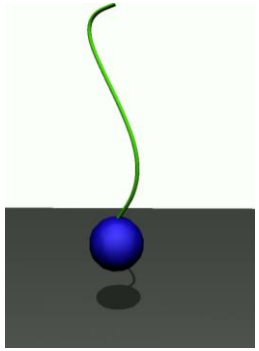
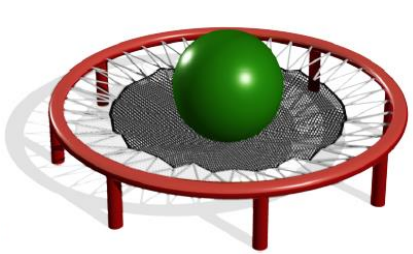
Time stepping scheme for interactive stiff simulation with changes in topology

$$\mathbb{M}(\mathbf{q})\dot{\mathbf{v}} = \mathbb{P}(t) - \mathbb{F}\left(\mathbf{q}, \mathbf{v}\right) + \mathbf{H}^T\lambda$$

InSimo

WE CARE FOR SIMULATION





Background: stiff systems with hard/soft constraints

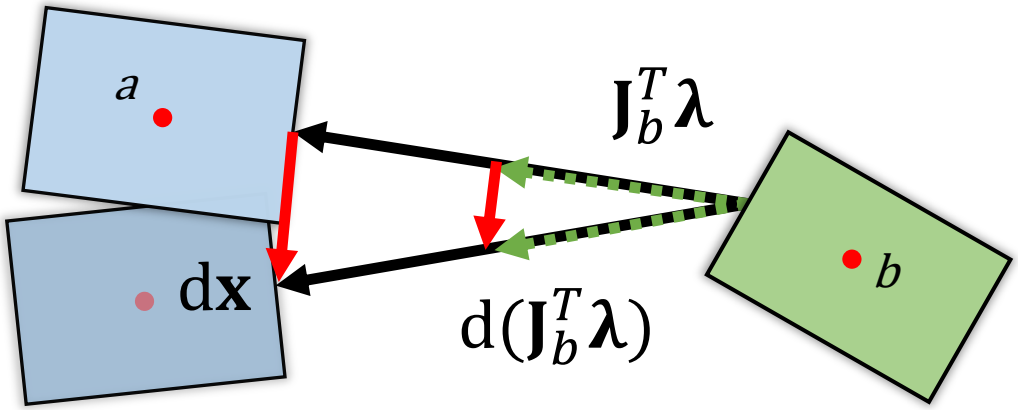
[Stable constraint dynamics (Tournier, Nesme, Gilles, Faure)]

PR#459 : [https://github.com/sofa-framework/sofa/pull/459]

InSimo
WE CARE FOR SIMULATION

Geometric stiffness

- Stiffness which arises from the non linearity of constraint equations
- Tensor which model change in constraint direction



$$\tilde{\mathbf{K}} = \frac{\partial \mathbf{J}^T}{\partial \mathbf{x}}\,\boldsymbol{\lambda}$$

InSimo
WE CARE FOR SIMULATION

Current implementation in FreeMotionAnimationLoop

$ApplyDJT(\lambda)$ → $\widetilde{K}$

$OdeSolve: q, v \rightarrow q_{free}, v_{free}$ → $v_{free} = A^{-1}b + v$ $q_{free} = q + hv_{free}$

$Project(v_{free})$

$ApplyJ(v_{free})$

$BuildConstraintSystem$ → J^T $JA^{-1}J^T$

$ConstraintSolve: q_{free}, v_{free} \rightarrow q^+, v^+$ → $dv = A^{-1}J^T\lambda$

