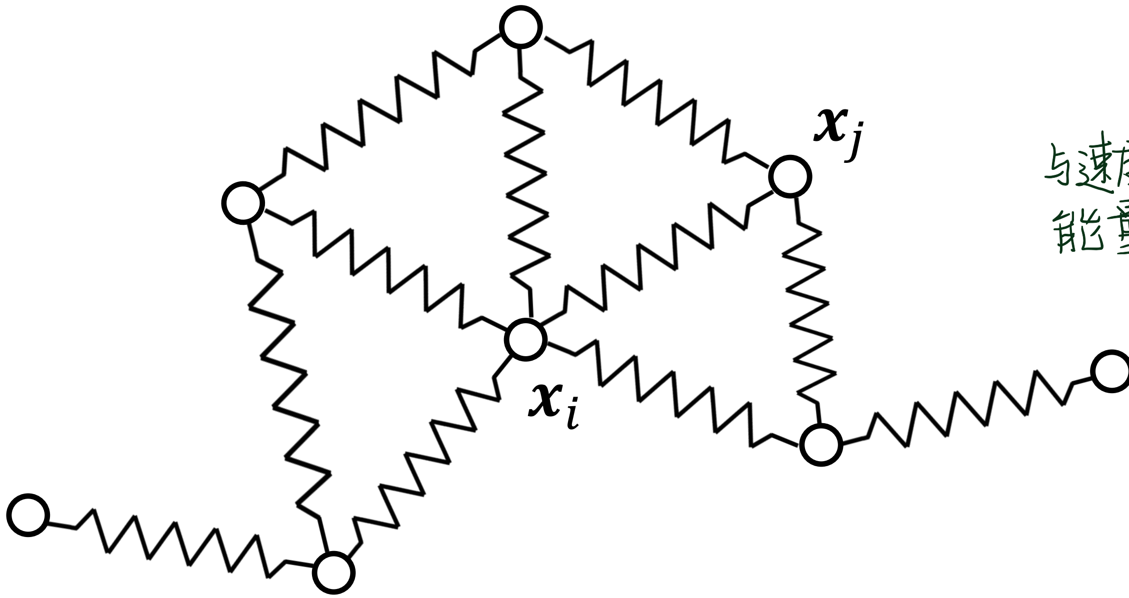
Damping?



Damping?



Damping?

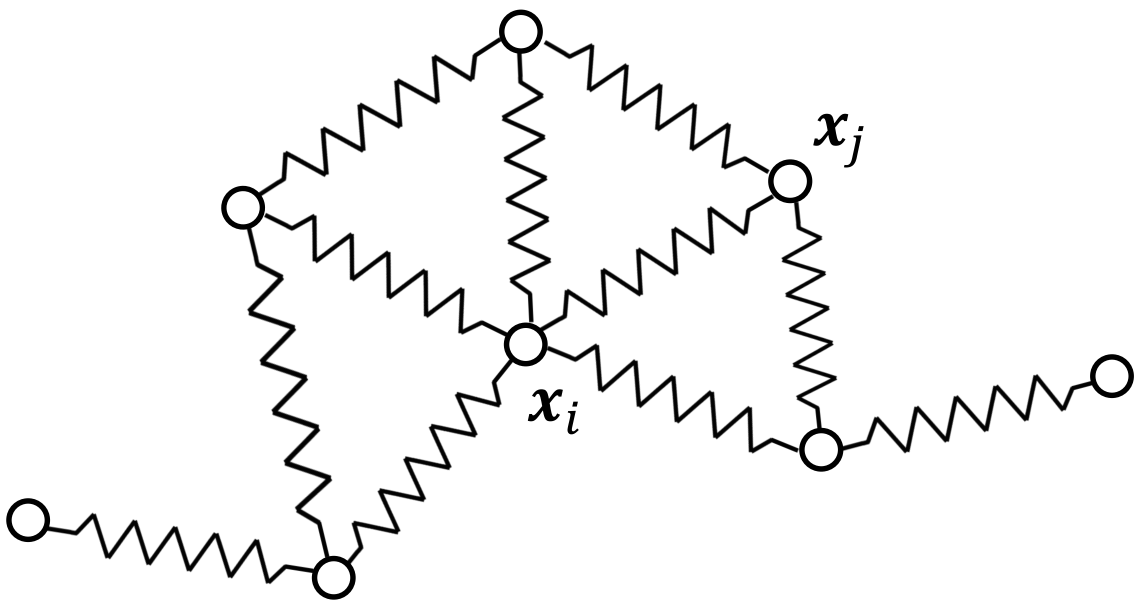


与速度相关
能量相关

$$f_i = \sum_{j \in N(i)} f_{ij} - k_d v_i$$

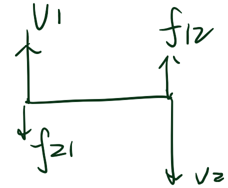
类似空阻

Damping?



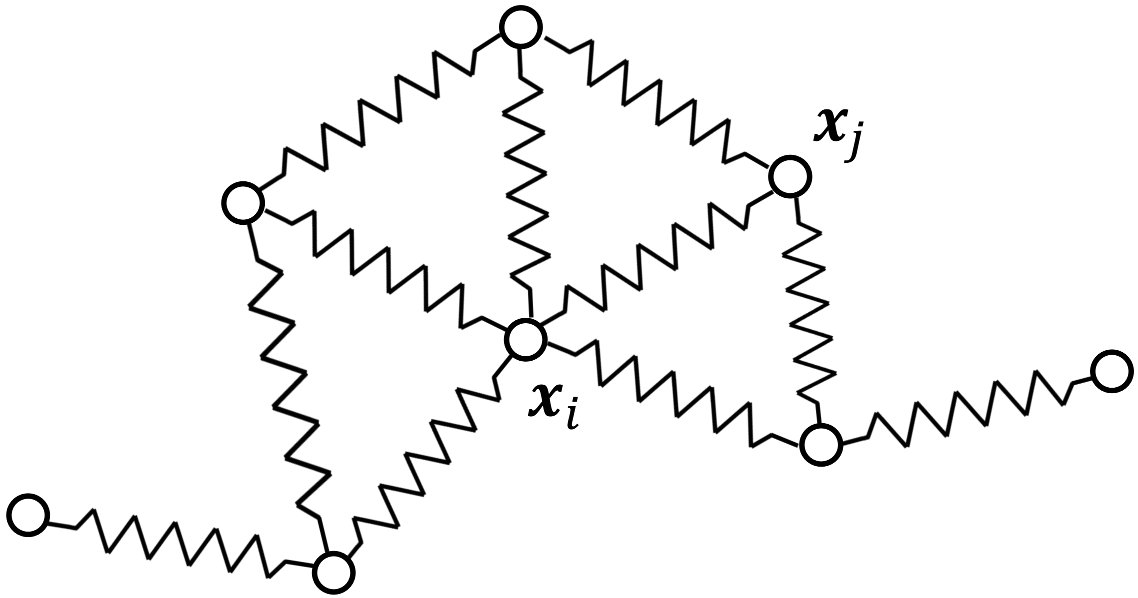
$$f_i = \sum_{j \in N(i)} f_{ij} - k_d (v_i - v_j)$$

不抖动
阻止相对运动
合外力为0



但有力矩 可能自转
且加快

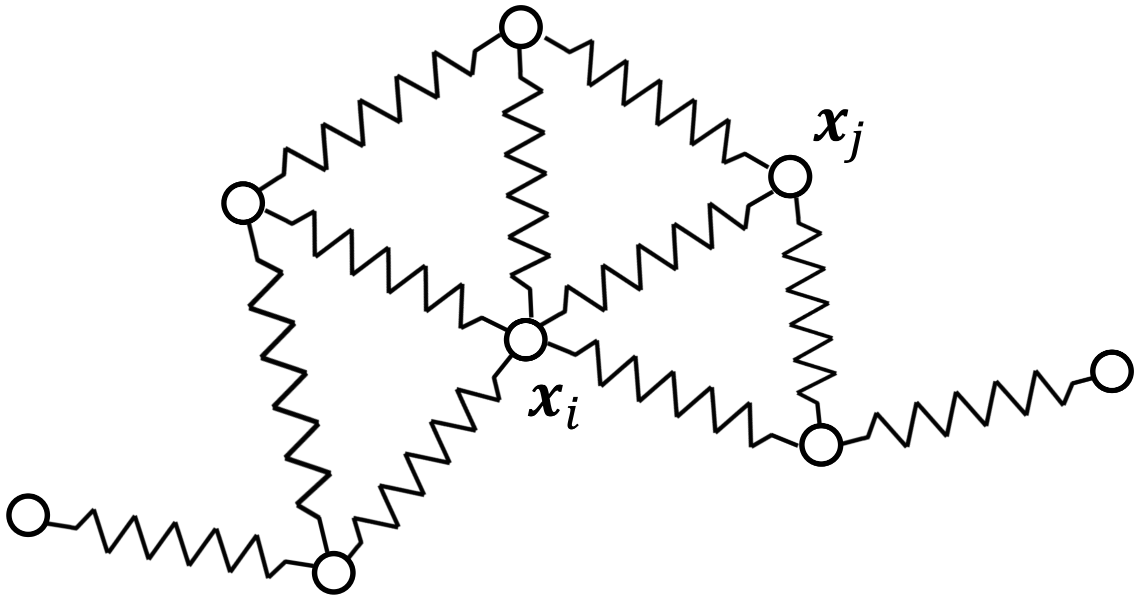
Damping?



