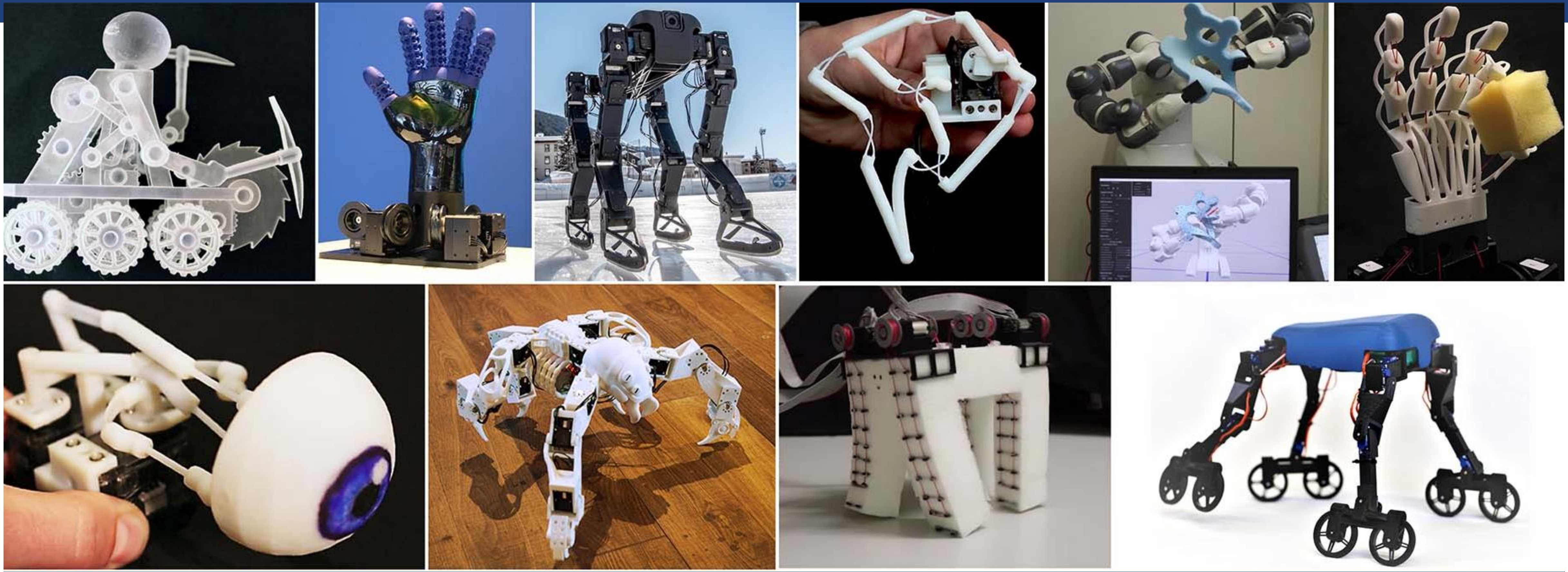
ETH zürich



Computational Models of Motion

Dynamics

CRL

Dynamic Motion

How do things move?

Why do they move?

How can we model dynamic motion computationally?

Kinematics vs Dynamics

kinematics

/ˌkɪnɪˈmatɪks,ˌkʌɪnɪˈmatɪks/

noun

the branch of mechanics concerned with the motion of objects without reference to the forces which cause the motion.

dynamics

/dʌɪˈnamɪks/

noun

noun: **dynamics**; plural noun: **dynamics**

the branch of mechanics concerned with the motion of bodies under the action of forces.

Connection between Force and Motion

Newton's Second Law of Motion

"A change in motion is proportional to the motive force impressed and takes place along the straight line in which that force is impressed."

—Sir Isaac Newton, 1687

$$\underbrace{F}_{\text{force}} = \overbrace{m}^{\text{mass}}\overbrace{a}^{\text{acceleration}}$$

Some Background on Newtonian Mechanics

Newtonian Mechanics – Single Particle

- Consider a single particle with mass m and position $\boldsymbol{x}(t) \in \boldsymbol{R}^3$

- Velocity is the rate of change of positions

$$\boldsymbol{v}(t) = \frac{d\boldsymbol{x}(t)}{dt} = \dot{\boldsymbol{x}}(t)$$

- Acceleration is the rate of change of velocity

$$\boldsymbol{a}(t) = \frac{d\boldsymbol{v}(t)}{dt} = \dot{\boldsymbol{v}}(t) = \ddot{\boldsymbol{x}}(t)$$

- Linear momentum

$$\boldsymbol{p}(t) = m\boldsymbol{v}(t)$$

- Newton's 2nd law of motion

$$\boldsymbol{f}(t) = m\boldsymbol{a}(t) = m\ddot{\boldsymbol{x}}(t) = m\dot{\boldsymbol{v}}(t) = \dot{\boldsymbol{p}}(t)$$

Newtonian Mechanics – Systems of Particles

- Consider a system of n particles with masses m_i and positions $\boldsymbol{x}_i(t)$
- For each particle, we have $\boldsymbol{f}_i(t) = m\boldsymbol{a}_i(t)$
- Combine into single equation $\boldsymbol{f} = \boldsymbol{M}\boldsymbol{a}$
  - Mass matrix $\boldsymbol{M} = diag(m_1, m_1, m_1, \dots, m_n, m_n, m_n) \in \boldsymbol{R}^{3n\times 3n}$
  - $\boldsymbol{f} = \left(\boldsymbol{f}_1^T, \dots, \boldsymbol{f}_n^T\right)^T \in \boldsymbol{R}^{3n}$, $\boldsymbol{a} = \left(\boldsymbol{a}_1^T, \dots, \boldsymbol{a}_n^T\right)^T \in \boldsymbol{R}^{3n}$
- Consider sum of per particle equations

$$\sum_i \boldsymbol{f}_i = \sum_i m_i \boldsymbol{a}_i = \sum_i \dot{\boldsymbol{p}}_i$$

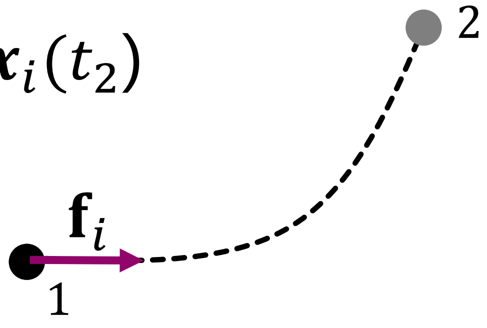
→ Total linear momentum $\boldsymbol{P} = \sum_i \boldsymbol{p}_i$ is conserved if net force is zero,

$$\boldsymbol{F} = \sum_i \boldsymbol{f}_i = \boldsymbol{0} \leftrightarrow \sum_i \boldsymbol{p}_i = \dot{\boldsymbol{P}} = \boldsymbol{0} \leftrightarrow \boldsymbol{P} = const.$$

CRL

Work and Kinetic Energy

- The work W_i done on a particle between position $\boldsymbol{x}_i(t_1)$ and $\boldsymbol{x}_i(t_2)$



$$W_i = \int_1^2 \mathbf{f}_i \cdot d\boldsymbol{x}_i$$

This is a dot product!

