Differential Constraints

A.C.Jalba@tue.nl

Agenda

- Differential Constraints
- Project 1 – overview
- Presentation schedule

Symplectic integration

- Consider the system below with $\mathbf{x} = [x\, y]^T$

$$\dot{\mathbf{x}} = \begin{bmatrix} 0 & -1 \\ 1 & 0 \end{bmatrix} \mathbf{x}$$

- What would happen if we solve for x explicitly and for y implicitly ?
- Would this work ? Why ?

- Solve implicitly:

$$x_{i+1} = x_i - h\, y_i$$
$$y_{i+1} = y_i + h\, x_{i+1}$$

Long-term Evolution

$$x_{i+1} = x_i - h\, y_i$$
$$y_{i+1} = y_i + h\, x_{i+1}$$

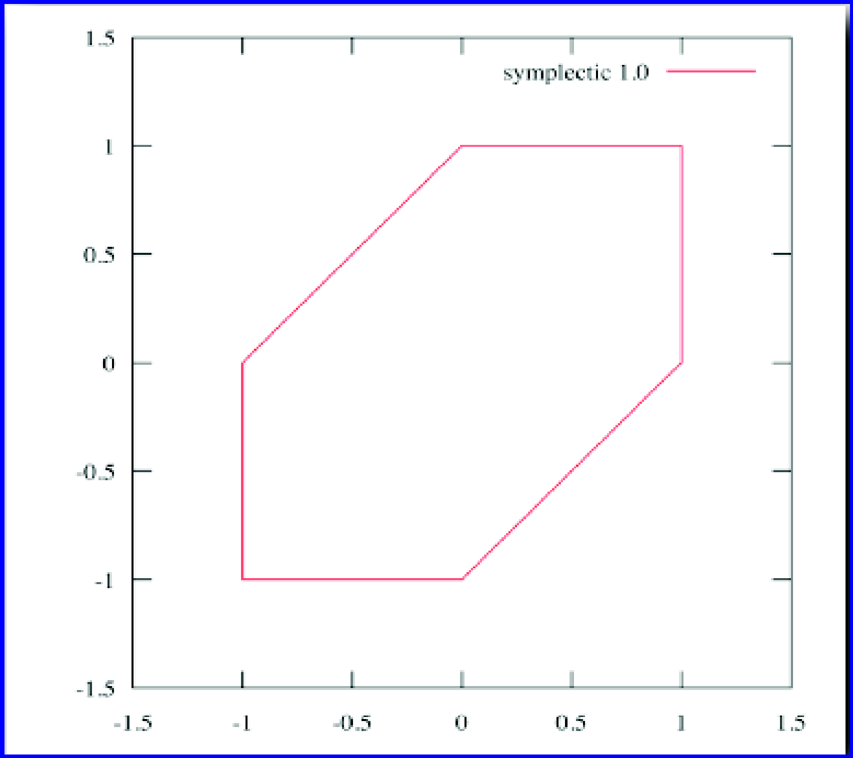
(h = 1.0)

```
x =  1.0;
y =  0.0;
while(true):
  print x, y;
  x -= y;
  y += x;
```

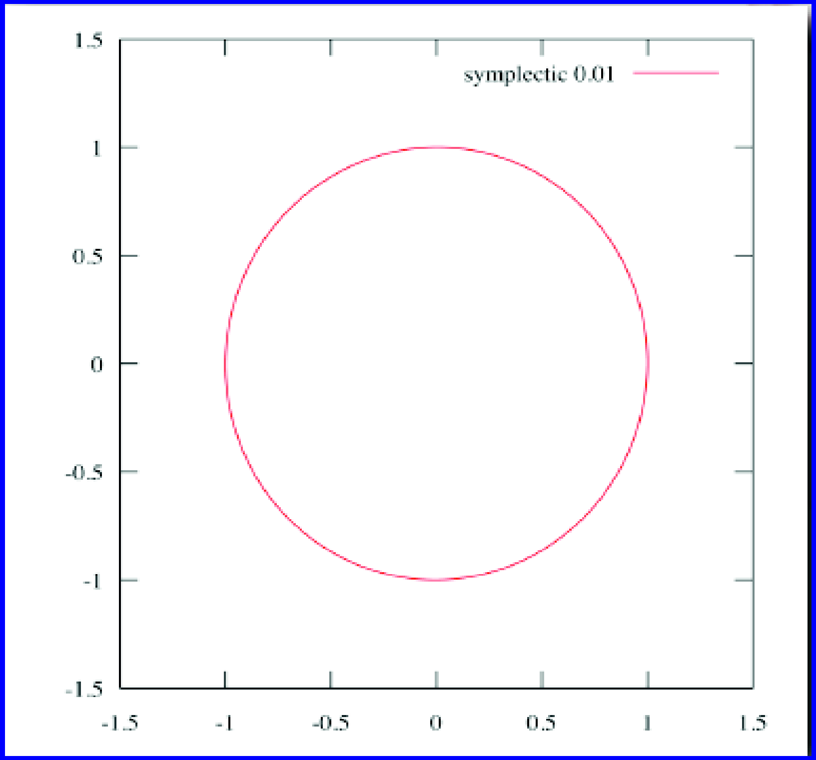
```
 1.0  0.0
 1.0  1.0
 0.0  1.0
-1.0  0.0
-1.0 -1.0
 0.0 -1.0
 1.0  0.0
 1.0  1.0
 0.0  1.0
-1.0  0.0
-1.0 -1.0
 0.0 -1.0
 1.0  0.0
```

- Trajectory in phase space should be a circle around origin.
- Radius is given by initial condition.