平衡力矩，只在质心加力

$$f_i = \sum_{j \in N(i)} f_{ij} - k_d (v_i - v_j) \cdot \frac{x_i - x_j}{\|x_i - x_j\|}$$

Damping?



每个小三角形的质心

$$f_i = \sum_{j \in N(i)} f_{ij} - f_d(v)$$

Update

- (Semi-) Explicit Euler Integration
  - Need small time step h
  - May be very small when k is large

Explicit Euler Integration

$$v_{n+1} = v_n + hM^{-1}f_n h$$
$$x_{n+1} = x_n + v_{\boldsymbol{n}} h$$

Semi-implicit Euler Integration

$$v_{n+1} = v_n + hM^{-1}f_n$$
$$x_{n+1} = x_n + v_{\boldsymbol{n+1}} h$$

Implicit Integration

Implicit Euler Integration

$$v_{n+1} = v_n + M^{-1} f_{\boldsymbol{n+1}} h$$

$$x_{n+1} = x_n + v_{\boldsymbol{n+1}} h$$

$$\Leftarrow \quad f_{\boldsymbol{n+1}} = f(\boldsymbol{x}_{\boldsymbol{n+1}})$$

$$x_{n+1} = x_n + h v_n + h^2 M^{-1} f(x_{n+1})$$

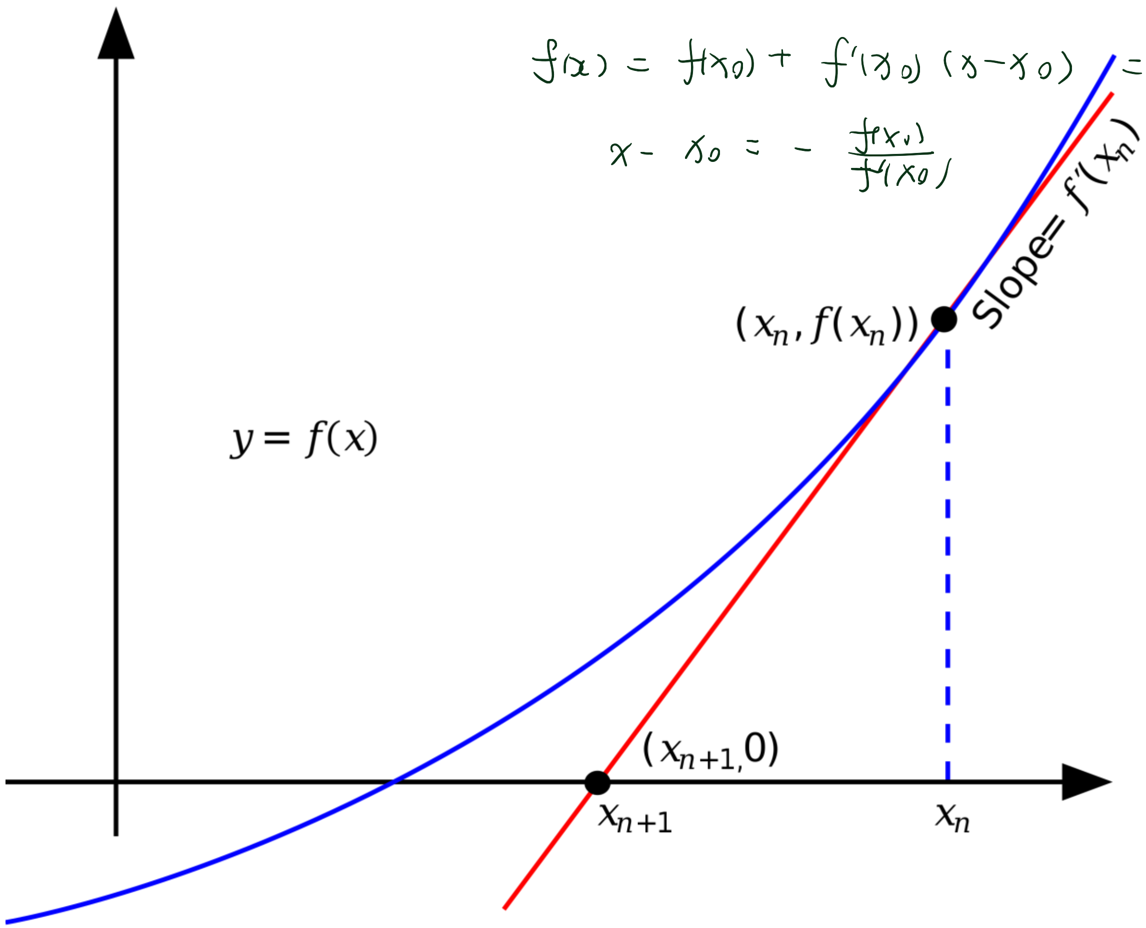
$f_{ij} = -k(\|x_{ij}\| - l_0)\frac{x_{ij}}{\|x_{ij}\|}$

$x_{ij} = x_i - x_j$

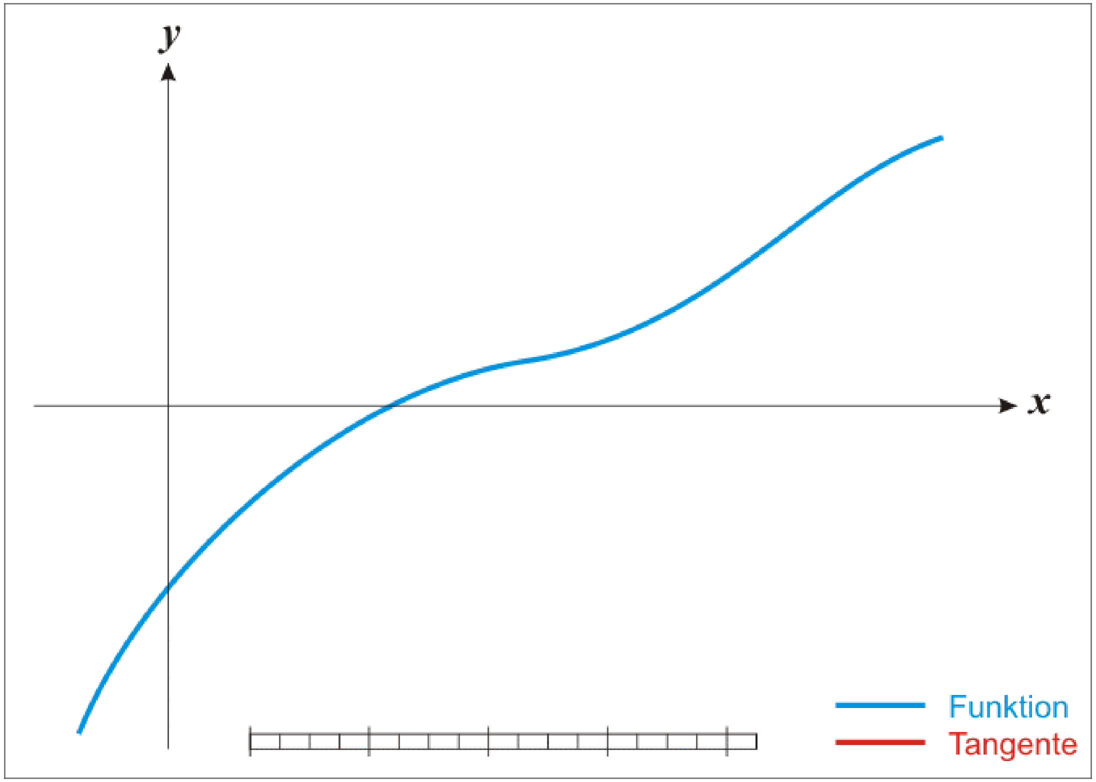
Newton-Raphson Method

$$f(x) = f(x_0) + f'(x_0)(x - x_0) = 0$$

$$x - x_0 = -\frac{f(x_0)}{f'(x_0)}$$



Newton-Raphson Method



$$x_{n+1} = x_n - \frac{f(x_n)}{f'(x_n)}$$

初始值选择

Simulation by Newton's Method

$$x_{n+1} = x_n + hv_n + h^2 M^{-1} f(x_{n+1})$$

$g(x) = 0$

$$F\left(x_{n+1}^{(k)}\right) = \frac{1}{h^2} M \left(x_{n+1}^{(k)} - x_n - hv_n\right) - f(x_{n+1}^{(k)})$$

$$F'\left(x_{n+1}^{(k)}\right) = \frac{1}{h^2} M - f'\left(x_{n+1}^{(k)}\right)$$