- Transforming the integrand

$$\boldsymbol{f}_i d\boldsymbol{x}_i = \frac{d\boldsymbol{p}_i}{dt} \cdot d\boldsymbol{x}_i = m\frac{d\boldsymbol{v}_i}{dt} \cdot \boldsymbol{v}_i dt = \frac{m}{2}\frac{d}{dt}(\boldsymbol{v}_i \cdot \boldsymbol{v}_i)dt = d\left(\frac{1}{2}m\boldsymbol{v}^2\right)dt$$

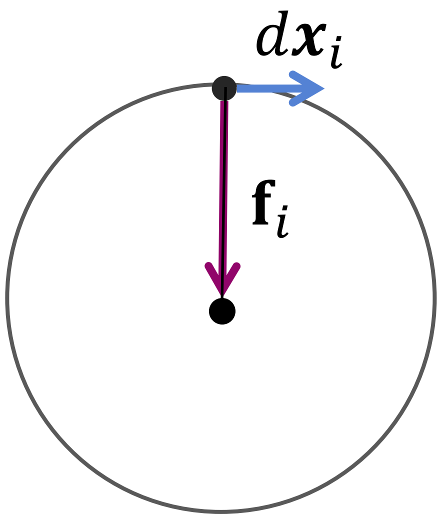
Kinetic energy T

- Hence

$$\int_1^2 \mathbf{f}_i \cdot d\boldsymbol{x}_i = \int_1^2 dT dt = T_2 - T_1$$

Work and Kinetic Energy

$$\int_1^2 \boldsymbol{f}_i \cdot d\boldsymbol{x}_i = \int_1^2 dTdt = T_2 - T_1$$



Two special cases

- Force does no work (e.g., particle moving on string)

  $\int \boldsymbol{f}_i d\boldsymbol{x}_i = 0 \;\rightarrow\; T_2 = T_1$

  $\rightarrow$ kinetic energy T is constant

- Force derives from a potential, i.e., $\boldsymbol{f}_i = -\frac{\partial U}{\partial \boldsymbol{x}_i}$

  $\int \boldsymbol{f}_i d\boldsymbol{x}_i = \int -\frac{\partial U}{\partial \boldsymbol{x}_i} d\boldsymbol{x}_i = U_1 - U_2 \rightarrow -\Delta U = \Delta T$

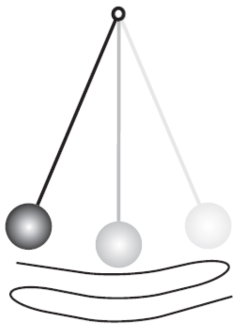
  $\rightarrow$ total energy $E = U + T$ is constant

Generalized Coordinates

State Representations

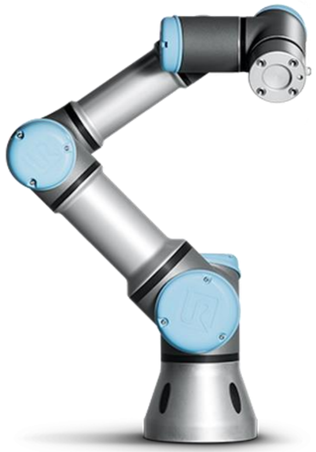
- So far, we represented particle systems using per-particle positions $\boldsymbol{x}_i(t)$ and velocities $\boldsymbol{v}_i(t)$ in Cartesian coordinates

Is this representation always ideal?



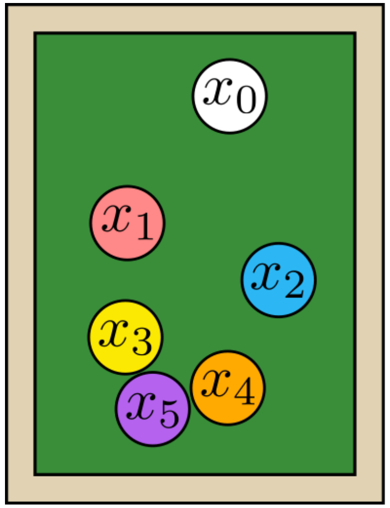
Angle between chord and vertical axis

Position and orientation

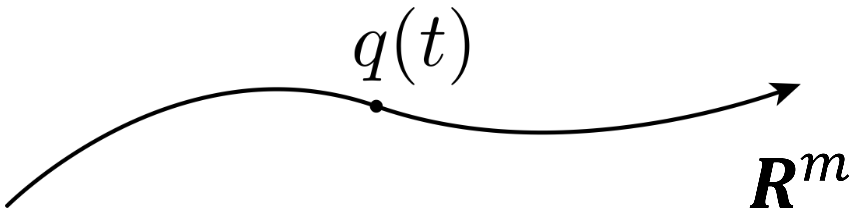


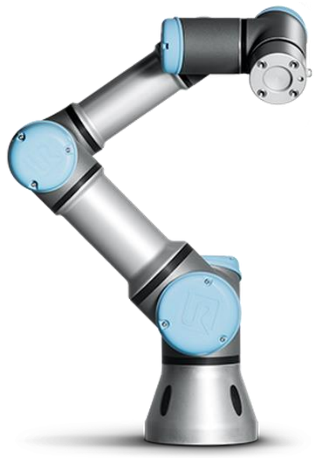
Position and orientation of base + 1 angle per motor