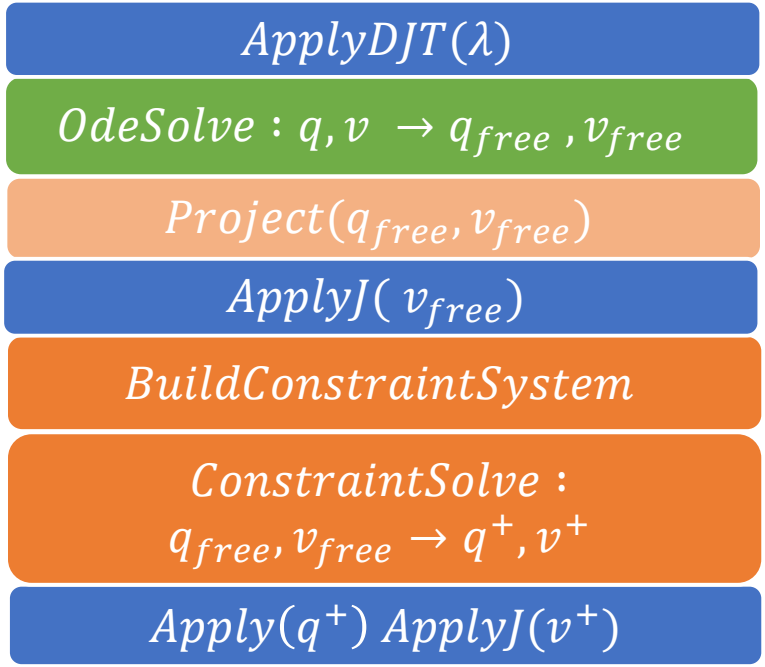
$Project(q^+, v^+)$

$Apply(q^+)\ ApplyJ(v^+)$

InSimo
WE CARE FOR SIMULATION

Modified FreeMotionAnimationLoop (at InSimo)

Apply projective “impulses” only on the free motion state vectors to remain compatible with the constraint formulation.
(For example to clamp the velocity of objects to a fixed upper bound)



InSimo
WE CARE FOR SIMULATION

Introducing changes in the topology

InSimo
WE CARE FOR SIMULATION

Possible implementation : change the topology between time steps

- Modify the topology, go through topology mappings
- Interpolate kinematic quantities (rest position, current position, velocity) : not necessarily “valid” for mapped dofs unless the mapping is linear
- Go through mechanical mappings to account for non linear relationship between independent dofs and mapped dofs generalized position and velocity
- Start a new time step

$$TopologyChange(q, v)$$

$$Apply(q_{rest})\ Apply(q)\ ApplyJ(v)$$

InSimo
WE CARE FOR SIMULATION

Implementation at InSimo : Change the topology during the time stepping scheme

$ApplyDJT(\lambda)$

$OdeSolve : q, v \rightarrow q_{free}, v_{free}$

$Project\ (q_{free}, v_{free})$

$ApplyJ(v_{free})$

$BuildConstraintSystem$

$ConstraintSolve :$
$q_{free}, v_{free} \rightarrow q^+, v^+$

$TopologyChange\ (q^+, v^+)$

$Apply(q_{rest})$
$Apply(q^+)\ ApplyJ(v^+)$

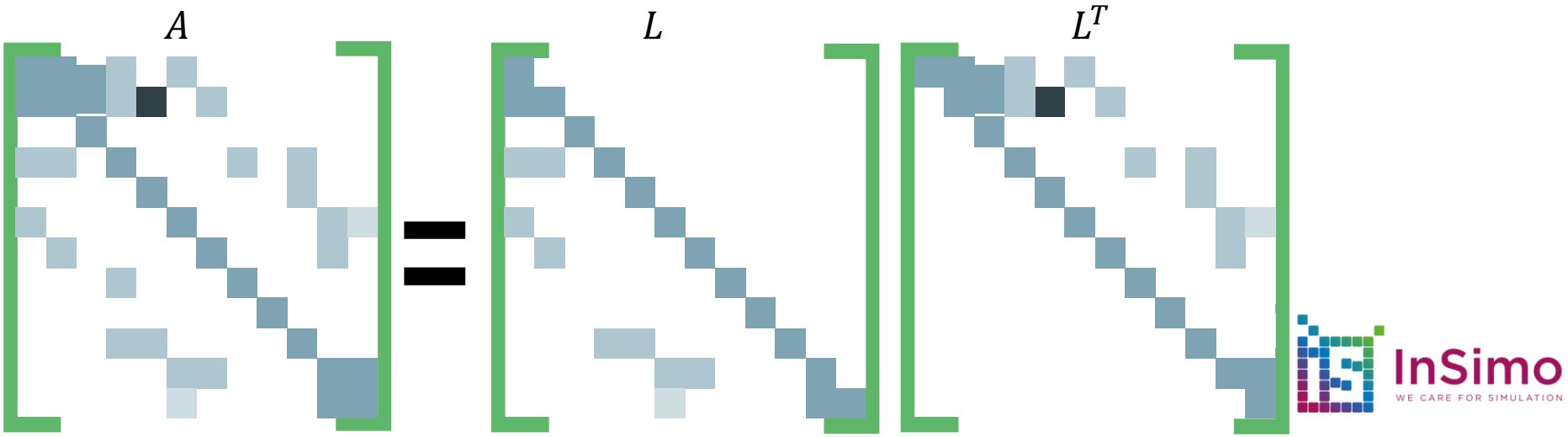
Reduce the number of mapping traversals, which can be expensive for non linear mappings
(e.g. computing a continuous normal field and its derivative of an object boundary)