Initialize $\boldsymbol{x}_{n+1}^{(0)}$, often as $\boldsymbol{x}_n$ or $\boldsymbol{x}_n + h\boldsymbol{v}_n$

For $k = 0 \ldots K$

Solve $\left(\frac{1}{h^2} M - f'\left(x_{n+1}^{(k)}\right)\right) \Delta \boldsymbol{x} = -\frac{1}{h^2} M \left(x_{n+1}^{(k)} - x_n - hv_n\right) + f(x_{n+1}^{(k)})$

简单弹簧 $f(x) = -kx$. $f'(x) = k$

$\boldsymbol{x}_{n+1}^{(k+1)} \leftarrow \boldsymbol{x}_{n+1}^{(k)} + \Delta \boldsymbol{x}$

If $\|\Delta \boldsymbol{x}\|$ is small then break

$\boldsymbol{x}_{n+1} \leftarrow \mathbf{x}^{(k+1)}$

$\boldsymbol{v}_{n+1} \leftarrow (\boldsymbol{x}_{n+1} - \boldsymbol{x}_n) / \boldsymbol{h}$

Implicit Integration as Optimization

$$x_{n+1} = x_n + hv_n + h^2 M^{-1} f(x_{n+1})$$

$$\Updownarrow$$

$$x_{n+1} = \text{argmax}_x \, F(x)$$

$$= \text{argmax}_x \frac{1}{2h^2} \|x - x_n - hv_n\|_M^2 + E(x)$$

$$\boxed{\|\mathbf{x}\|_{\mathbf{M}}^2 = \mathbf{x}^{\mathrm{T}}\mathbf{M}\mathbf{x}}$$

Elastic Energy

Newton-Raphson Method for Optimization

优化问题 : 求 $F'(x)=0$ 的 x

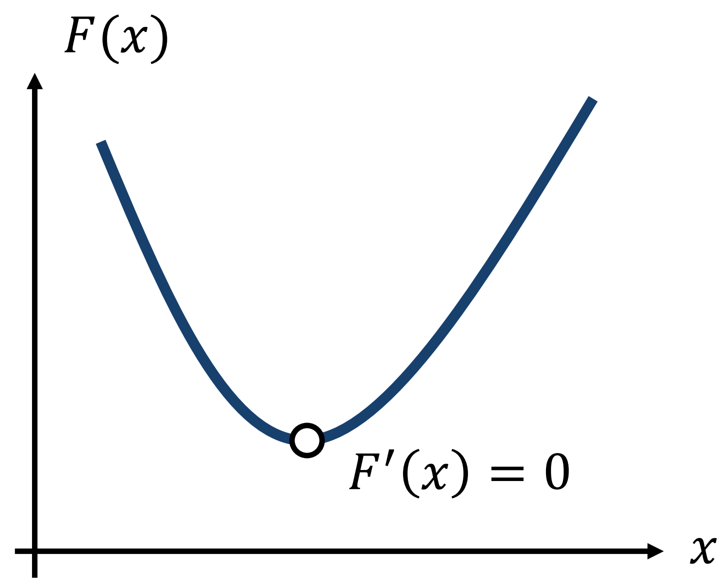
$\Updownarrow$ 牛顿法

$$x_{n+1} = \text{argmax}_x \, F(x) \; = \text{argmax}_x \frac{1}{2h^2}\|x - x_n - hv_n\|_M^2 + E(x)$$

Optimality condition:

$$x - x^{(k)} = -\frac{F'(x^{(k)})}{F''(x^{(k)})}$$

$$0 = F'(x) \approx F'\left(x^{(k)}\right) + F''\left(x^{(k)}\right)\left(x - x^{(k)}\right)$$



Newton-Raphson Method for Optimization

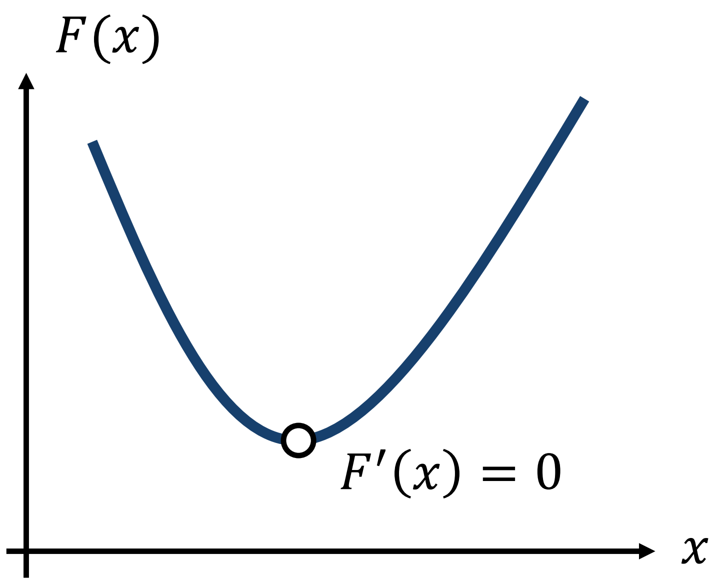
$$x_{n+1} = \mathrm{argmax}_x\, F(x) = \mathrm{argmax}_x \frac{1}{2h^2}\|x - x_n - hv_n\|_M^2 + E(x)$$

Optimality condition:

$$0 = F'(x) \approx F'\left(x^{(k)}\right) + F''\left(x^{(k)}\right)\left(x - x^{(k)}\right)$$

$$\Delta x \leftarrow -\left(F''\left(x^{(k)}\right)\right)^{-1} F'\left(x^{(k)}\right)$$

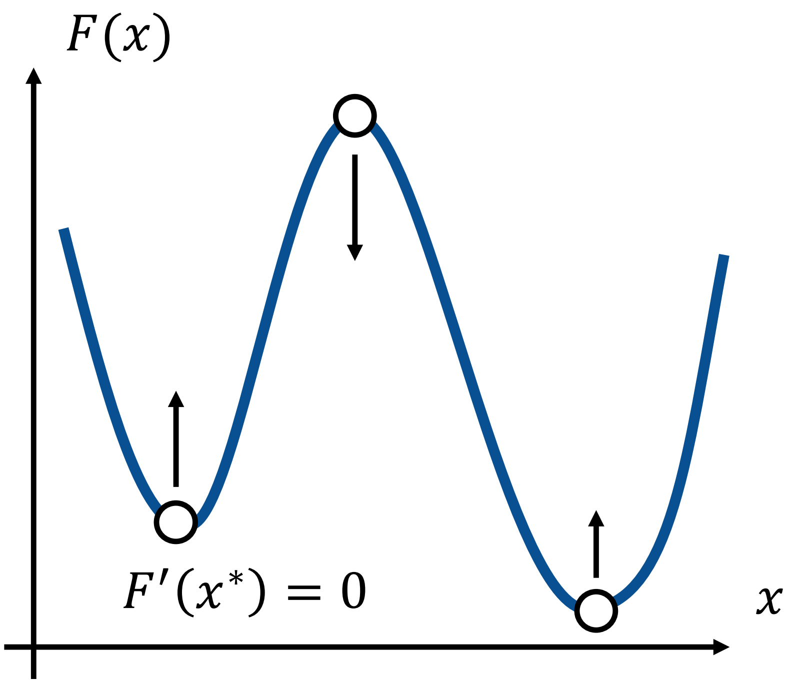
$$x^{(k+1)} \leftarrow x^{(k)} + \Delta x$$



Hessian Matrix

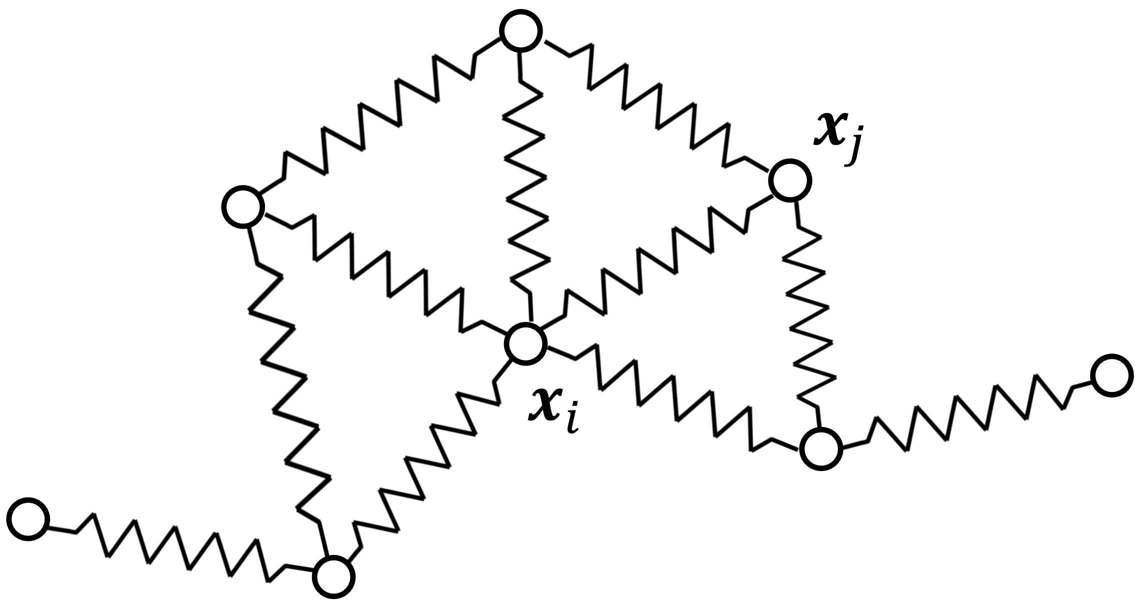
$$0 = F'(x) \approx F'\left(x^{(k)}\right) + F''\left(x^{(k)}\right)\left(x - x^{(k)}\right)$$

$$H(x) = F''(x) = \left[\frac{\partial^2 F}{\partial x_i \partial x_j}\right]$$



- minimum $x^* \Rightarrow H(x^*) > 0$ (SPD)
- maximum $x^* \Rightarrow H(x^*) < 0$ (SND)
- If $F''(x) > 0$ for any x
  $\Rightarrow F(x)$ has no maximum.
  $\Rightarrow F(x)$ has only one minimum.