Generalized Coordinates



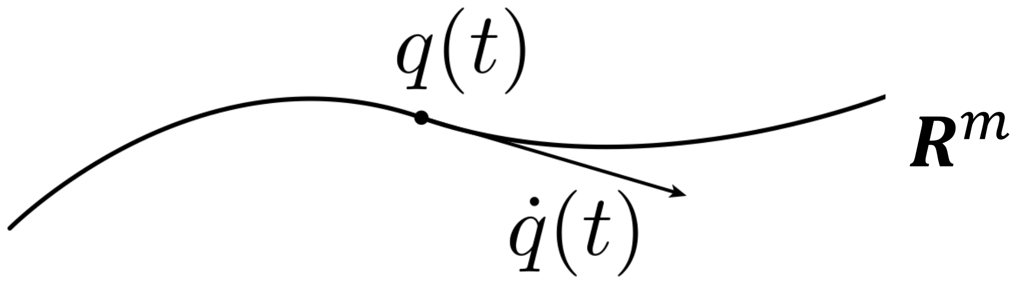
- Generalized coordinates $\boldsymbol{q}$ abstract away the concrete representation of the degrees of freedom (DOF) of a system
- Examples
  - A collection of m billiard balls $\boldsymbol{q} = \left(\boldsymbol{x}_1^t, \dots, \boldsymbol{x}_m^t\right)^t$
  - A 6-axis robot arm $\boldsymbol{q} = (\alpha_1, \dots, \alpha_6)^t$
- Can think of $\boldsymbol{q}$ as a *single point* moving along a trajectory in $\boldsymbol{R}^m$

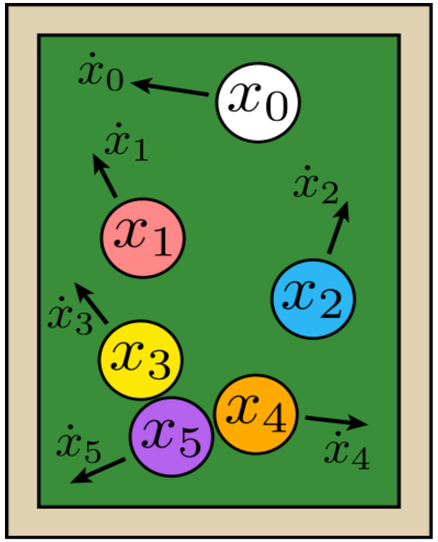




Generalized Velocity

- The generalized velocity $\dot{\boldsymbol{q}}$ is simply the time derivative of the generalized coordinates





- For a collection of m billiard balls, $\dot{\boldsymbol{q}} = \left(\dot{\boldsymbol{x}}_1^t, \ldots, \dot{\boldsymbol{x}}_m^t\right)^t$
- *State* of the system is described by the pair $\boldsymbol{s}(t) = (\boldsymbol{q}^t(t), \dot{\boldsymbol{q}}^t(t))$
- The set of all possible $\boldsymbol{s}(t)$ is referred to as the *state space* or *phase space*

Linking Representations

- Can interpret generalized coordinates as reparameterization of particle system with Cartesian coordinates.
- Particle positions are functions of generalized coordinates

$$\boldsymbol{x}_i(t) = \boldsymbol{x}_i\big(\boldsymbol{q}(t)\big)$$

- Particle velocities are functions of generalized velocities

$$\dot{\boldsymbol{x}}_i(t) = \frac{d\boldsymbol{x}_i}{dt} = \frac{\partial \boldsymbol{x}_i}{\partial \boldsymbol{q}}\frac{d\boldsymbol{q}}{dt} = \boldsymbol{J}_i\dot{\boldsymbol{q}}(t)$$

where $\boldsymbol{J}_i = \frac{\partial \boldsymbol{x}_i}{\partial \boldsymbol{q}} \in \boldsymbol{R}^{3\times m}$ is the Jacobian of the map $\boldsymbol{q} \to \boldsymbol{x}_i(\boldsymbol{q})$.

Generalized Coordinates – Equations of Motion

- Newton's 2nd law of motion in generalized form

$$\boldsymbol{f}_q(\boldsymbol{q}, \dot{\boldsymbol{q}}, t) = \boldsymbol{M}_q(\boldsymbol{q})\ddot{\boldsymbol{q}}$$

  - Generalized forces $\boldsymbol{f}_q \in \boldsymbol{R}^m$
    - Can depend on generalized position, generalized velocity and time
  - Generalized mass matrix $\boldsymbol{M}_q \in \boldsymbol{R}^{m\times m}$
    - $\boldsymbol{M}_q(\boldsymbol{q})$ generally depends on configuration, changes with time
    - $\boldsymbol{M}_q$ is generally dense and always symmetric since

$$\frac{1}{2}\dot{\boldsymbol{q}}^t\boldsymbol{M}_q\dot{\boldsymbol{q}} = \frac{1}{2}\dot{\boldsymbol{x}}^t\boldsymbol{J}^t\boldsymbol{M}\boldsymbol{J}\dot{\boldsymbol{x}}$$

  must give kinetic energy.

Generalized Coordinates – Equations of Motion

- Newton's 2nd law of motion in generalized form
$$\boldsymbol{f}_q(\boldsymbol{q}, \dot{\boldsymbol{q}}, t) = \boldsymbol{M}_q(\boldsymbol{q})\ddot{\boldsymbol{q}}$$
- A second order ordinary differential equation (ODE)
- The unknown trajectory $\boldsymbol{q}(t)$ is described through its second derivative $\ddot{\boldsymbol{q}}(t)$
- In order to compute $\boldsymbol{q}(t)$, we have to *solve* the ODE

Ordinary Differential Equations

Ordinary Differential Equations

- An Ordinary Differential Equation (ODE) describes an unknown function $y(t)$ through its derivatives with respect to a **single** variable t,