InSimo
WE CARE FOR SIMULATION

Direct solver for bilateral constraints

Idea: augment existing Cholesky factorization

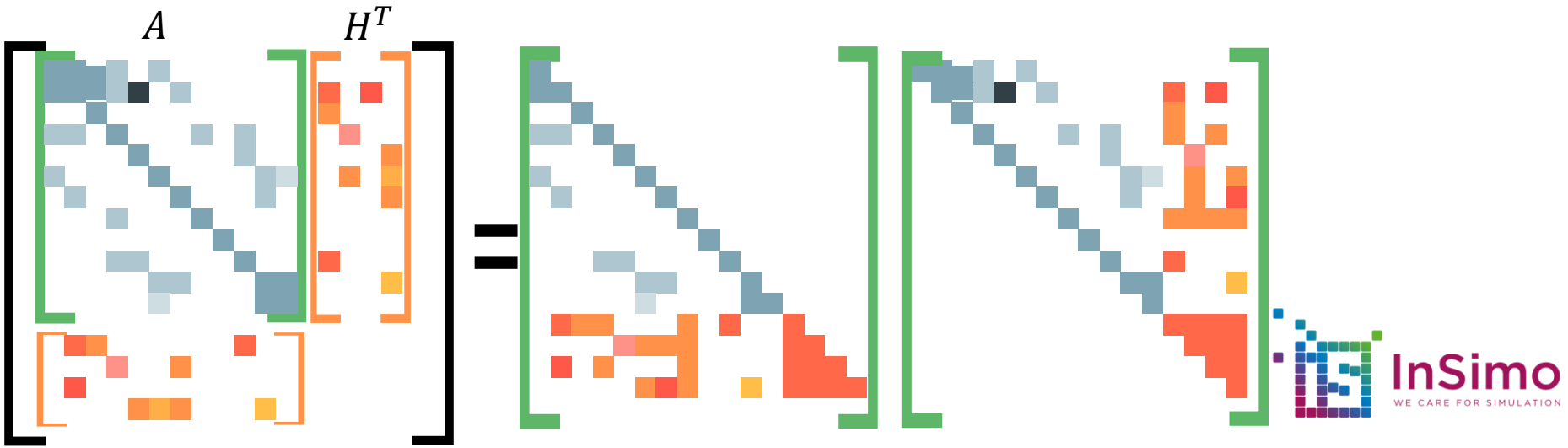
- Robust treatment of bilateral constraints not really possible without using a (prohibitively) large number of iterations in Projected Gauss Seidel



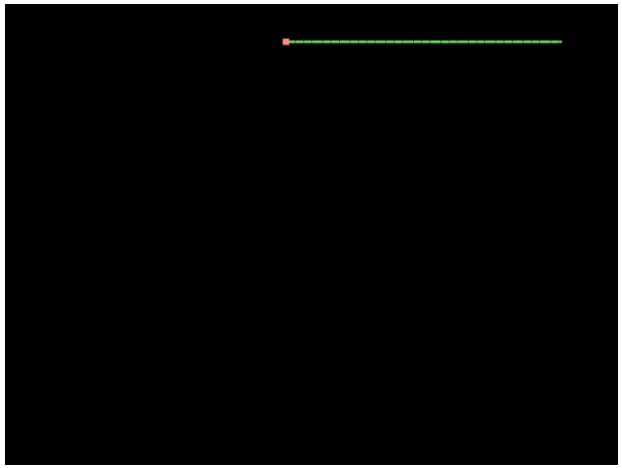
InSimo
WE CARE FOR SIMULATION

Idea: augment existing Cholesky factorization

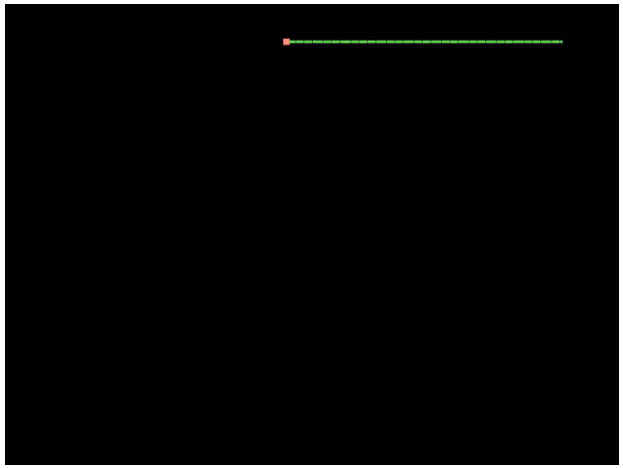
- Robust treatment of bilateral constraints not really possible without using a (prohibitively) large number of iterations in Projected Gauss Seidel