Hessian Matrix of Mass-Spring System

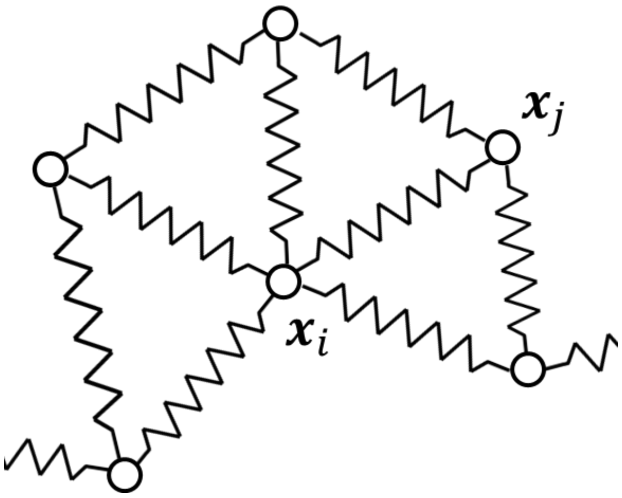


$$f_i = \sum_{j \in N(i)} f_{ij} = \sum_{j \in N(i)} -k\left(\left\|x_i - x_j\right\| - l_0\right)\frac{x_i - x_j}{\left\|x_i - x_j\right\|}$$

Hessian Matrix of Mass-Spring System

Elastic Energy

$$E(x) = \sum_{(i,j)\in\mathcal{E}} \frac{k}{2}\left(\|x_i - x_j\| - l_0\right)^2$$



Force

$$f_i(x) \overset{?}{=} \frac{\partial E}{\partial x_i} = \sum_{j\in N(i)} -k\left(\|x_i - x_j\| - l_0\right)\frac{x_i - x_j}{\|x_i - x_j\|}$$

Hessian

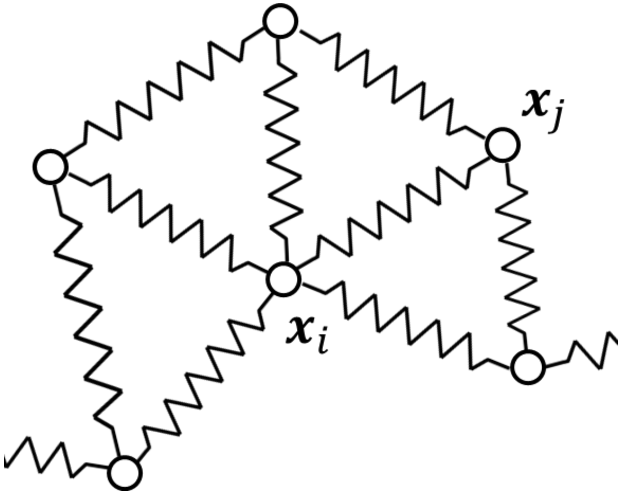
$$H(x) = \left[\frac{\partial E}{\partial x_i \partial x_j}\right]$$

Hessian Matrix of Mass-Spring System

Hessian

$$H(x) = \left[\frac{\partial E}{\partial x_i \partial x_j}\right]$$

$$= \sum_{\boldsymbol{e}=(i,j)\in\mathcal{E}} \begin{bmatrix} \ddots & \vdots & \vdots & \iddots \\ \cdots & \frac{\partial^2 E}{\partial x_i^2} & \frac{\partial^2 E}{\partial x_i \partial x_j} & \cdots \\ \cdots & \frac{\partial^2 E}{\partial x_i \partial x_j} & \frac{\partial^2 E}{\partial x_j^2} & \cdots \\ \iddots & \vdots & \vdots & \ddots \end{bmatrix}$$



$$= \sum_{\boldsymbol{e}=(i,j)\in\mathcal{E}} \begin{bmatrix} \ddots & \vdots & \vdots & \iddots \\ \cdots & H_e & -H_e & \cdots \\ \cdots & -H_e & H_e & \cdots \\ \iddots & \vdots & \vdots & \ddots \end{bmatrix}$$

Hessian Matrix of Mass-Spring System

Hessian

$$H(x) = \left[\frac{\partial E}{\partial x_i \partial x_j}\right] = \sum_{\boldsymbol{e}=(i,j)\in\mathcal{E}} \begin{bmatrix} \ddots & \vdots & \vdots & ⋰ \\ \cdots & H_e & -H_e & \cdots \\ \cdots & -H_e & H_e & \cdots \\ ⋰ & \vdots & \vdots & \ddots \end{bmatrix}$$

$$x_{ij} = x_i - x_j$$

$$\boldsymbol{H}_e = k\frac{\boldsymbol{x}_{ij}\boldsymbol{x}_{ij}^T}{\left\|\boldsymbol{x}_{ij}\right\|^2} + k\left(1 - \frac{l_0}{\left\|\boldsymbol{x}_{ij}\right\|}\right)\left(\boldsymbol{I} - \frac{\boldsymbol{x}_{ij}\boldsymbol{x}_{ij}^T}{\left\|\boldsymbol{x}_{ij}\right\|^2}\right)$$

Hessian Matrix of Mass-Spring System

Hessian

$$H(x) = \left[\frac{\partial E}{\partial x_i \partial x_j}\right] = \sum_{e=(i,j)\in\mathcal{E}} \begin{bmatrix} \ddots & \vdots & \vdots & ⋰ \\ \cdots & H_e & -H_e & \cdots \\ \cdots & -H_e & H_e & \cdots \\ ⋰ & \vdots & \vdots & \ddots \end{bmatrix}$$

$$x_{ij} = x_i - x_j$$

$$\boldsymbol{H}_e = k\frac{\boldsymbol{x}_{ij}\boldsymbol{x}_{ij}^T}{\|\boldsymbol{x}_{ij}\|^2} + k\left(1 - \frac{l_0}{\|\boldsymbol{x}_{ij}\|}\right)\left(\boldsymbol{I} - \frac{\boldsymbol{x}_{ij}\boldsymbol{x}_{ij}^T}{\|\boldsymbol{x}_{ij}\|^2}\right)$$

SPD

SPD

Hessian Matrix of Mass-Spring System

Hessian

$$H(x) = \left[\frac{\partial E}{\partial x_i \partial x_j}\right] = \sum_{e=(i,j)\in\mathcal{E}} \begin{bmatrix} \ddots & \vdots & \vdots & ⋰ \\ \cdots & H_e & -H_e & \cdots \\ \cdots & -H_e & H_e & \cdots \\ ⋰ & \vdots & \vdots & \ddots \end{bmatrix}$$

$$x_{ij} = x_i - x_j$$

$$\boldsymbol{H}_e = k\frac{\boldsymbol{x}_{ij}\boldsymbol{x}_{ij}^T}{\|\boldsymbol{x}_{ij}\|^2} + k\left(1 - \frac{l_0}{\|\boldsymbol{x}_{ij}\|}\right)\left(\boldsymbol{I} - \frac{\boldsymbol{x}_{ij}\boldsymbol{x}_{ij}^T}{\|\boldsymbol{x}_{ij}\|^2}\right)$$

SPD

SPD

Positive Definiteness of Hessian

$x_{ij} = x_i - x_j$

$$\boldsymbol{H}_e = k\frac{\boldsymbol{x}_{ij}\boldsymbol{x}_{ij}^T}{\|\boldsymbol{x}_{ij}\|^2} + k\left(1 - \frac{l_0}{\|\boldsymbol{x}_{ij}\|}\right)\left(\boldsymbol{I} - \frac{\boldsymbol{x}_{ij}\boldsymbol{x}_{ij}^T}{\|\boldsymbol{x}_{ij}\|^2}\right)$$