$$y^{(n)} = f\left(t, y, y', \dots, y^{(n-1)}\right)$$

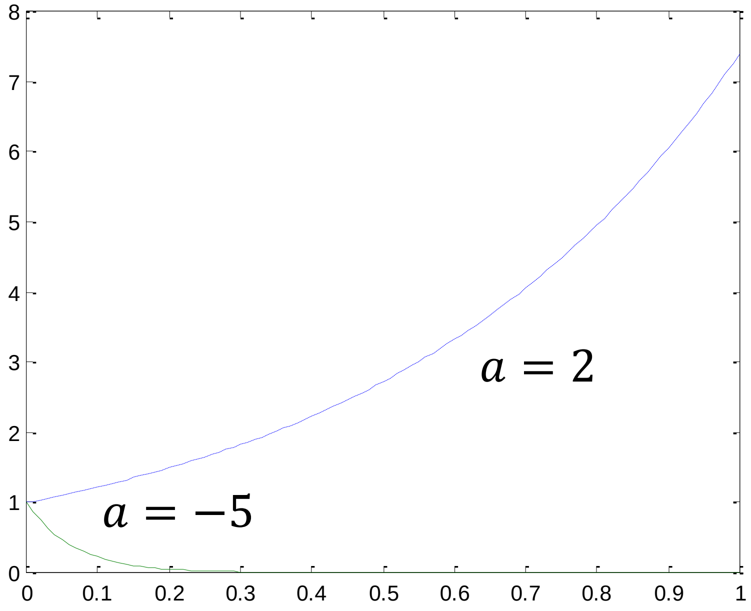
- The order of the ODE is the order of the highest derivative
- Example of a first order ODE

$$\dot{y} = ay \quad \text{with } a \in \boldsymbol{R}$$

- Solution?

$$y = ce^{at} \quad \text{with } c \in \boldsymbol{R}$$

  - Exponential growth for $a > 0$
  - Exponential decay for $a < 0$



Initial Value Problems

- Any function of the form $y = ce^{at}$ satisfies the ODE $\dot{y} = ay$
- To solve for $y(t)$, we need further information → initial conditions

$$y(0) = C \rightarrow y(t) = Ce^{at}$$

- Initial value problem (IVP): initial conditions + ODE

Is the solution unique?

Picard-Lindeloef-Theorem: if f is Lipschitz continuous (bounded variation), then the IVP $\dot{y} = f(t, y)$ with initial values $y(t_0) = y_0$ has a unique solution $y(t)$

Higher Order ODEs

- Newton's 2nd law is a second order ODE

$$\ddot{q} = f(t, q)/m$$

- Write as system of two first order ODEs by introducing *new* variables for velocity

$$\begin{aligned} \dot{q} &= v \\ \dot{v} &= f(t, q)/m \end{aligned} \qquad \frac{d}{dt}\begin{bmatrix} q \\ v \end{bmatrix} = \begin{bmatrix} v \\ f/m \end{bmatrix}$$

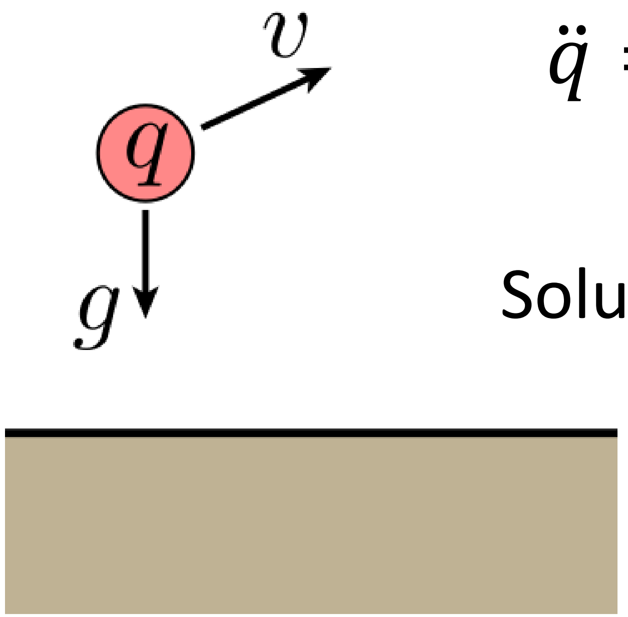
- Introduce state vector $\boldsymbol{y} = (q, v)^t$ and write system of ODEs as single ODE in standard form,

$$\boldsymbol{y}' = \boldsymbol{f}(t, \boldsymbol{y})$$

CRL

Simple Example: Throwing a Rock

- Consider a rock* of mass *m* tossed under force of gravity *g*
- The only active force is gravity → constant acceleration:



$$F = mg \qquad \dot{q} = v$$
$$\ddot{q} = g \quad \xrightarrow{\text{Transform to first order system}} \quad \dot{v} = g$$

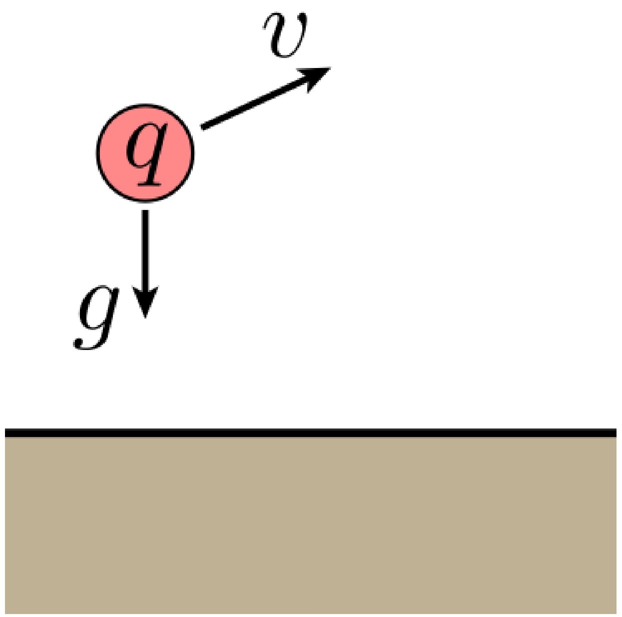
Solution: $$v(t) = v_0 + \frac{t}{m}F$$

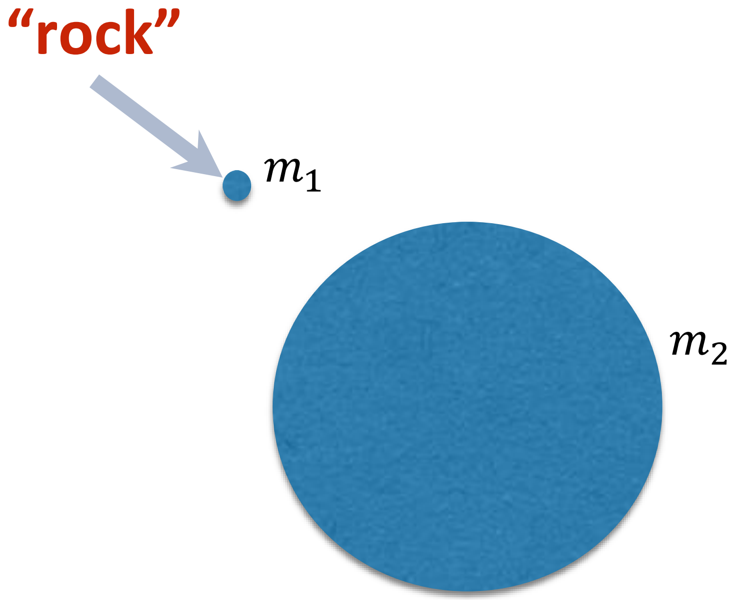
$$q(t) = q_0 + tv_0 + \frac{t^2}{2m}F$$

*This rock is spherical and has uniform density.