Long-term Evolution



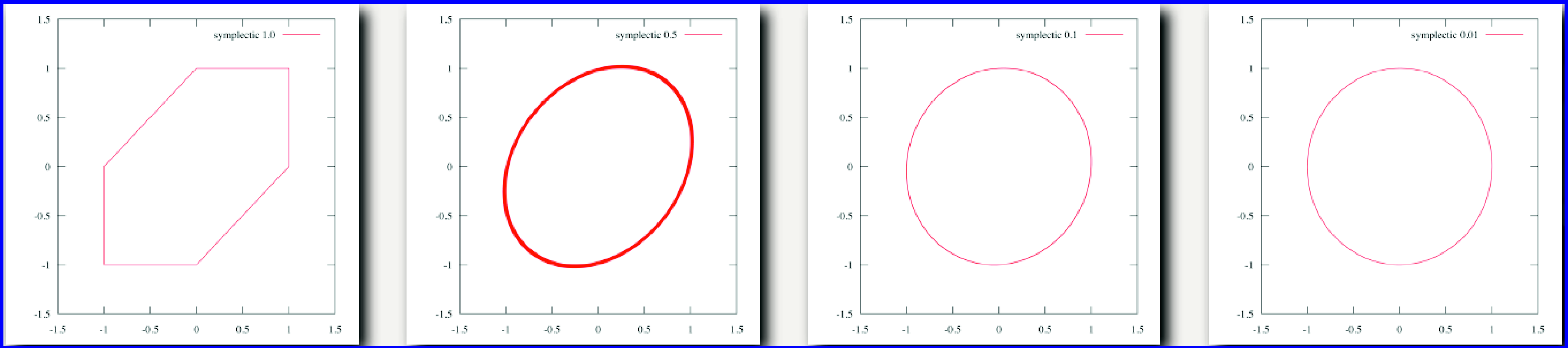
Symplectic *h = 1.0*



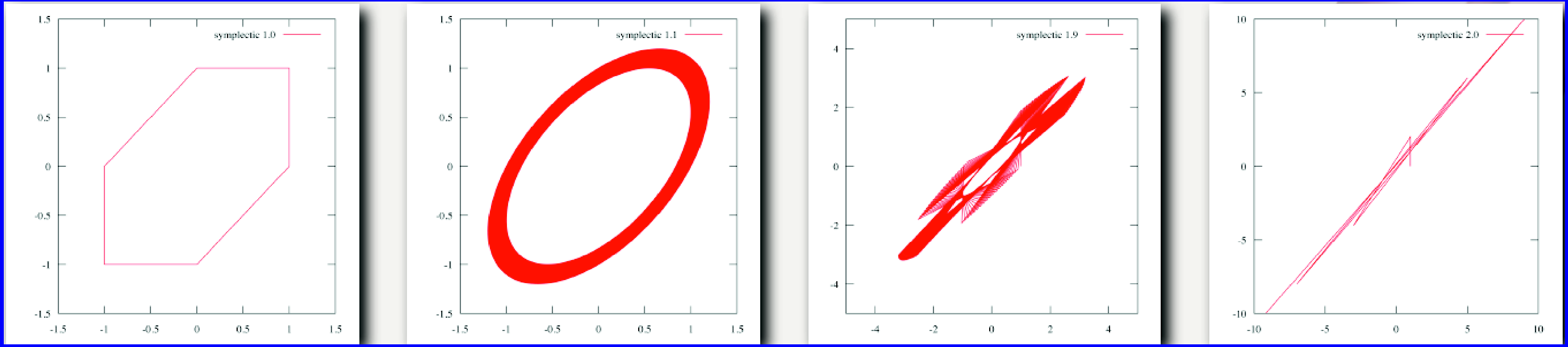
Symplectic *h = 0.01*

Long-term Evolution

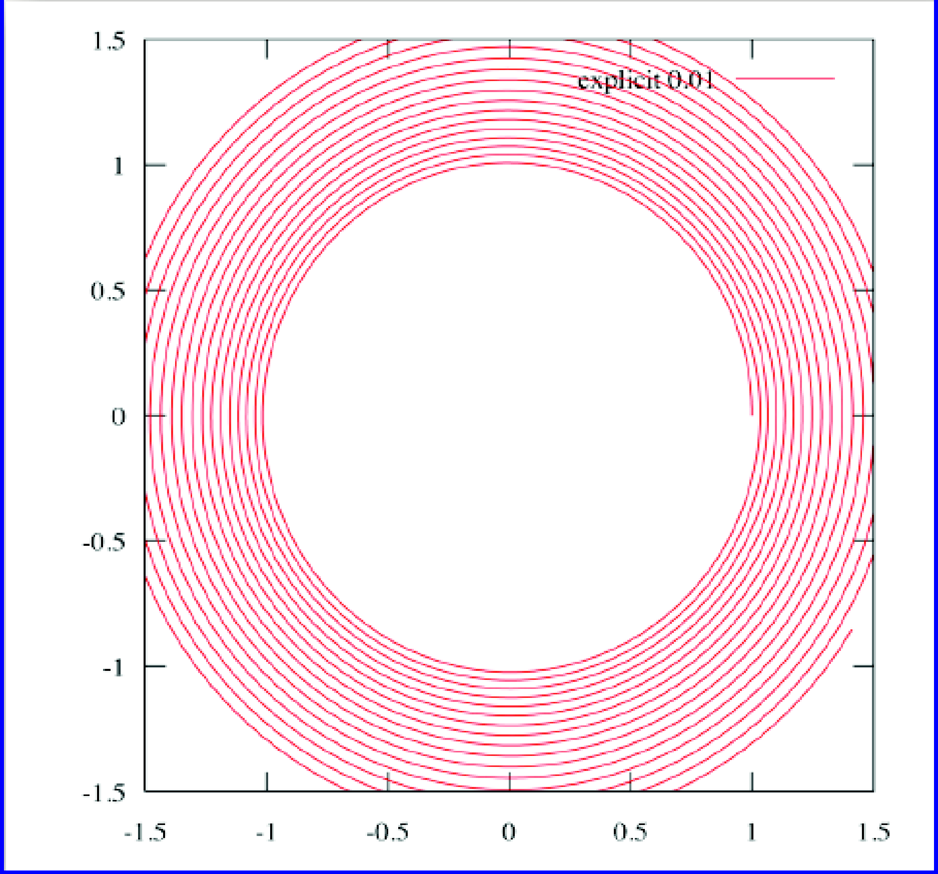
- Decreasing time step:



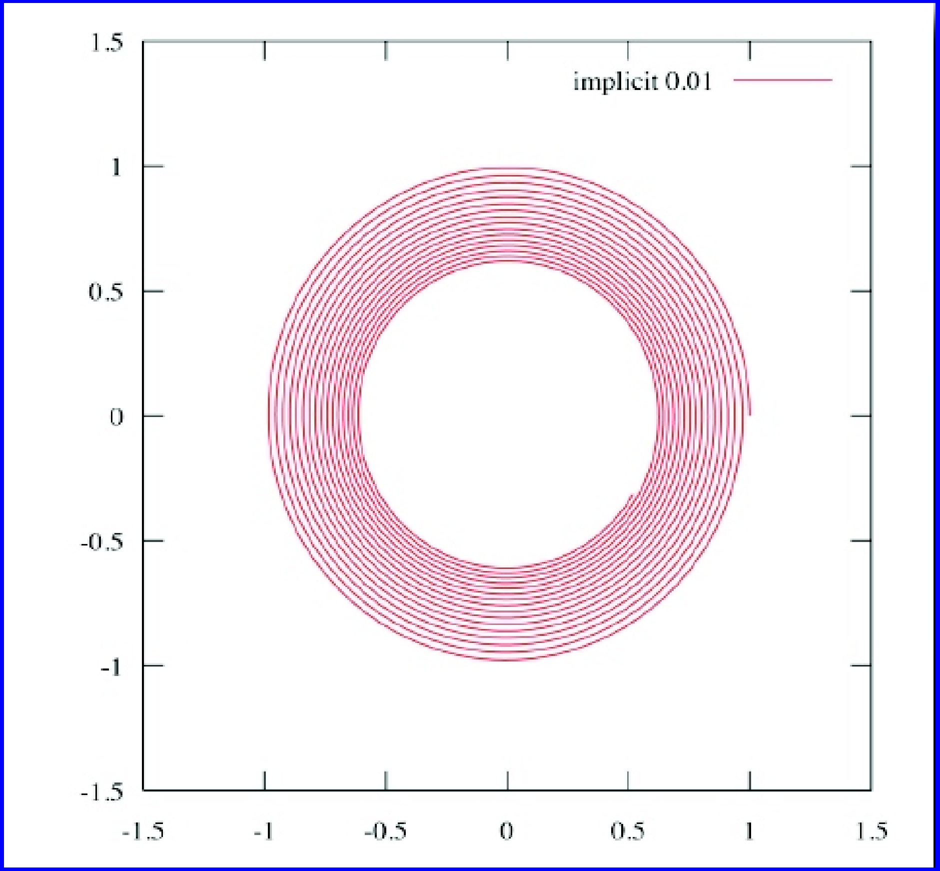
- Increasing time step:



Long-term Evolution



Explicit $h = 0.01$



Implicit $h = 0.01$

Why it works ?

$$x_{i+1} = x_i - h\, y_i$$
$$y_{i+1} = y_i + h\, x_{i+1}$$

- Integrator can be written as:

$$\mathbf{x}_{i+1} = \begin{bmatrix} 1 & -h \\ h & 1-h^2 \end{bmatrix} \mathbf{x}_i$$

- Which means:

$$\mathbf{x}_i = \begin{bmatrix} 1 & -h \\ h & 1-h^2 \end{bmatrix}^i \mathbf{x}_0 = A^i \mathbf{x}_0$$

- Stable iff. $\boxed{\mu(A) < 1}$, $\mu(A) = |\lambda_{max}|$, $\mu(A)$ spectral radius.
- Our case: $\mu(A) < 1$ if $h < 2$.

Symplectic integration

- Is not general:
  - Hamiltonian systems (total energy is potential energy + kinetic energy).
- Also called semi-implicit integration:
  - Verlet integrator.
  - Semi-implicit Euler.
- Numerical integration is subtle !
  - Small changes can have profound long-term effects.

Agenda

- Differential Constraints
- Project 1 – overview
- Presentation schedule

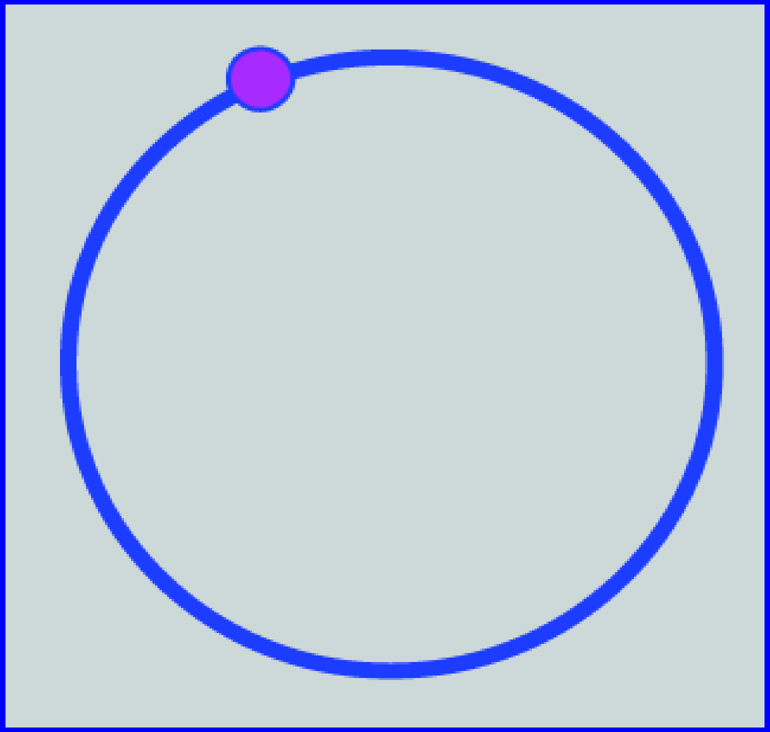
Differential Constraints

Based on Witkin's lecture notes

Constrained dynamics

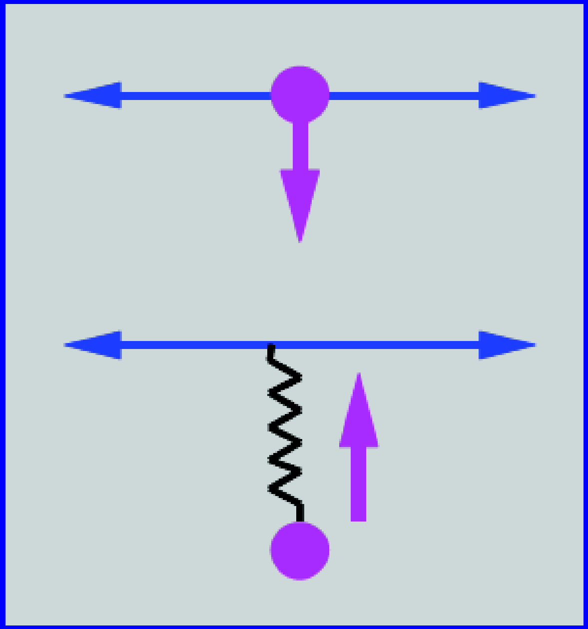
- Beyond Points and springs
  - We can make about anything out of point masses and springs, *in principle*...
  - **Constrained dynamics (particle systems):**
    - particles obey Newton's laws and (geometric) constraints.
    - Example: spring force competes with all other forces (gravity, other springs) to keep two particles at a given distance apart.
    - Gives rise to stiff systems.
    - So far we used additional energy terms to impose constraints: *penalty method*.
    - Accurate constraints and numerical tractibility – penalty methods do not work well.

Example. A bead on a wire



- Desired behaviour:
  - The bead can slide freely along the unit circle.
  - It can never come off, however hard we pull.