SPD

positive
when $\|\boldsymbol{x}_{ij}\| > l_0$

SPD

Positive Definiteness of Hessian

$$x_{ij} = x_i - x_j$$

$$\boldsymbol{H}_e = k\frac{\boldsymbol{x}_{ij}\boldsymbol{x}_{ij}^T}{\|\boldsymbol{x}_{ij}\|^2} + k\left(1 - \frac{l_0}{\|\boldsymbol{x}_{ij}\|}\right)\left(\boldsymbol{I} - \frac{\boldsymbol{x}_{ij}\boldsymbol{x}_{ij}^T}{\|\boldsymbol{x}_{ij}\|^2}\right)$$

SPD

negative
when $\|\boldsymbol{x}_{ij}\| < l_0$

SPD

Positive Definiteness of Hessian

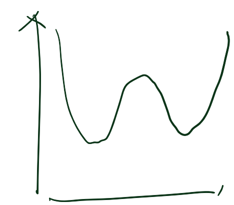
$x_{ij} = x_i - x_j$

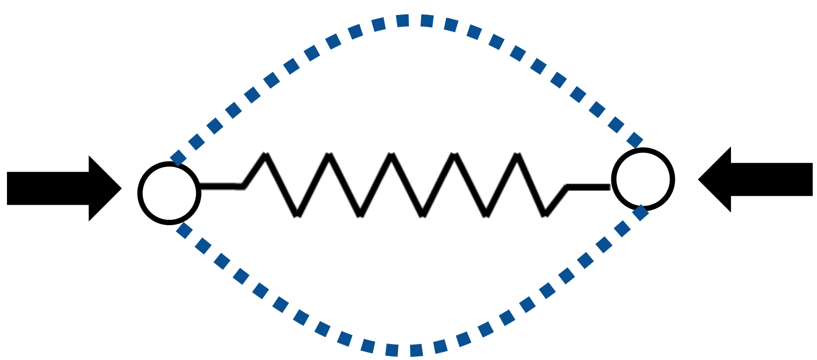
$$\boldsymbol{H}_e = k\frac{\boldsymbol{x}_{ij}\boldsymbol{x}_{ij}^T}{\|\boldsymbol{x}_{ij}\|^2} + k\left(1 - \frac{l_0}{\|\boldsymbol{x}_{ij}\|}\right)\left(\boldsymbol{I} - \frac{\boldsymbol{x}_{ij}\boldsymbol{x}_{ij}^T}{\|\boldsymbol{x}_{ij}\|^2}\right)$$

SPD

negative
when $\|\boldsymbol{x}_{ij}\| < l_0$

SPD





Multiple local minina

Positive Definiteness of Hessian

- Why positive definiteness is important?
  - Newton's method: $F''\left(x^{(k)}\right)\Delta x = -F'\left(x^{(k)}\right)$
  - Some linear solvers can fail when A in $A\Delta x = b$ is not positive definite

$$\boldsymbol{H}_e = k\frac{\boldsymbol{x}_{ij}\boldsymbol{x}_{ij}^T}{\left\|\boldsymbol{x}_{ij}\right\|^2} + k\left(1 - \frac{l_0}{\left\|\boldsymbol{x}_{ij}\right\|}\right)\left(\boldsymbol{I} - \frac{\boldsymbol{x}_{ij}\boldsymbol{x}_{ij}^T}{\left\|\boldsymbol{x}_{ij}\right\|^2}\right)$$

Linear Solvers

- Direct solvers
  - LU, LDLT, Cholesky, …
  - Solve $Ax = b$ in one shot
  - Slow when A is large, inaccurate when $\mathrm{cond}(A)$ is large

dense or sparse

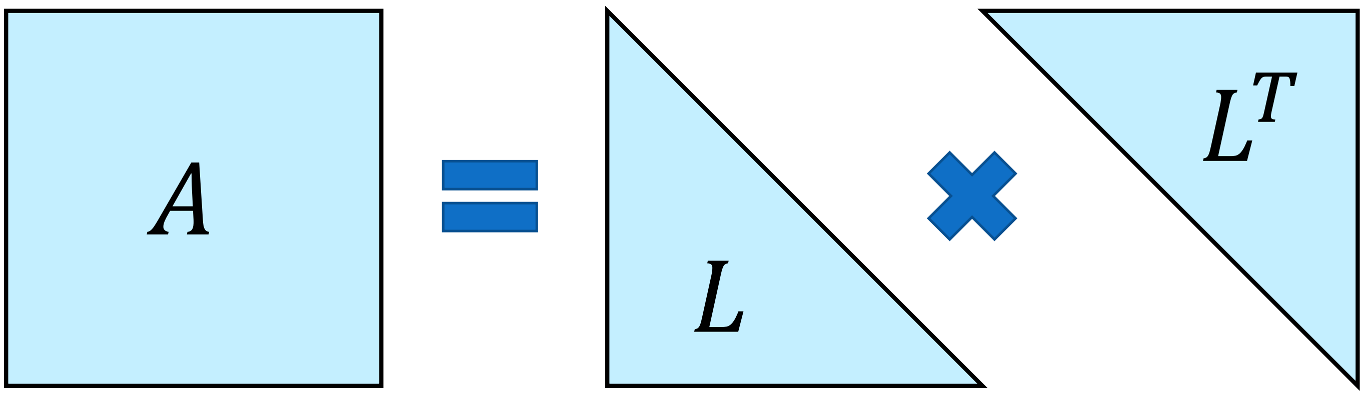
Intel MKL PARDISO

- Iterative solvers
  - Gauss–Seidel method, Jacobi method, …
  - Iteratively approaching the true solution
  - Easy to implement
  - Convergence often rely on positive definiteness of A

Cholesky Decomposition

- A positive definite matrix A can be decomposed as

$$A = LL^T \text{ or } A = LDL^T$$



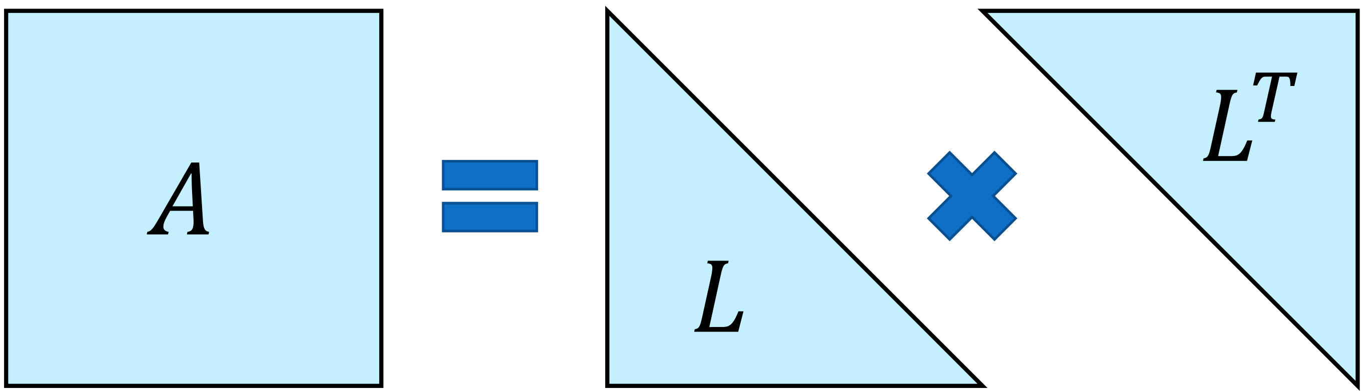
Cholesky Decomposition

半正定分解不唯一

- A positive definite matrix A can be decomposed as

$$A = LL^T \text{ or } A = LDL^T$$

$$L \begin{bmatrix} a_{11} & & & \\ a_{21} & a_{22} & & \\ \vdots & & & \\ a_{n1} & \cdots & & a_{nn} \end{bmatrix} \begin{bmatrix} y_1 \\ \vdots \\ y_2 \end{bmatrix} = \begin{bmatrix} a_{11}y_1 \\ a_{21}y_1 + a_{22}y_2 \\ \vdots \\ \vdots \end{bmatrix}$$



前项代入

$$a_{11}y_1 = b_1 \quad y_1 = \frac{b_1}{a_{11}}$$

$$y_2 = \frac{1}{a_{22}}(b_2 - a_{21}y_1)$$

$$\vdots$$

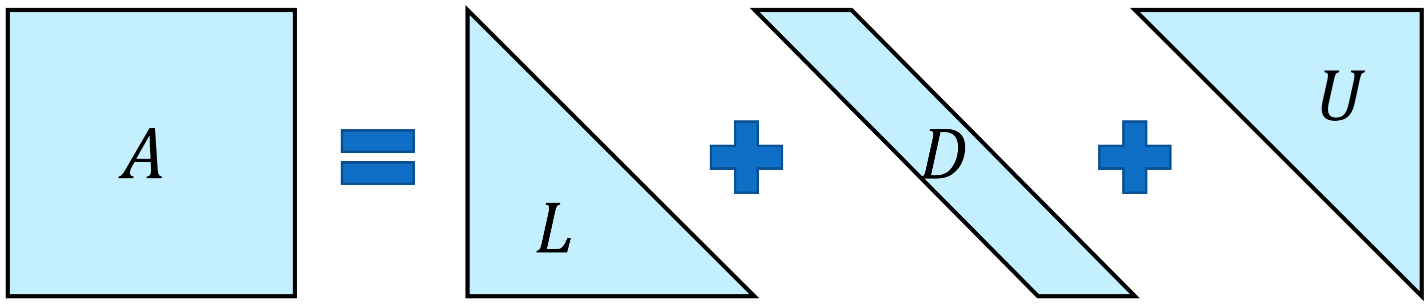
这个算法不用求逆，但难以并行

$$Ax = b \Rightarrow L(L^Tx) = b \Rightarrow \begin{matrix} Ly = b \\ L^Tx = y \end{matrix}$$