Simple Example: the two-body problem

- With non-constant gravitation forces

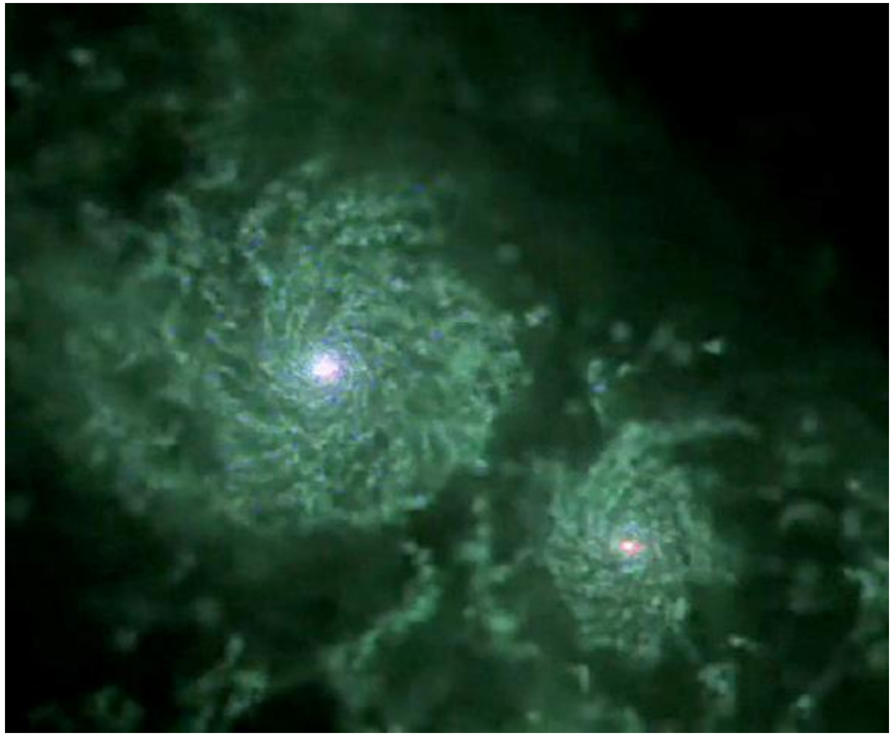




$$F_{12} = -Gm_1m_2 \frac{\boldsymbol{x}_1 - \boldsymbol{x}_2}{|\boldsymbol{x}_1 - \boldsymbol{x}_2|^3}$$

Not-So-Simple Example: *n*-Body Problem

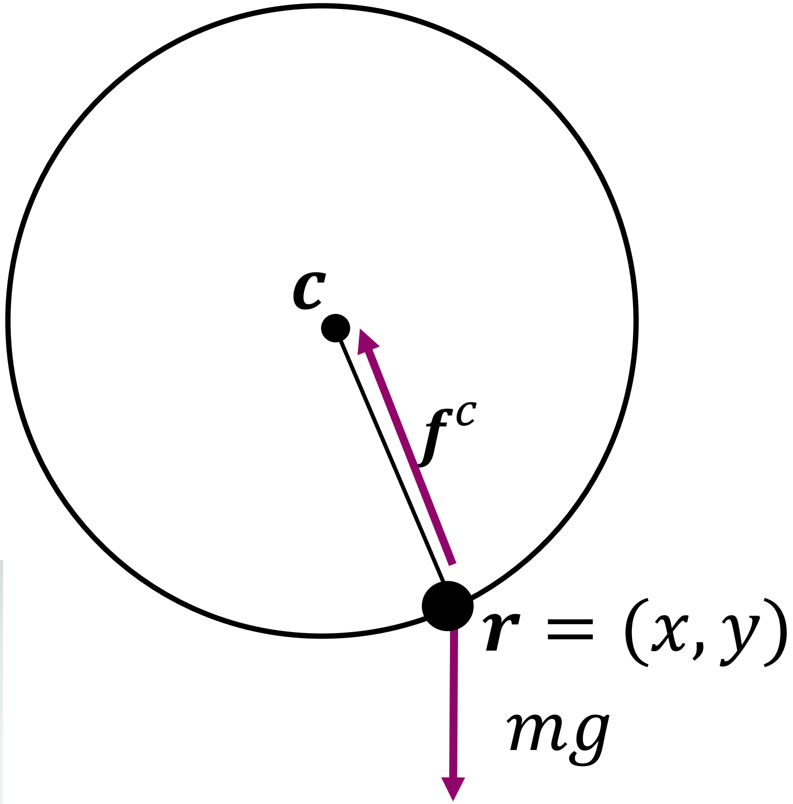
- Consider the Earth, moon, and sun—where do they go?
- As soon as $n > 3$, no closed form solutions exist
- What if we want to simulate entire galaxies?



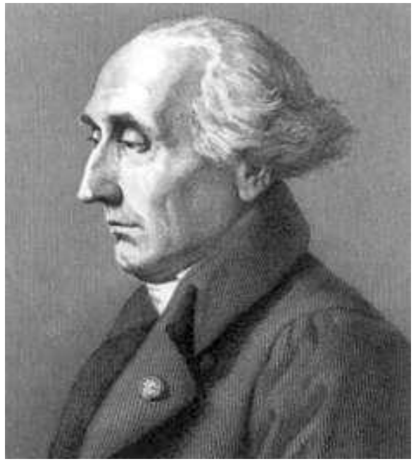
Credit: Governato et al / NASA

Slightly Harder Simple Example: Pendulum

- Point mass on string, swinging under gravity
- Same as "rock" problem, but constrained
- What are the equations of motion?
  - $ma_y = -mg + f_y^c$
  - $ma_x = f_x^c$
  - $\boldsymbol{f}^c = \lambda(\boldsymbol{r} - \boldsymbol{c})$



Lagrangian Mechanics



Joseph-Louis Lagrange

- Simple and general recipe:
  - Kinetic energy K
  - Potential energy U
  - Write down *Lagrangian* $\mathcal{L} := K - U$

Dynamics given by *Euler-Lagrange equation*

becomes (generalized) "MASS TIMES ACCELERATION" $\rightarrow$

$$\frac{d}{dt}\frac{\partial \mathcal{L}}{\partial \dot{q}} = \frac{\partial \mathcal{L}}{\partial q}$$

$\leftarrow$ becomes (generalized) "FORCE"

- Why is this useful?
  - often easier to find (scalar) energies than forces
  - very general, works with any kind of generalized coordinates

Applied to Pendulum

- Generalized coordinates for pendulum?

$$q = \theta$$

- Kinetic energy (mass *m*)?