Penalty Constraints



- Why not use a spring to hold the particle on the wire ?
- Problem:
  - Weak springs: weak constraints.
  - Strong springs: Neptune express!
- A classical stiff system.

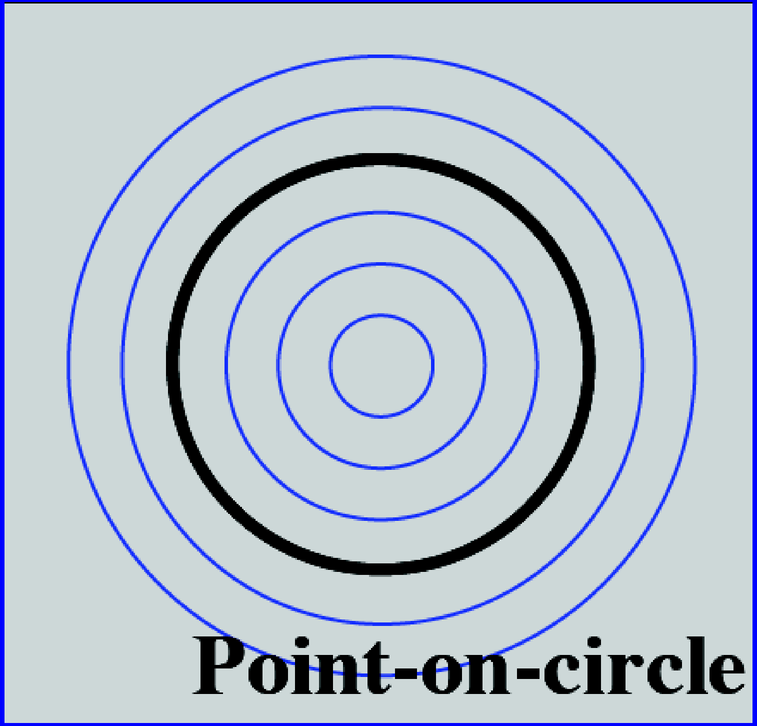
Solution. Constraint forces

- Directly calculate forces required to maintain the constraints.
- Instead of relying on displacements and restoring forces to do the job.

- Constraint forces: cancel just those parts of the applied forces that act against the constraints.
- Convert particle accelerations into “legal” accelerations.

Now for the Algebra...

- First, a single constrained particle...
- Then, generalize to constrained particle systems.



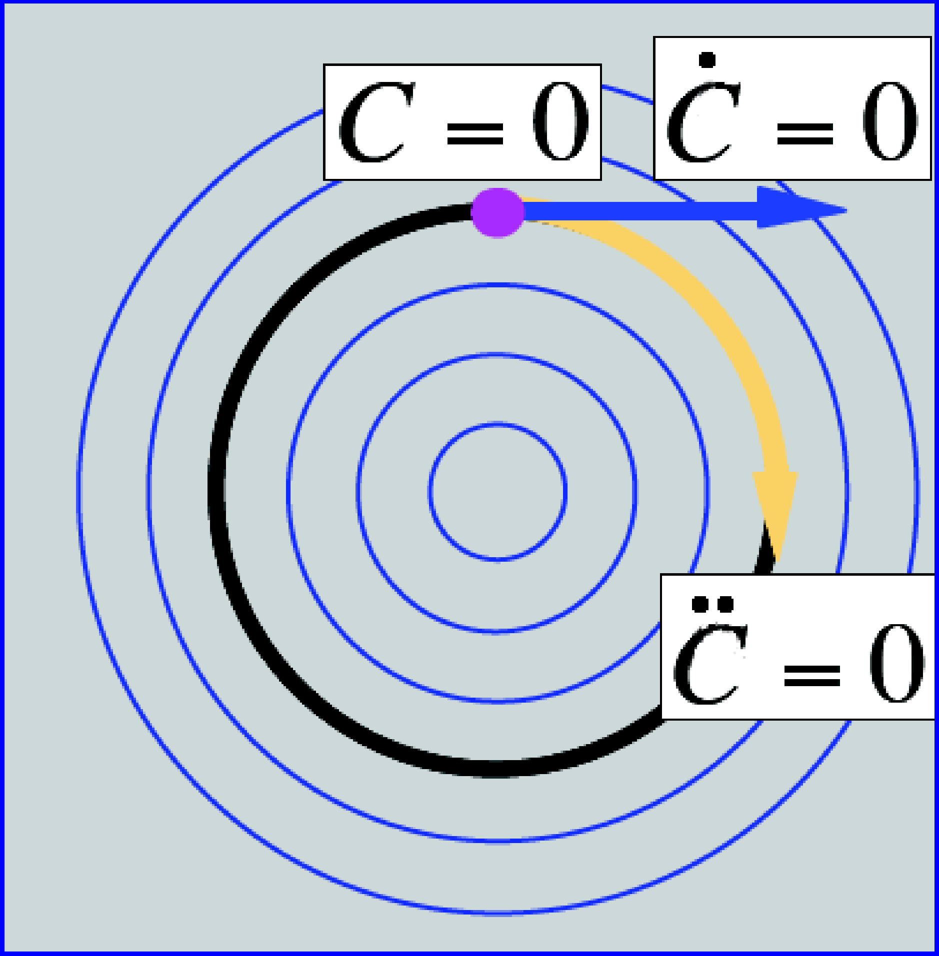
1. Implicit:

$$C(\mathbf{x}) = \frac{1}{2}(\mathbf{x} \cdot \mathbf{x} - 1) = 0$$

2. Parametric:

$$\mathbf{x} = [\cos\theta \;\; \sin\theta]$$

Maintaining constraints differentially



- Start with legal position and velocity.
- Use constraint forces to ensure legal velocity and acceleration.