Jacobi Method

$$Ax = b$$

- A can be decomposed as



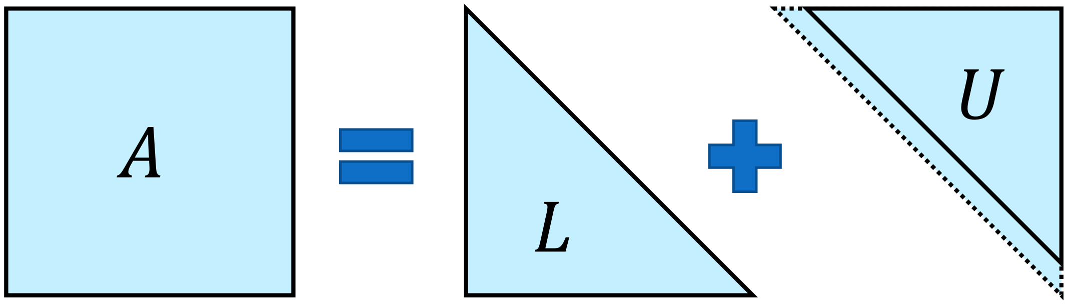
- Then

$$x^{(k+1)} = D^{-1}\left(b - (L + U)x^{(k)}\right)$$

Gauss–Seidel Method

$$Ax = b$$

- A can be decomposed as



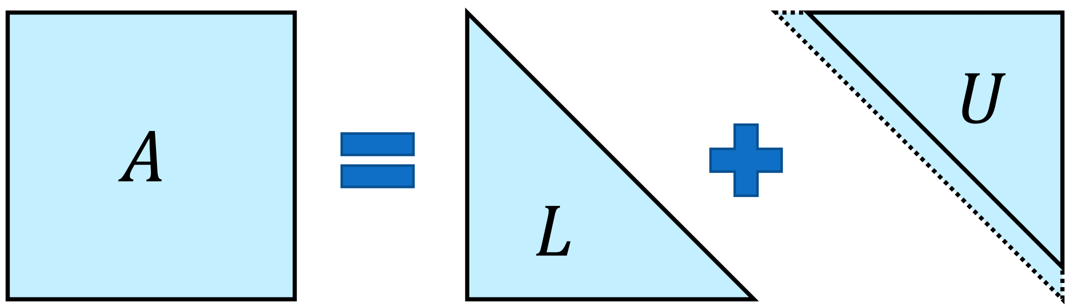
- Then

$$x^{(k+1)} = L^{-1}\left(b - Ux^{(k)}\right)$$

Gauss–Seidel Method

$$Ax = b$$

- A can be decomposed as



- Then

Forward substitution

$$x_i^{(k+1)} = \frac{1}{a_{ii}}\left(b_i - \sum_{j=1}^{i-1} a_{ij}x_j^{(k+1)} - \sum_{j=i+1}^{n} a_{ij}x_j^{(k)}\right)$$

Positive Definiteness of Hessian

- Why positive definiteness is important?
  - Newton's method: $F''\left(x^{(k)}\right)\Delta x = -F'\left(x^{(k)}\right)$
  - Some linear solvers can fail when A in $A\Delta x = b$ is not positive definite
- Drop the last term when $\left\|x_{ij}\right\| < l_0$

$$\boldsymbol{H}_e = k\frac{\boldsymbol{x}_{ij}\boldsymbol{x}_{ij}^T}{\left\|\boldsymbol{x}_{ij}\right\|^2} + k\,\cancel{\left(1 - \frac{l_0}{\left\|\boldsymbol{x}_{ij}\right\|}\right)\left(\boldsymbol{I} - \frac{\boldsymbol{x}_{ij}\boldsymbol{x}_{ij}^T}{\left\|\boldsymbol{x}_{ij}\right\|^2}\right)}$$

- Other solutions, for example:
  - Choi and Ko. 2002. Stable But Responive Cloth. TOG (SIGGRAPH)

Get Rid of Hessian?

- Newton's method $x^{(k+1)} = x^{(k)} + \alpha \Delta$

$$\Delta \leftarrow -\left(H\left(x^{(k)}\right)\right)^{-1} F'\left(x^{(k)}\right)$$

$$H(x) = \mathrm{F}''(x)$$ ☹

Get rid of Hessian?

- Newton's method $x^{(k+1)} = x^{(k)} + \alpha \Delta$

$$\Delta \leftarrow -\left(H\left(x^{(k)}\right)\right)^{-1} F'\left(x^{(k)}\right)$$

$$H(x) = \mathrm{F}''(x) \quad ☹$$

- Quasi-Newton method

$$\Delta \leftarrow -(B_k)^{-1} F'\left(x^{(k)}\right)$$

$$B_k \approx H\left(x^{(k)}\right) \quad ☺$$

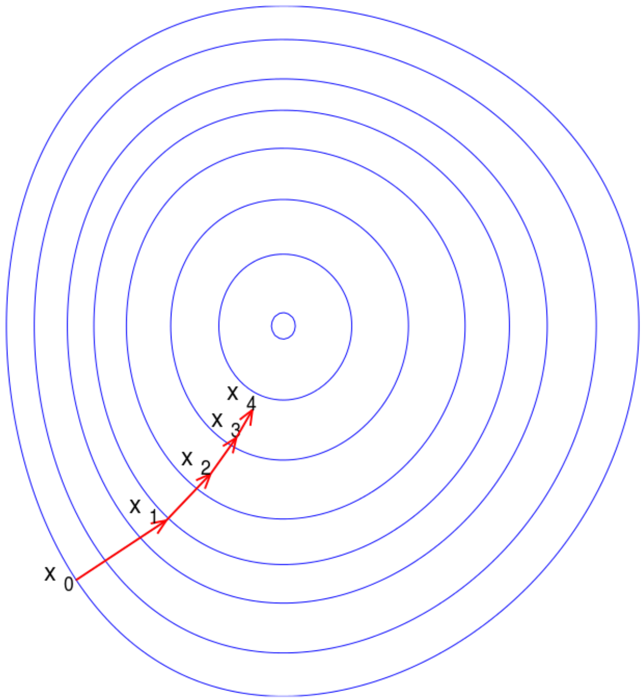
Gradient Descent

$$x^{(k+1)} = x^{(k)} + \alpha \Delta$$

$$\Delta = -(B_k)^{-1} F'\left(x^{(k)}\right)$$

$$B_k = I$$

$$\Delta = -F'\left(x^{(k)}\right)$$

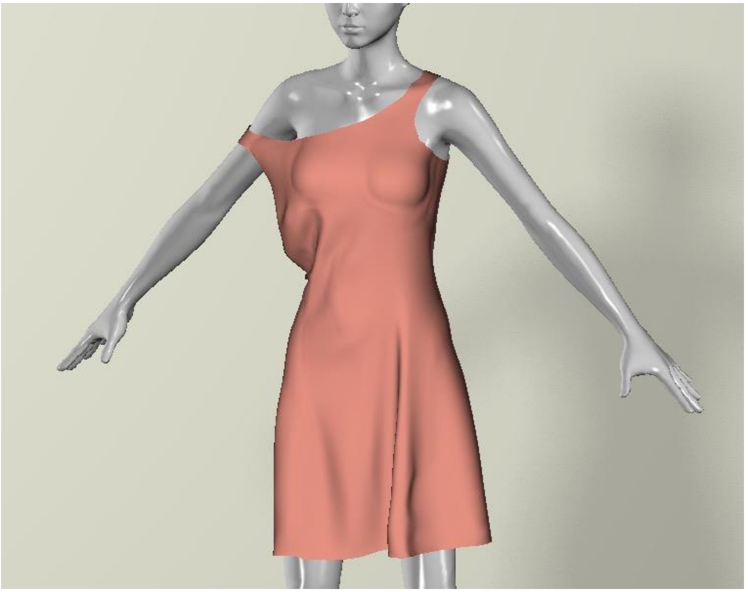


Diagonal Hessian

$$x^{(k+1)} = x^{(k)} + \alpha \Delta$$

$$\Delta = -(B_k)^{-1} F'\left(x^{(k)}\right)$$

$$B_k = \mathrm{diag}\left(\frac{\partial^2 F}{\partial x_i \partial x_i}\right)$$



Huamin Wang. 2015. ***A chebyshev semi-iterative approach for accelerating projective and position-based dynamics****. ACM Trans. Graph. 34, 6, Article 246 (November 2015)*