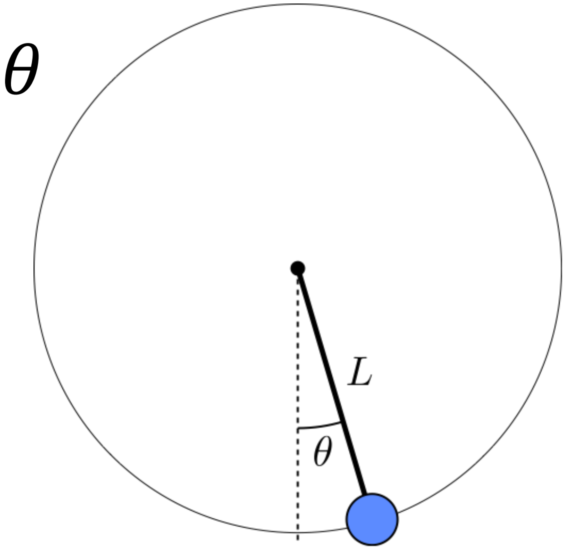
$$K = 1/2m(L\dot{\theta})^2$$

$$x = L\sin\theta, y = -L\cos\theta$$

$$\rightarrow \ |\boldsymbol{v}| = L\dot{\theta}$$

- Potential energy?

$$U = mgh = -mgL\cos\theta$$



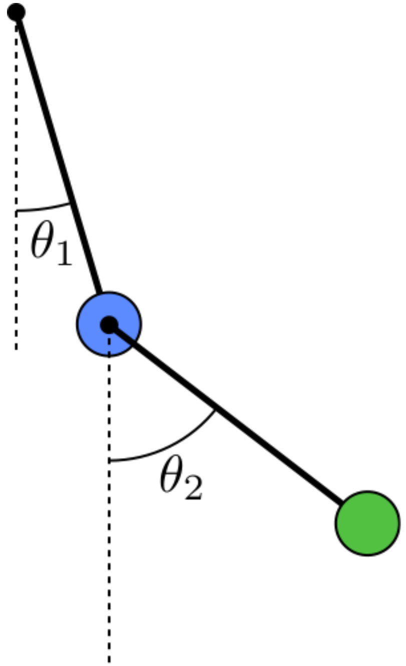
$$\mathcal{L} = K - U = m(\tfrac{1}{2}L^2\dot{\theta}^2 + gL\cos\theta)$$

$$\frac{\partial\mathcal{L}}{\partial\dot{q}} = \frac{\partial\mathcal{L}}{\partial\dot{\theta}} = mL^2\dot{\theta} \qquad \frac{\partial\mathcal{L}}{\partial q} = \frac{\partial\mathcal{L}}{\partial\theta} = -mgL\sin\theta$$

$$\frac{d}{dt}\frac{\partial\mathcal{L}}{\partial\dot{q}} = \frac{\partial\mathcal{L}}{\partial q} \quad \Rightarrow \quad \boxed{\ddot{\theta} = -\frac{g}{L}\sin\theta}$$

Not-So-Simple Example: Double Pendulum

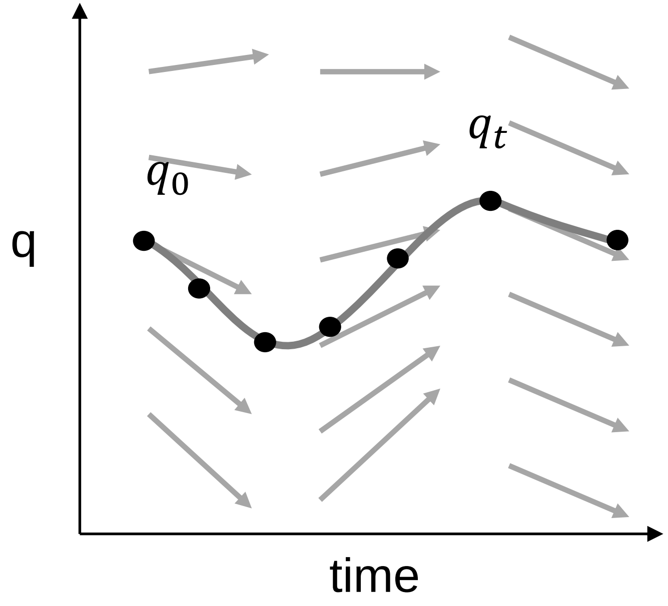
- Two pendulums attached end-to-end
- Simple system, but rather complex motion!
- Chaotic: small changes to input cause large changes to output
- No closed-form solution exists, must use numerical methods



Numerical Solution of Ordinary Differential Equations

Numerical Integration

- *Problem statement*: given initial conditions $q(0)$ and the derivative $\dot{q}(t) = f(t, q)$, compute an approximation to the unknown function $q(t)$
- Replace time-continuous function $q(t)$ with discrete samples q_i at time t_i