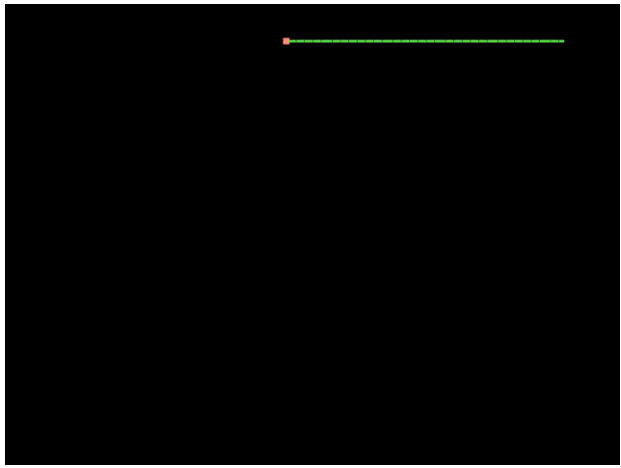
GS 100 iterations



GS 200 iterations



Bilateral factorized



20 cm rod | mass 1,5g | 100 dofs | timestep 0,01 ms

InSimo
WE CARE FOR SIMULATION

Questions ?


InSimo
WE CARE FOR SIMULATION

Soft constraints formulation, geometric stiffness

- Formulation where the unknown is the velocity increment

$$\begin{bmatrix} M - h^2(K + \widetilde{K}) & J^\top \\ J & \frac{1}{h^2}C \end{bmatrix} \begin{bmatrix} dv \\ -\mu^+ \end{bmatrix} = \begin{bmatrix} hf - h^2(K + \widetilde{K})v \\ \frac{-\phi}{h} - Jv \end{bmatrix}$$

- Formulation where the unknown is the new velocity

$$\begin{bmatrix} M - h^2(K + \widetilde{K}) & J^\top \\ J & \frac{1}{h^2}C \end{bmatrix} \begin{bmatrix} v^+ \\ -\mu^+ \end{bmatrix} = \begin{bmatrix} Mv + hf \\ \frac{-\phi}{h} \end{bmatrix}$$

InSimo
WE CARE FOR SIMULATION

Soft constraints formulation, geometric stiffness

- Compute geometric stiffness using previous constraint forces
- Solve to find unconstrained velocities (free motion)

$$\begin{cases} A = M - h^2(K + \tilde{K}) \\ b = hf - h^2(K + \tilde{K})v \\ v_{free} = A^{-1}b + v \end{cases}$$

- Assemble and solve reduced system to find constraint forces

$$(JA^{-1}J^{\top} + \frac{1}{h^2}C)\mu^{+} = \frac{-1}{h}\phi - J(v + v_{free})$$