BFGS Algorithm

- Broyden-Fletcher-Goldfarb-Shanno Algorithm

BFGS Algorithm

$$x^{(k+1)} = x^{(k)} + \mathit{\Delta}$$

- Newton's method

$$H\left(\boldsymbol{x}^{(\boldsymbol{k})}\right)\Delta = -F'(\boldsymbol{x}^{(\boldsymbol{k})})$$

- Quasi-Newton method

$$B_k\Delta = -F'(\boldsymbol{x}^{(\boldsymbol{k})})$$

- BFGS Algorithm

$$\mathit{\Delta}_k \leftarrow B_k\mathit{\Delta} = -F'\left(x^{(k)}\right)$$
$$y_k \leftarrow F'\left(x^{(k+1)}\right) - F'\left(x^{(k)}\right)$$
$$B_{k+1} \leftarrow B_k + \frac{y_k y_k^T}{y_{\mathrm{k}}^{\mathrm{T}}\Delta_k} - \frac{B_k\Delta_k\Delta_k^T B_k^T}{{\Delta_k}^T B_k\Delta_k}$$

BFGS Algorithm

$$x^{(k+1)} = x^{(k)} + \mathit{\Delta}$$

- Newton's method

$$H\left(\boldsymbol{x}^{(\boldsymbol{k})}\right)\Delta = -F'(\boldsymbol{x}^{(\boldsymbol{k})})$$

- Quasi-Newton method

$$B_k\Delta = -F'(\boldsymbol{x}^{(\boldsymbol{k})})$$

- BFGS Algorithm

$$\mathit{\Delta}_k \leftarrow -B_k^{-1}F'\left(x^{(k)}\right)$$
$$y_k \leftarrow F'\left(x^{(k+1)}\right) - F'\left(x^{(k)}\right)$$
$$B_{k+1}^{-1} \leftarrow \left(I - \frac{\Delta_k y_k^T}{y_{\mathrm{k}}^{\mathrm{T}}\Delta_k}\right)B_k^{-1}\left(I - \frac{y_k\Delta_k^T}{y_{\mathrm{k}}^{\mathrm{T}}\Delta_k}\right) + \frac{\Delta_k\Delta_k^T}{y_{\mathrm{k}}^{\mathrm{T}}\Delta_k}$$

L-BFGS Algorithm

- Limited-memory BFGS

$$B_{k+1}^{-1} \leftarrow \left(I - \frac{\Delta_k y_k^T}{y_\mathrm{k}^\mathrm{T}\Delta_k}\right) B_k^{-1} \left(I - \frac{y_k \Delta_k^T}{y_\mathrm{k}^\mathrm{T}\Delta_k}\right) + \frac{\Delta_k \Delta_k^T}{y_\mathrm{k}^\mathrm{T}\Delta_k}$$

  - $B \in \mathbb{R}^{n\times n}$ can be very big…
  - $\Delta_k, y_k \in \mathbb{R}^n$ is small

L-BFGS Algorithm

- Limited-memory BFGS

$$B_{k+1}^{-1} \leftarrow \left(I - \frac{\Delta_k y_k^T}{y_{\mathrm{k}}^{\mathrm{T}} \Delta_k}\right) B_k^{-1} \left(I - \frac{y_k \Delta_k^T}{y_{\mathrm{k}}^{\mathrm{T}} \Delta_k}\right) + \frac{\Delta_k \Delta_k^T}{y_{\mathrm{k}}^{\mathrm{T}} \Delta_k}$$

  - $B \in \mathbb{R}^{n \times n}$ can be very big…
  - $\Delta_k, y_k \in \mathbb{R}^n$ is small

- A possible solution
  - Record $\rho_k = \left(y_{\mathrm{k}}^{\mathrm{T}} \Delta_k\right)^{-1}, \Delta_k, y_k$
  - Then unroll up to m $(\ll n)$ steps of the above equation
  - Many different implementations

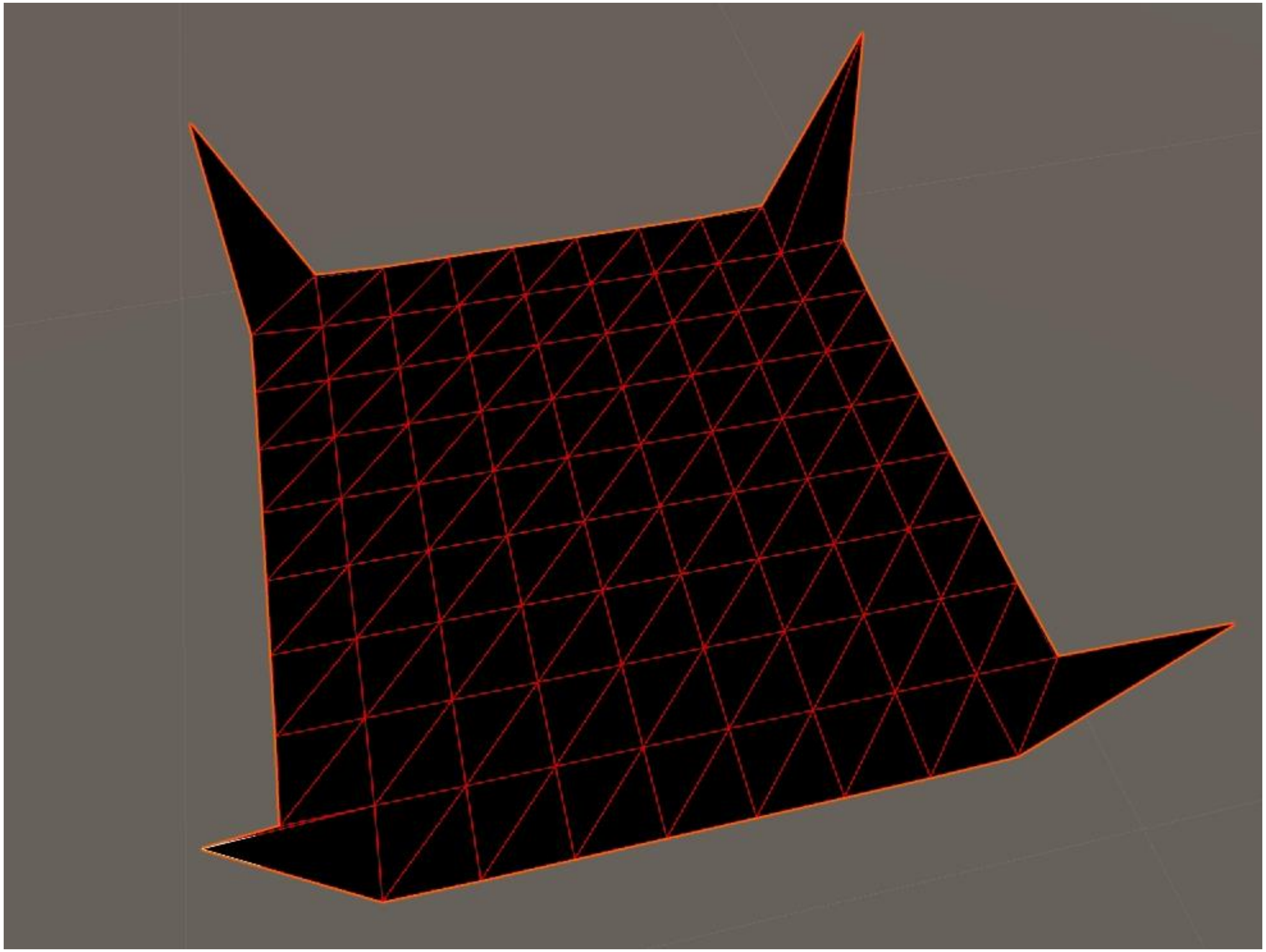
Outline

- Basic concepts of simulation
  - Types of simulation
  - Discretization in time
  - Forward/Backward/Symplectic Euler integration
- Particle System
- Mass-Spring System
- Numerical Methods
  - Newton’s method
  - Linear solvers
  - Quasi-Newton methods

Constraints and Contacts

Some simple cases, we will discuss this topic again later

System with Constraints



Inverse Mass in System Dynamics

$$v_{n+1} = v_n + hM^{-1}f$$

$$v_{n+1} = v_n + h\begin{bmatrix} 1/m & & \\ & 1/m & \\ & & 1/m \end{bmatrix} f$$

Modified Inverse Mass

$$v_{n+1} = v_n + h\textcolor{red}{M^{-1}}f$$

$$v_{n+1} = v_n + h\begin{bmatrix} 1/m & & \\ & 1/m & \\ & & 0 \end{bmatrix} f$$

无穷大质量质点

What does this mean?