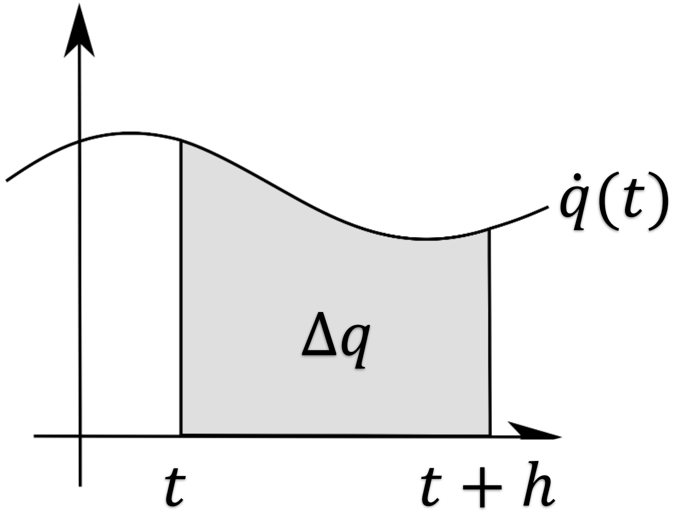
Numerical Integration

- How to compute time-discrete samples q_i?
  → by solving the ODE numerically
- Solving ODEs numerically → numerical time integration

$$q(t+h) = q(t) + \int_{t}^{t+h} \dot{q}(t)\, dt$$

- How do we solve this integral numerically?
  → by using numerical integration schemes

Numerical Integration



Continuous problem:

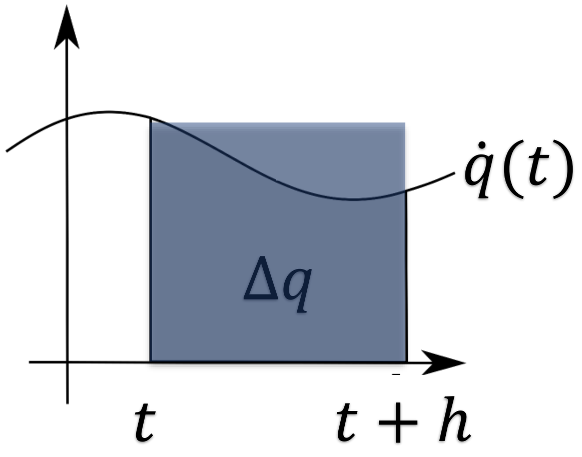
$$q(t+h) = q(t) + \int_t^{t+h} \dot{q}(t)\, dt$$

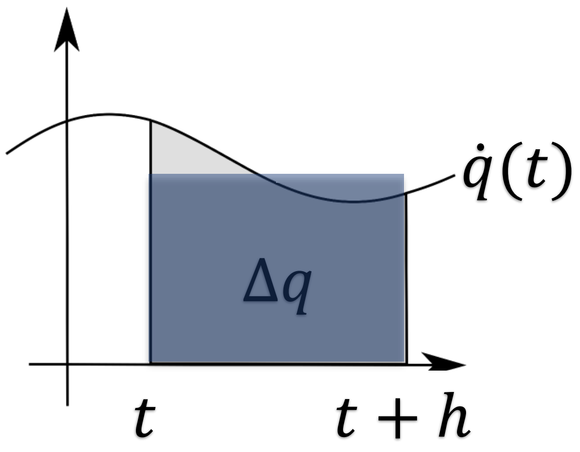
Discrete approximation:

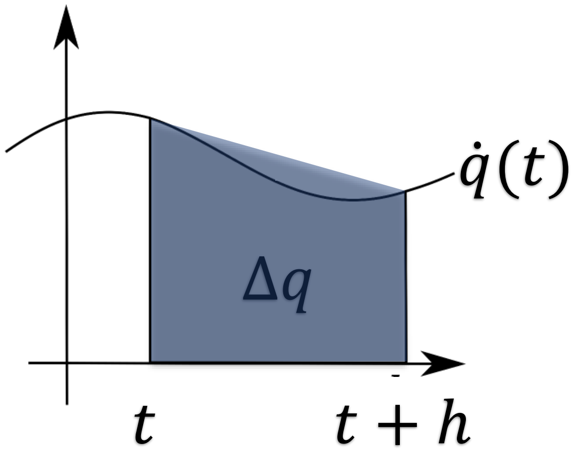
$$q_{i+1} = q_i + \Delta q_i$$

$$\Delta q_i \approx \int_t^{t+h} \dot{q}(t)\, dt$$

Numerical Integration Rules







Rectangle rule

$$\Delta q_i \approx \dot{q}(t) \cdot h$$

Midpoint rule

$$\Delta q_i \approx \dot{q}(t + h/2) \cdot h$$

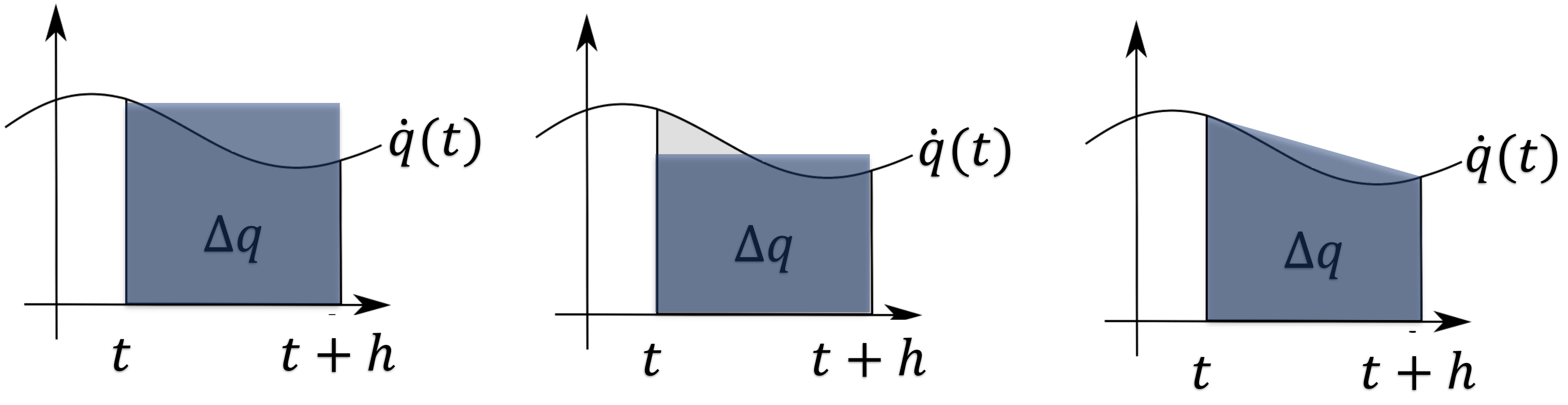
Trapezoid rule

$$\Delta q_i \approx \frac{\dot{q}(t)h + \dot{q}(t+h)h}{2}$$

Configuration update (rectangle rule):

$$q_{i+1} = q_i + h \cdot \dot{q}_i$$

Numerical Integration Rules



Rectangle rule | Midpoint rule | Trapezoid rule

$$\Delta q_i \approx \dot{q}(t) \cdot h$$

$$\Delta q_i \approx \dot{q}(t + h/2) \cdot h$$

$$\Delta q_i \approx \frac{\dot{q}(t)h + \dot{q}(t+h)h}{2}$$

- Integration schemes differ in terms of
  - accuracy/approximation order
  - number/location of function evaluations
  - ...