- Inject back constraint forces to find corrective motion

$$v^{+} = v_{free} + A^{-1}\mu^{+}$$

InSimo
WE CARE FOR SIMULATION

Soft constraint : spring damper

$$\mathbf{C}\lambda^{+} - hJv^{+} - \mathbf{k}\phi = 0$$

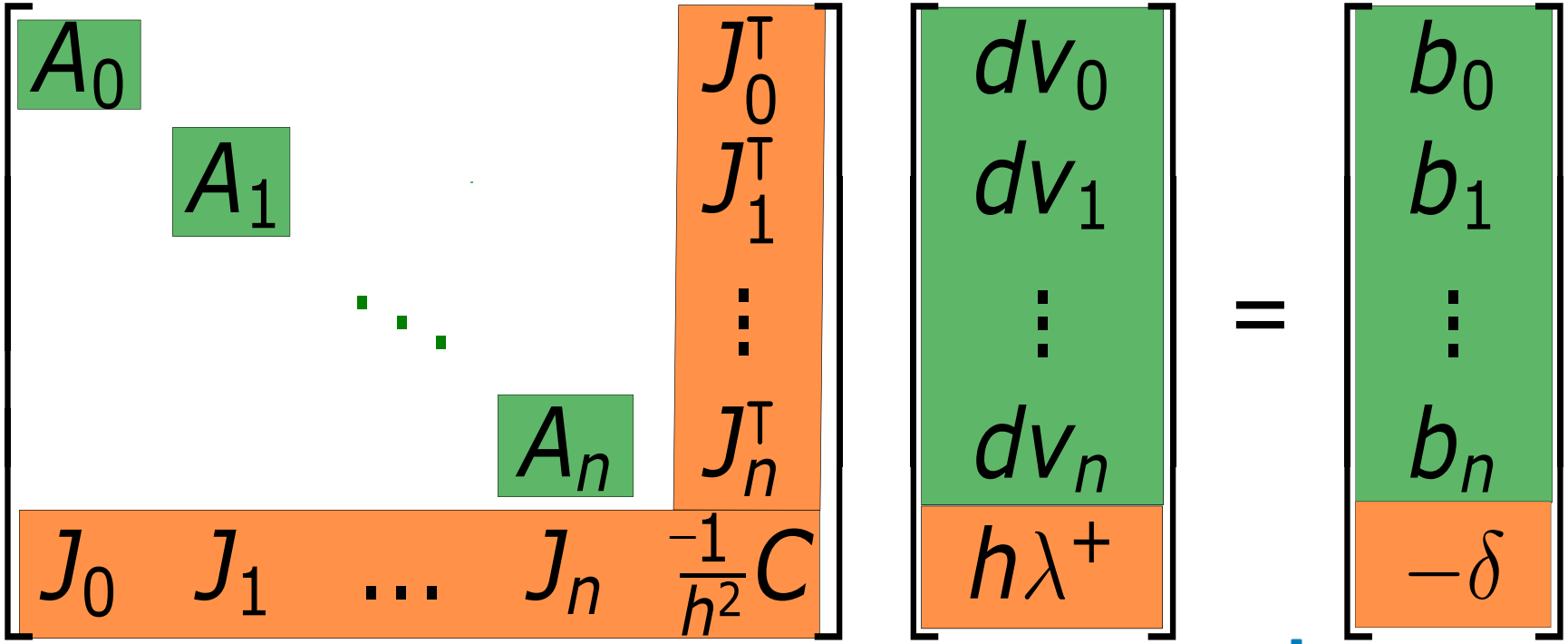