Modified Inverse Mass

$$v_{n+1} = v_n + hM^{-1}f$$

$$v_{n+1} = v_n + h\frac{1}{m}(I - \underbrace{pp^T}_{n\times n})f$$

约束不在p的方向上移动

A more general case:
no change is allowed in direction p

Modified Inverse Mass

$$v_{n+1} = v_n + hM^{-1}f$$

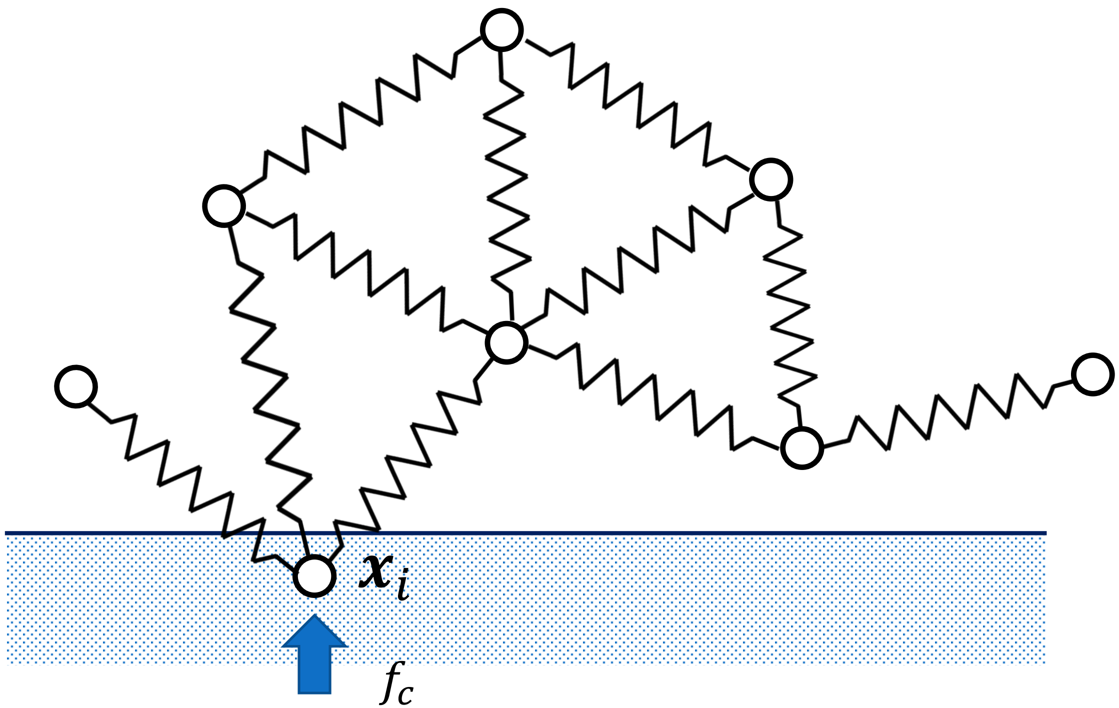
$$v_{n+1} = v_n + h\frac{1}{m}\left(I - pp^T - qq^T\right)f$$

A more general case:
no change is allowed in directions p and q

Contacts

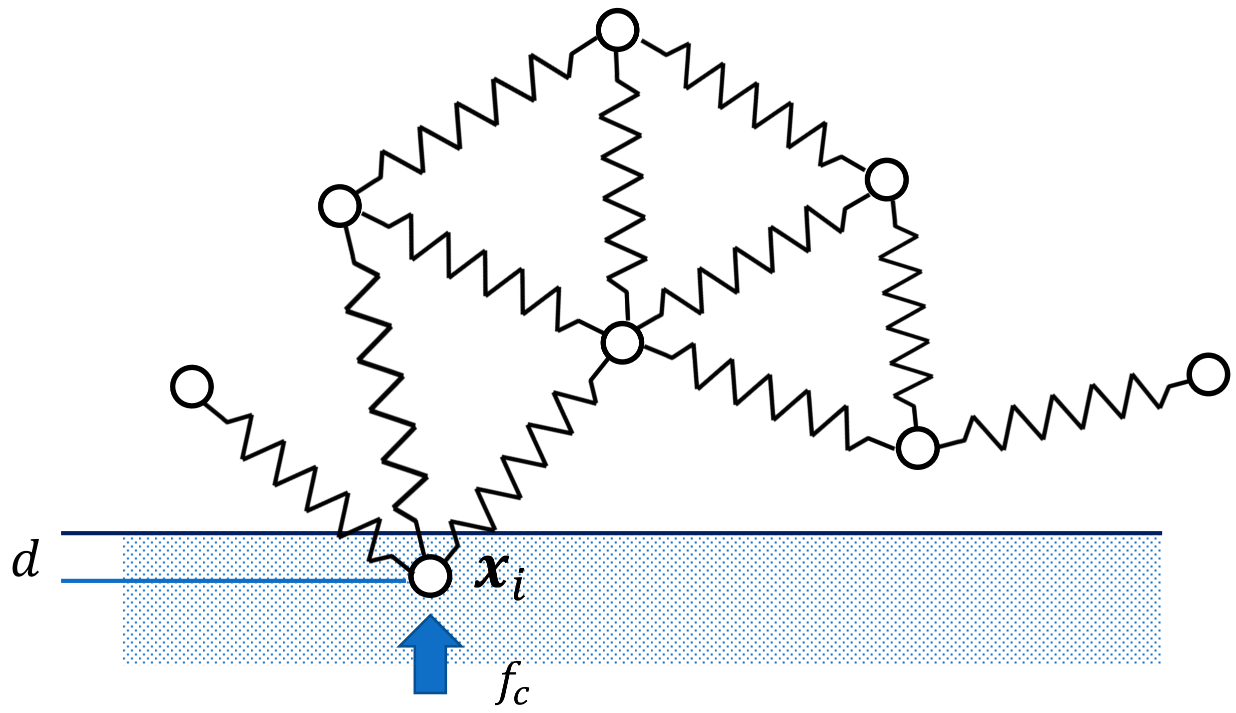
- Contact handling is a BIG topic in simulation
  - Collision detection
    - Discrete Collision Detection (DCD)
    - Continuous Collision Detection (CCD)
  - Contact models
    - Penalty-based methods
    - Constraint-based methods

Penalty-based Methods



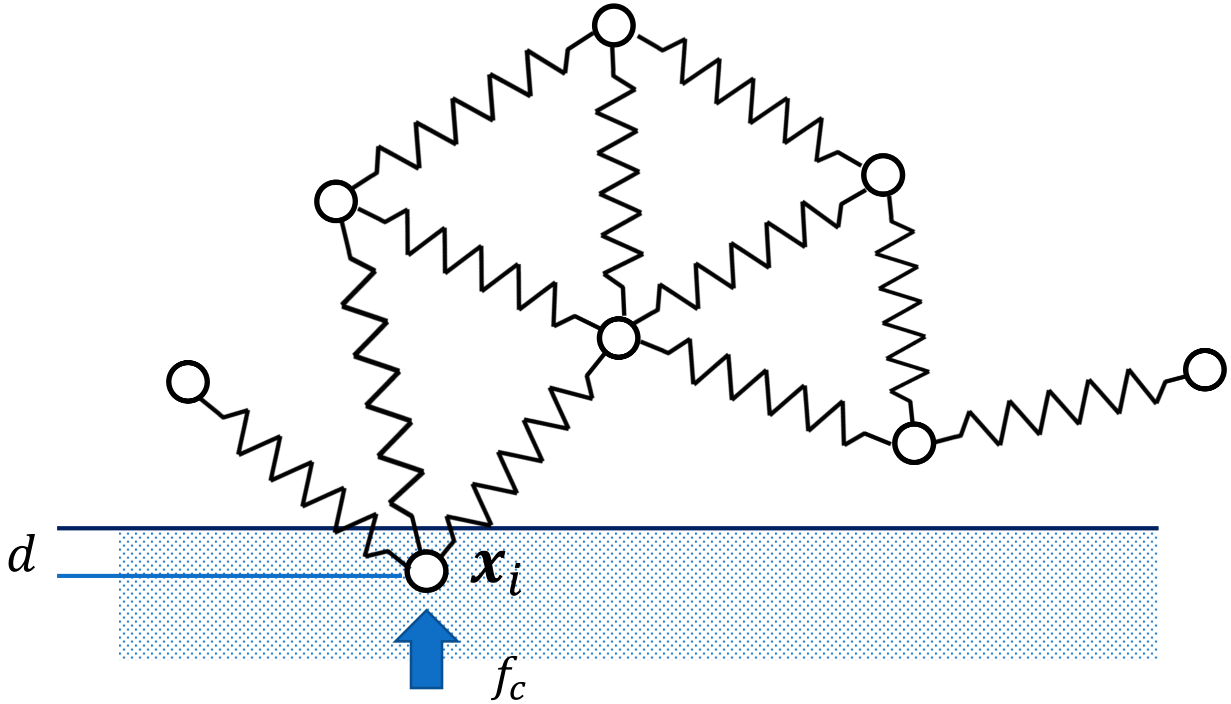
$$f_i = \sum_{j \in N(i)} f_{ij} + f_c$$

Penalty-based Methods



$$f_c = -k_p d$$

Penalty-based Methods



$$f_c = -k_p d - k_d v_i$$ damp. 能量损耗

Any Questions?