$C = 0$ – legal position
$\dot{C} = 0$ – legal velocity
$\ddot{C} = 0$ – legal acceleration

Calculus...

$$\boxed{C(\mathbf{x}) = \frac{1}{2}(\mathbf{x}\cdot\mathbf{x} - 1) = 0}$$

- Legal velocity:

$$\dot{C} = \mathbf{x}\cdot\dot{\mathbf{x}} = 0$$

- Legal acceleration:

$$\ddot{C} = \ddot{\mathbf{x}}\cdot\mathbf{x} + \dot{\mathbf{x}}\cdot\dot{\mathbf{x}} = 0$$

- Particle's acceleration:

$$\ddot{\mathbf{x}} = \frac{\mathbf{f} + \hat{\mathbf{f}}}{m}$$

- Substitution gives: $\ddot{C} = \frac{\mathbf{f} + \hat{\mathbf{f}}}{m}\cdot\mathbf{x} + \dot{\mathbf{x}}\cdot\dot{\mathbf{x}} = 0$
- Or: $\boxed{\hat{\mathbf{f}}\cdot\mathbf{x} = -\mathbf{f}\cdot\mathbf{x} - m\dot{\mathbf{x}}\cdot\dot{\mathbf{x}}}$

Principle of virtual work

- One equation with two unknowns:

$$\boxed{\hat{\mathbf{f}} \cdot \mathbf{x} = -\mathbf{f} \cdot \mathbf{x} - m\dot{\mathbf{x}} \cdot \dot{\mathbf{x}}}$$

- Additional condition is needed:
  - Require that the constraint force never adds/removes (kinetic) energy to/from the system.
  - Kinetic energy: $T = \frac{m}{2}\dot{\mathbf{x}} \cdot \dot{\mathbf{x}}$
  - Its time derivative: $\dot{T} = m\ddot{\mathbf{x}} \cdot \dot{\mathbf{x}} = \left(\mathbf{f} + \hat{\mathbf{f}}\right) \cdot \dot{\mathbf{x}}$
  - Require: $\hat{\mathbf{f}} \cdot \dot{\mathbf{x}} = 0, \ \forall \dot{\mathbf{x}} \,|\, \mathbf{x} \cdot \dot{\mathbf{x}} = 0$

legal velocity

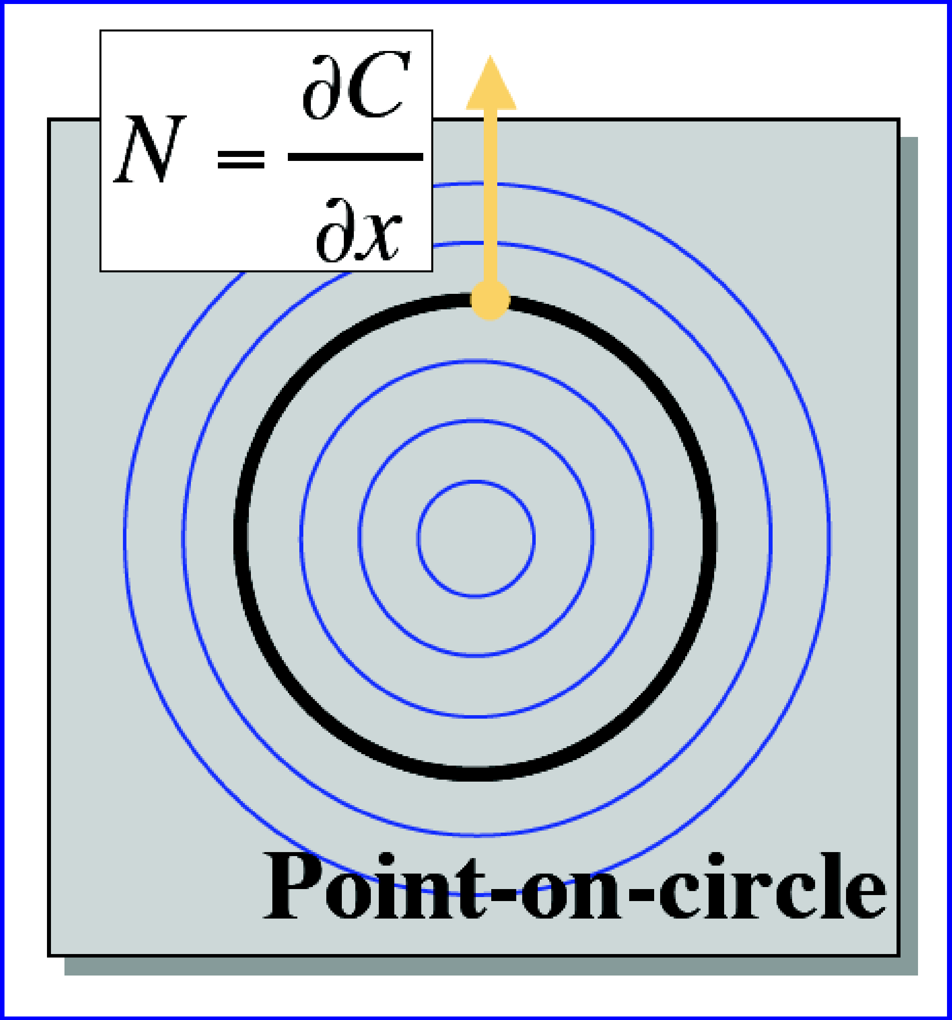
Constraint force

- Condition $\hat{\mathbf{f}} \cdot \dot{\mathbf{x}} = 0, \;\; \forall \dot{\mathbf{x}} \,|\, \mathbf{x} \cdot \dot{\mathbf{x}} = 0$ states that $\hat{\mathbf{f}}$ should point in the direction of $\mathbf{x}$.
- Implies: $\hat{\mathbf{f}} = \lambda \mathbf{x}$
- Solve $\hat{\mathbf{f}} \cdot \mathbf{x} = -\mathbf{f} \cdot \mathbf{x} - m\dot{\mathbf{x}} \cdot \dot{\mathbf{x}}$ for lambda gives

$$\boxed{\lambda = \frac{-\mathbf{f} \cdot \mathbf{x} - m\dot{\mathbf{x}} \cdot \dot{\mathbf{x}}}{\mathbf{x} \cdot \mathbf{x}}}$$

- Remark: At rest $\hat{\mathbf{f}} = -\dfrac{\mathbf{f} \cdot \mathbf{x}}{\mathbf{x} \cdot \mathbf{x}} \mathbf{x}$. Constraint force is simply an orthogonal projection of $\mathbf{f}$ onto the circle's normal.

Interpretation. Constraint Gradient



$$C(\mathbf{x}) = \frac{1}{2}(\mathbf{x} \cdot \mathbf{x} - 1) = 0$$

- Differentiating C gives a normal vector to the circle.
- This is the direction our constraint force acts (steepest descent).

Generalization. Particle system

- We know how to simulate a point on a wire.
- Next: a constrained particle system.