$\boldsymbol{C}$: damping factor
- softW
- CFM (Bullet, ODE)

$\boldsymbol{k}$: position error factor
- Erp (ODE, Bullet)
- Baumgarte stabilisation

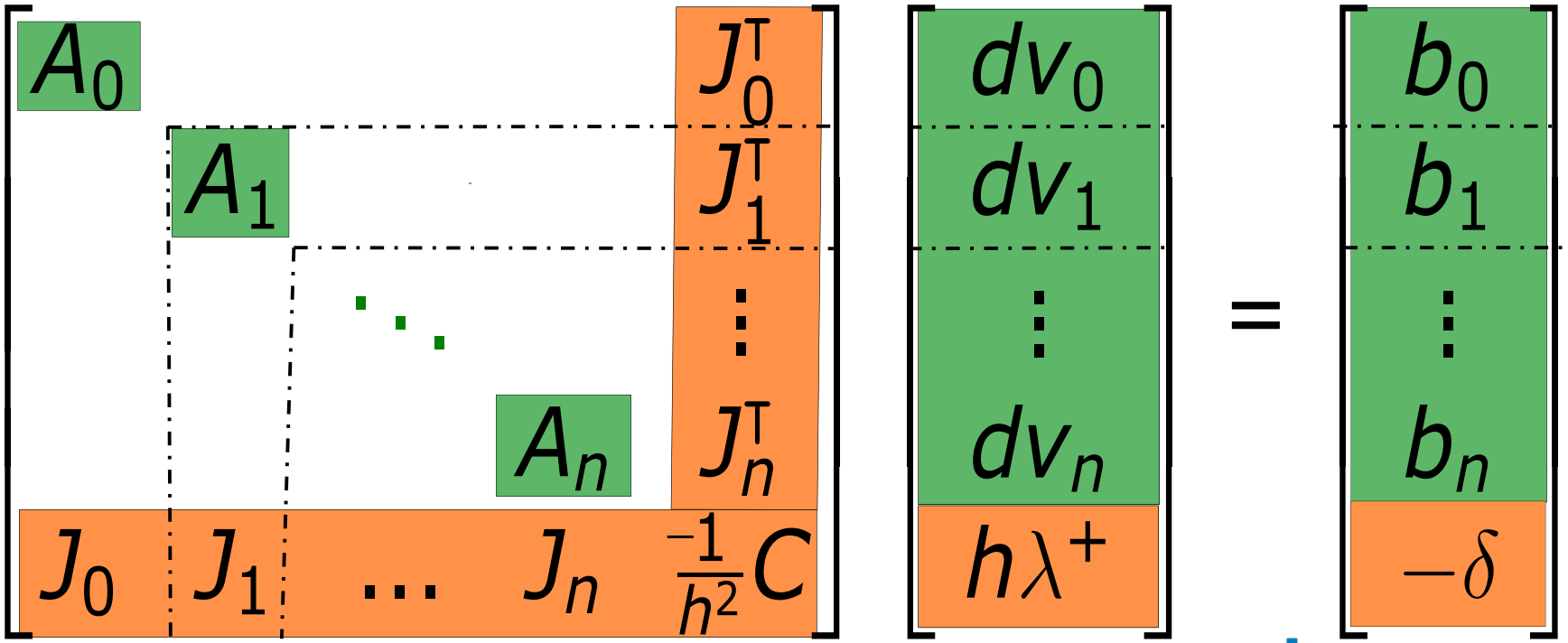
$$\ddot{x} + 2\zeta\omega_0\dot{x} + \omega_0^2 x = 0$$

Erin Catto (Box2D) Soft Constraints GDC 2011

InSimo
WE CARE FOR SIMULATION

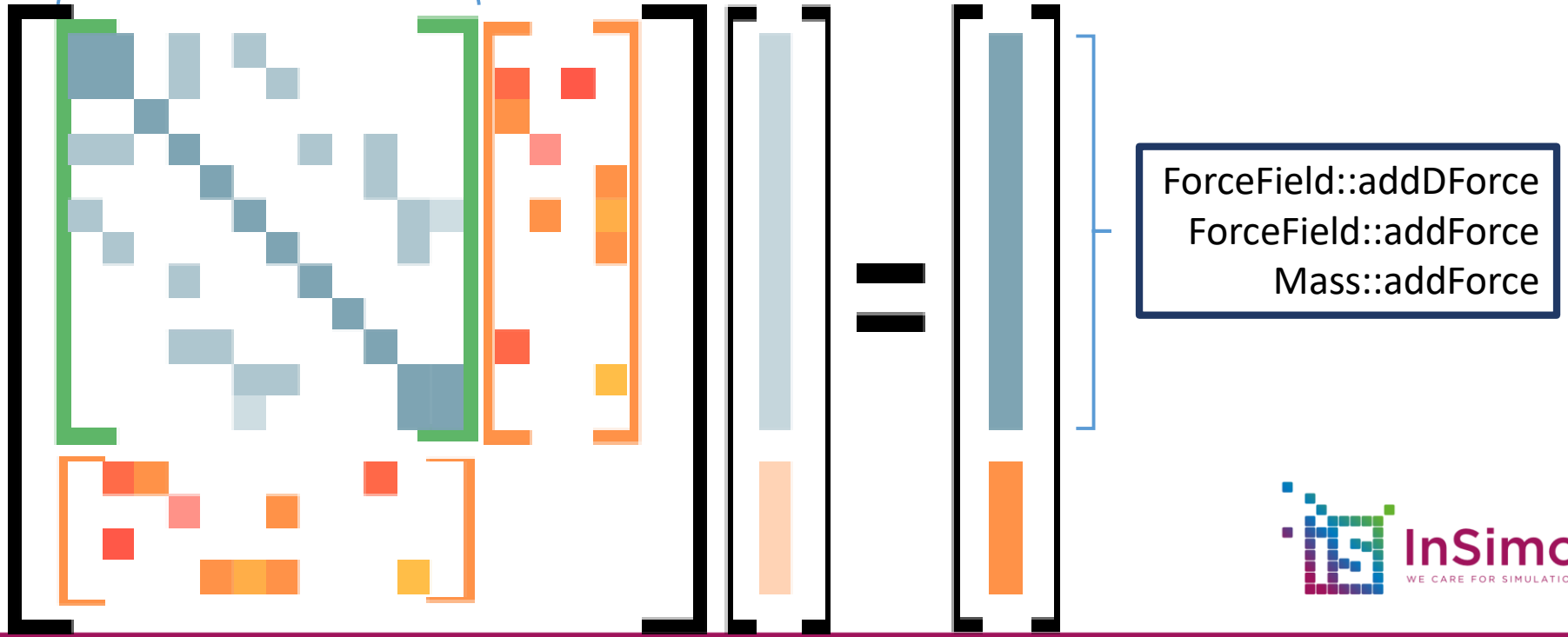


InSimo
WE CARE FOR SIMULATION



InSimo
WE CARE FOR SIMULATION

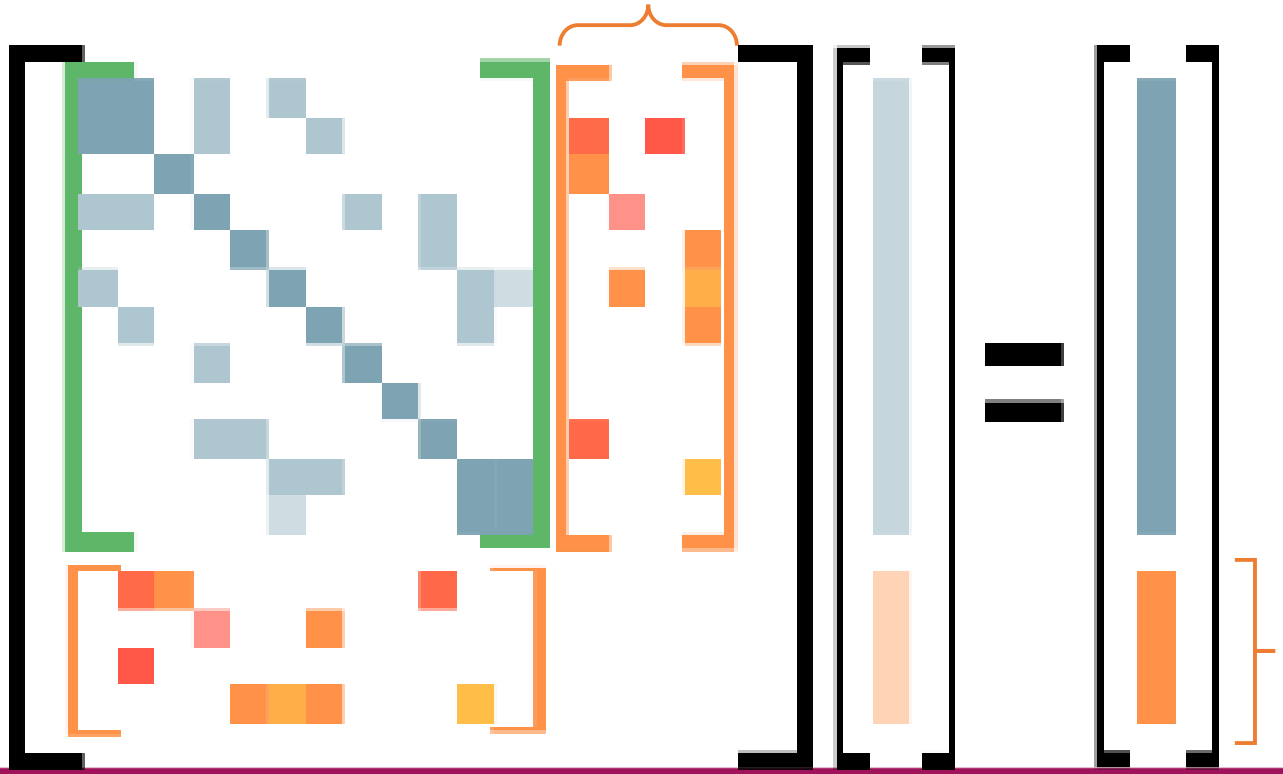
ForceField::addKtoMatrix
Mass::addMToMatrix