Compact Particle System Notation

$$\ddot{q} = WQ$$

q: ***3n*-long *state vector.***

Q: ***3n*-long *force vector.***

M: ***3n* x *3n* diagonal *mass matrix.***

W: **M-inverse (element- wise reciprocal)**

$$q = [x_1, x_2, \quad , x_n]$$

$$Q = [f_1, f_2, \quad , f_n]$$

$$M = \begin{bmatrix} m_1 & & & & & \\ & m_1 & & & & \\ & & m_1 & & & \\ & & & m_n & & \\ & & & & m_n & \\ & & & & & m_n \end{bmatrix}$$

$$W = M^{-1}$$

$$C = [C_1, C_2, \quad , C_m]$$

$$\lambda = [\lambda_1, \lambda_2, \quad , \lambda_m]$$

Solving for Constraint Force

- Assume that initially both configurations $\mathbf{q}$ and $\dot{\mathbf{q}}$ are legal.
- Problem is to solve for constraint force (vector) $\hat{\mathbf{Q}}$ that, added to the applied force $\mathbf{Q}$ guarantees that $\ddot{\mathbf{C}} = 0$.
- By the Chain rule, $$\dot{\mathbf{C}} = \frac{\partial C}{\partial \mathbf{q}} \dot{\mathbf{q}}^T = \mathbf{J}\dot{\mathbf{q}}^T$$
- Differentiating again w.r.t. time gives

$$\ddot{\mathbf{C}} = \dot{\mathbf{J}}\dot{\mathbf{q}}^T + \mathbf{J}\ddot{\mathbf{q}}^T, \text{ with } \dot{\mathbf{J}} = \frac{\partial \dot{\mathbf{C}}}{\partial \mathbf{q}}$$

Solving for Constraint Force

- Using the motion equation, setting $\ddot{\mathrm{C}} = 0$ and re-arranging gives

$$\mathbf{JW}\hat{\mathbf{Q}}^T = -\dot{\mathbf{J}}\dot{\mathbf{q}}^T - \mathbf{JWQ}^T$$

- As in the point-on-wire example: more unknowns than equations.
- Use again the principle of virtual work according to which legal velocities satisfy

$$\hat{\mathbf{Q}} \cdot \dot{\mathbf{q}}^T = 0, \ \forall \dot{\mathbf{q}} \,|\, \mathbf{J}\dot{\mathbf{q}}^T = 0$$

- Implies: $\hat{\mathbf{Q}} = \lambda \mathbf{J}$

Solving for Constraint Force

- $\hat{\mathbf{Q}} = \lambda \mathbf{J}$: constraint force is a linear combination of constraint gradients.
- Since we require $\mathbf{C} = 0$, these gradients are normals to the hypersurfaces representing state-space directions in which the system *should not* move.
- Vector λ called: the vector of Lagrange multipliers.

Solving for Constraint Force

- We have $\hat{\mathrm{Q}} = \lambda \mathbf{J}$ and $\mathbf{J}\mathbf{W}\hat{\mathbf{Q}}^T = -\dot{\mathbf{J}}\dot{\mathbf{q}}^T - \mathbf{J}\mathbf{W}\mathbf{Q}^T$.
- Solve linear system to yield:

$$\boxed{\lambda^T = (\mathbf{J}\mathbf{W}\mathbf{J}^T)^{-1}(-\dot{\mathbf{J}}\dot{\mathbf{q}}^T - \mathbf{J}\mathbf{W}\mathbf{Q}^T)}$$

  - $\mathbf{J}\mathbf{W}\mathbf{J}^T$ is a square matrix of dimension same as $\mathbf{C}$.
  - Once λ^T is obtained, $\boxed{\hat{\mathrm{Q}} = \lambda \mathbf{J}}$ can be computed.

Drift and Feedback

- Numerical solutions of ODE's start drifting.
- Numerical errors can accumulate.
  - Numerical drift causes the circle to turn into a spiral.
- Constraints might not be met initially (invalid initial conditions).

- A feedback term (damped spring force) handles these problems:

$$\boxed{\ddot{\mathbf{C}} = -k_s \mathbf{C} - k_d \dot{\mathbf{C}}}$$

k_s, k_d are spring and damping constants.

- Feedback added *after* the constraint force calculation.

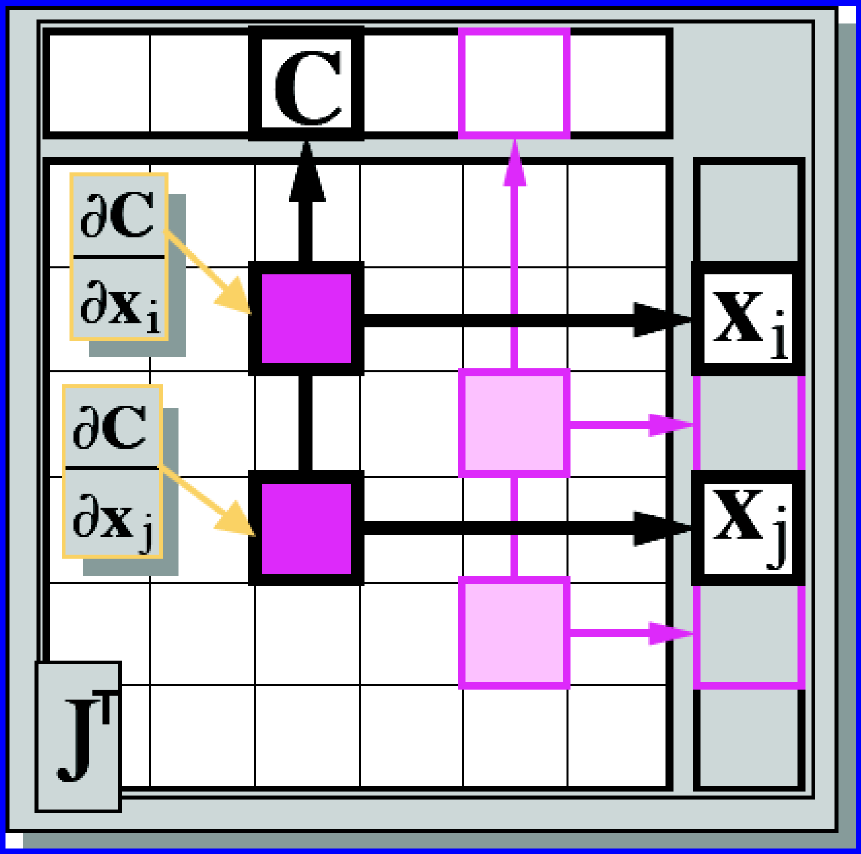
How to implement all this ?

- We have a global matrix equation.
- We want to build models and add constraints on the fly.

- Approach:
  - Each constraint adds its own piece to the equation.
  - Each constraint object is responsible for evaluating the constraint function and function derivatives.

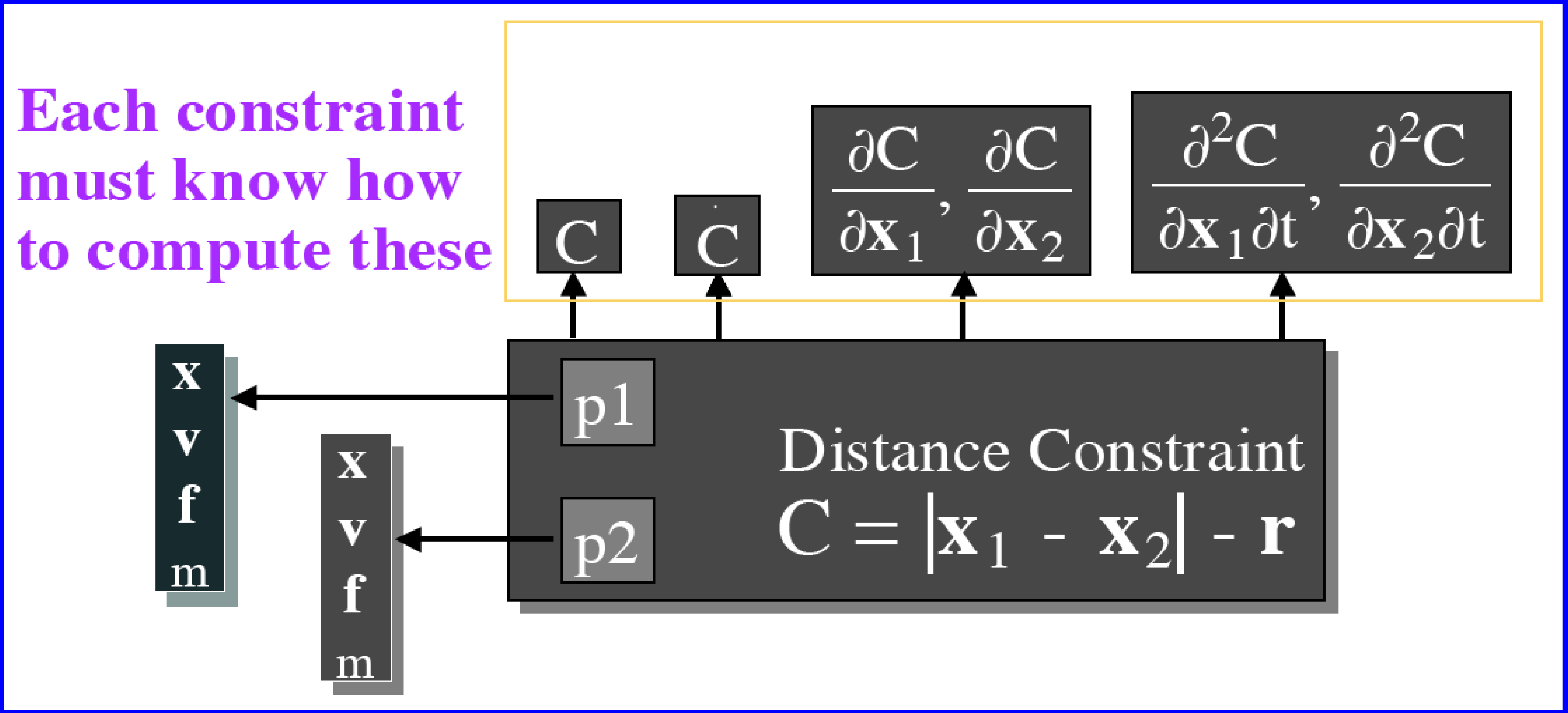
Jacobian: sparse-block structure

Global matrix:
block structure

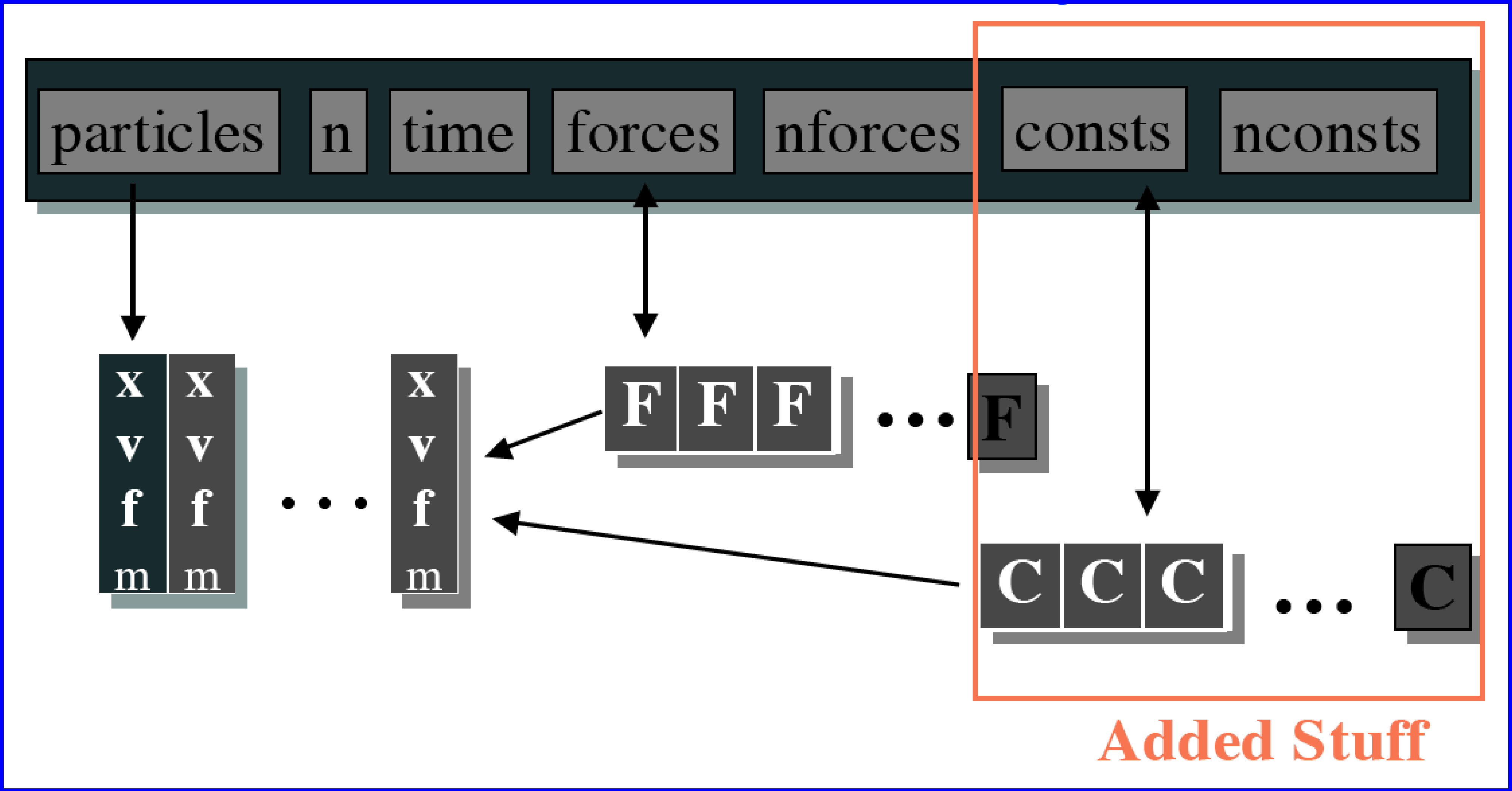


- Each constraint contributes one or more blocks to the matrix.
- Each block is the Jacobian of the constraint.
- Sparsity: many empty blocks.
- Modularity: each constraint computes its own blocks.
- Constraint and particles indices determine block locations within global “Jacobian” matrix.