ForceField::addDForce
ForceField::addForce
Mass::addForce

InSimo
WE CARE FOR SIMULATION

Constraint::buildConstraintMatrix
Mapping::applyJT

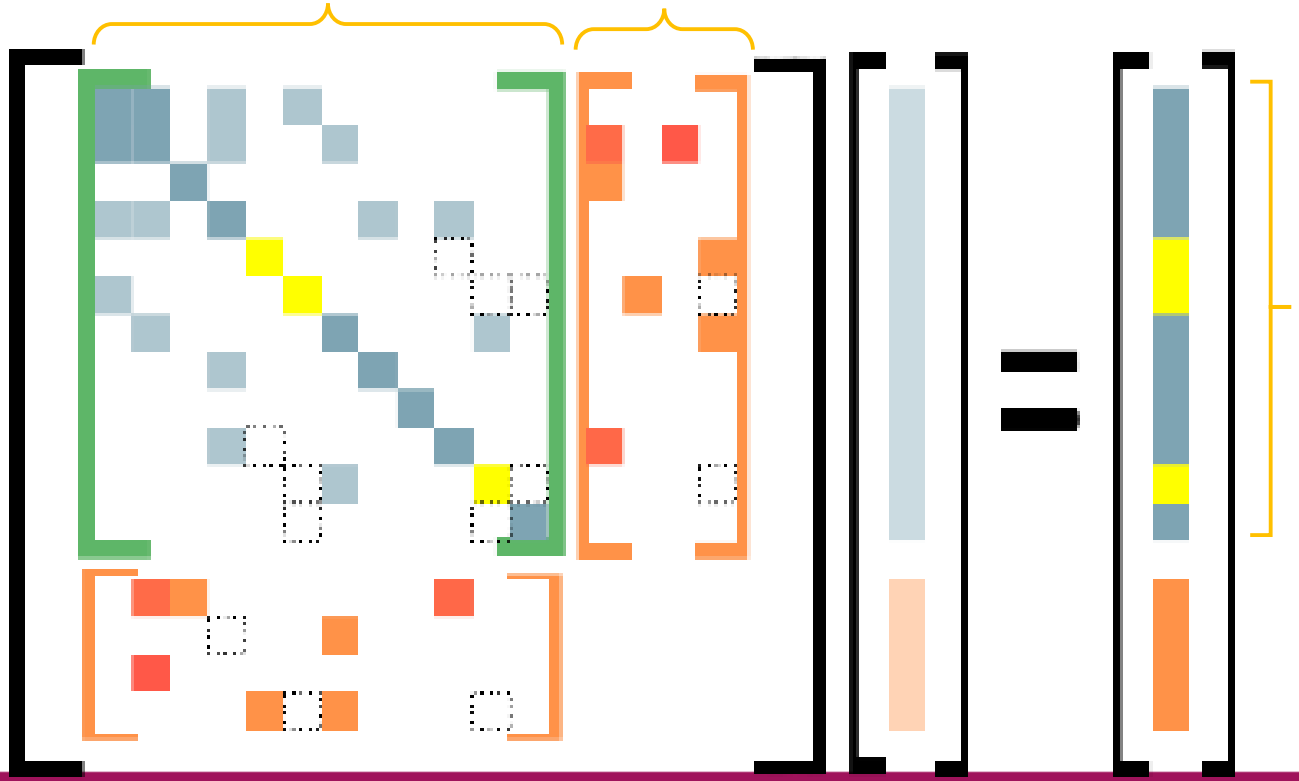


Constraint::
getConstraintViolation

InSimo
WE CARE FOR SIMULATION

ProjectiveConstraint::
projectMatrix

ProjectiveConstraint::
projectJacobianMatrix



ProjectiveConstraint::
projectResponse

InSimo
WE CARE FOR SIMULATION

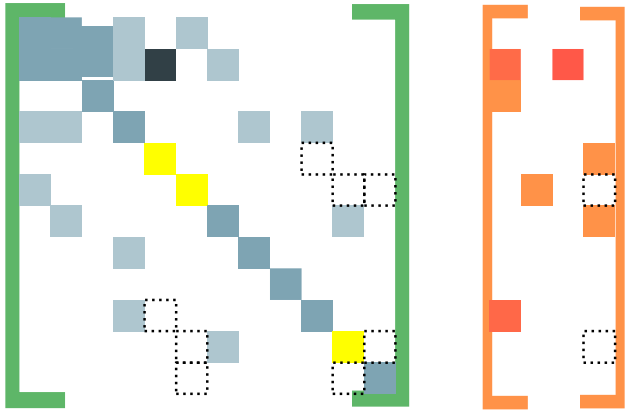
ConstraintCorrection::addJMinvJt

$$W = \left[\;\right]\left[\;\right]^{-1}\left[\;\right]$$

InSimo
WE CARE FOR SIMULATION

ConstraintCorrection::
computeMotionCorrectionFromLambda

$$dv = \left[\ \right]^{-1} h\lambda$$



InSimo
WE CARE FOR SIMULATION