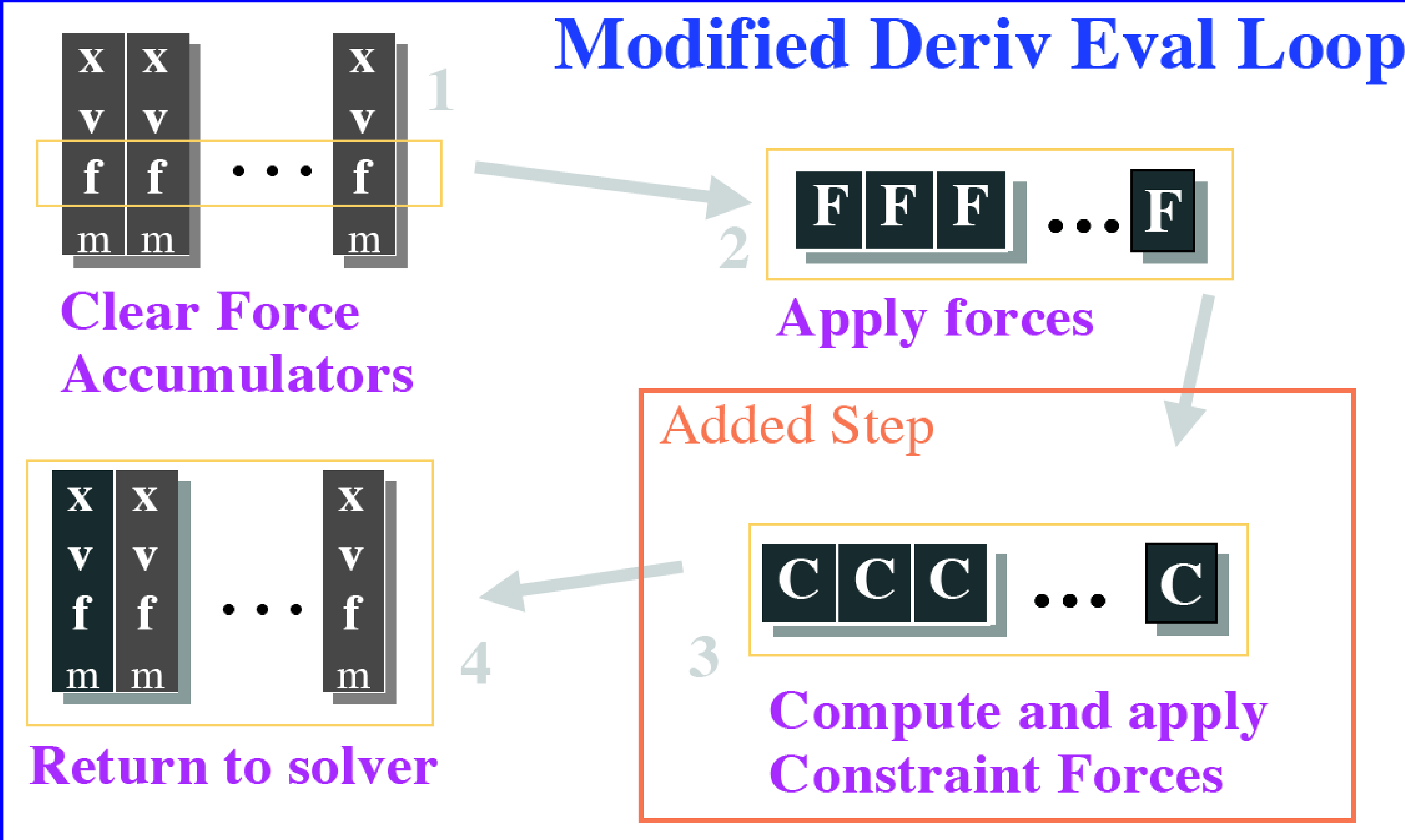
Constraint Structure



Constrained Particle System



DerivEval Loop (modified)



Constraint Force Eval

$$\lambda^T = (\mathbf{JWJ}^T)^{-1}(-\dot{\mathbf{J}}\dot{\mathbf{q}}^T - \mathbf{JWQ}^T)$$

- After computing ordinary forces:
  - Loop over constraint objects, assemble global matrices $\mathbf{J}, \dot{\mathbf{J}}$ and vectors to hold $\mathbf{C}, \dot{\mathbf{C}}$, etc.
  - Call linear solver to get λ, multiply by $\mathbf{J}$ to get constraint force vector.
  - Add constraint force to particle force accumulators.
  - Add feedback force.

Impress your friends :)

- Constraints do not add/remove energy: based on the *Principle of Virtual Work*.
- λ's are called *Lagrange Multipliers*.
- The matrix of derivatives, $\mathbf{J}$, is called the *Jacobian matrix*.

Constrained particle systems

- Explicit/implicit/semi-implicit integration.
- Constraints: penalty method, Lagrange multipliers.
- Simple collisions.

Next time

- Lecture: PDEs – basics
- Presentations: cloth simulation.

Agenda

- Differential Constraints
- Project 1 – overview
- Presentation schedule

Organization. Presentations

Cloth 14.05

- Sander Jurgens, Edmond van Dijk, Maikel Leemans: provot95
- Bram Cappers, Wouter van Heeswijk: bridson02

Hair 17.05

- Rick Hendricksen, Freek Marcelis: mwsst09
- ...

Fluids 31.05

- Marijn Grootjans, Sander den Heijer: foster96
- …

Fluid Coupling 04.06

- Stefan Patelski, Martin Schaefer: spray
- …

Rigid body collisions 11.06

- Michel Cijsouw, Leroy Bakker, wilco van Maanen: bender06
- …