A Different Point of View

- Taylor series expansion

$$q(t+h) = q(t) + h\frac{dq(t)}{dt} + \frac{h^2}{2}\frac{d^2q(t)}{dt^2} + \cdots = \sum_n^{\infty} \frac{1}{n!}\frac{d^nq(t)}{dt^n}$$

- Truncation yields arbitrary-order approximation

$$q(t+h) = q(t) + h\frac{dq(t)}{dt} + O(h^2)$$

- Gives rise to discrete update rule

$$q_{i+1} = q_i + h \cdot \dot{q}_i \quad *$$

- Also known as *Explicit* or *Forward Euler* scheme

* this expression is identical to the one obtained by applying the *rectangle rule*

Comparing Integration Schemes

How to evaluate an integration scheme?

Criteria

- *Convergence*: do approximations converge to true solution, i.e.,

$$q_i \to q(t_i) \quad \text{as} \quad h \to 0?$$

- *Accuracy*: how fast does the error $|q_i - q(t_i)|$ decrease as $h \to 0$?
- *Stability*: is the solution always bounded, i.e., $|q_i| < \infty$?
- *Efficiency*: is a given method a good choice for a given problem?

Comparing Integration Schemes

Which method is best?

Comparing integration schemes is challenging

- Different costs per step
- Depends strongly on problem
- Not a single *best* method

Approach

- Evaluate on (standard) model problems

$\rightarrow$ Test Equation

Test Equation

Test Equation

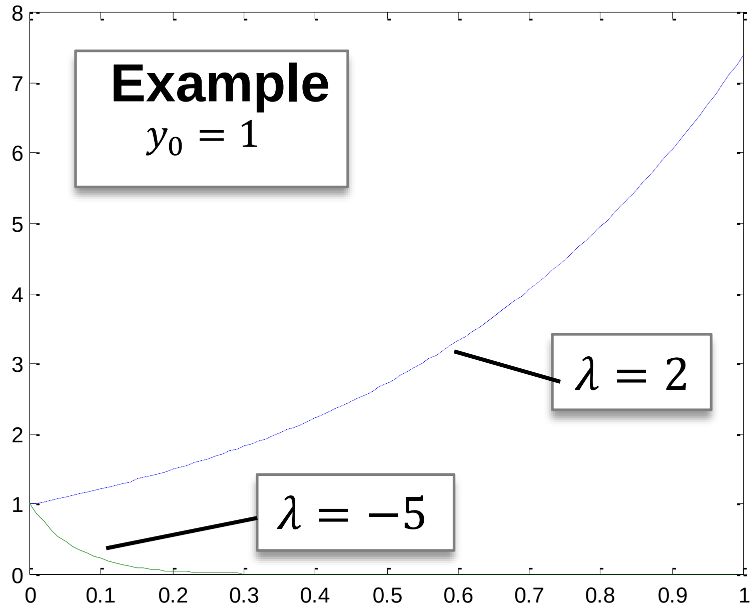
$$y'(t) = \lambda y(t)$$

Analytical Solution

$$y(t) = e^{\lambda t} y_0$$

For $\lambda \in \mathrm{R}$

- $\lambda > 0$ exponential growth
- $\lambda < 0$ exponential decay



Test Equation

Test Equation

$y'(t) = \lambda y(t)$

Analytical Solution

$y(t) = e^{\lambda t} y_0$

For $\lambda \in C$: $\lambda = a + ib$ $\Rightarrow$ $y(t) = y_0 \cdot \underbrace{\boxed{e^{at}}}_{\text{damping}} \cdot \underbrace{\boxed{e^{ibt}}}_{\text{oscillation}}$

- $a < 0$ damped oscillator
- $a = 0$ undamped oscillator
- $a > 0$ unstable

Prototype for mechanical systems

Euler's Formula

$e^{ibt} = \cos(tb) + i \sin(tb)$

Solving the Test Equation Numerically

- Test equation

$$y'(t) = \lambda y(t)$$

- Explicit Euler update rule

$$y_{n+1} = y_n + h\lambda y_n$$

How well does this work in practice?

Let's look at an example.

Explicit Euler Analysis

- Test equation $y'(t) = \lambda y(t)$
- Explicit Euler update rule $y_{n+1} = y_n + h\lambda y_n$
- Solve recursion

$$\begin{aligned} y_{n+1} &= y_n + h\lambda y_n \\ &= (1 + h\lambda)y_n \\ &= (1 + h\lambda)^n\, y_0 \end{aligned}$$

- Stability condition

  For $|(1 + h\lambda)^n\, y_0| < \infty$ we need $|(1 + h\lambda)| < 1$
- In order for EE to remain stable, the step size h must be sufficiently small

CRL

Solving the Test Equation Numerically

Let's try something else…

- Instead of evaluating the derivative at the current location, evaluate it at the next configuration, i.e.,

$$y_{n+1} = y_n + h\lambda y_{n+1}$$

- The update rule is now *implicit*, have to solve for y_{n+1}

$$y_{n+1} = (1 - h\lambda)^{-1} y_n$$

- This update rule is called *implicit* or *backward Euler*

How well does this work in practice?

Implicit Euler Analysis

- Test equation $y'(t) = \lambda y(t)$
- Implicit Euler update rule $y_{n+1} = y_n + h\lambda y_{n+1}$
- Solve recursion

$$\begin{aligned} y_{n+1} &= (1 - h\lambda)^{-1} y_n \\ &= (1 - h\lambda)^{-n} y_0 \end{aligned}$$

- Stability condition

  For $|(1 - h\lambda)^{-n}\, y_0| < \infty$ we need $|(1 - h\lambda)^{-1}| < 1$
- **Implicit Euler is unconditionally stable (for all $h > 0$) for linear ODEs*!**

* Remember that $\lambda < 0$ for damped systems

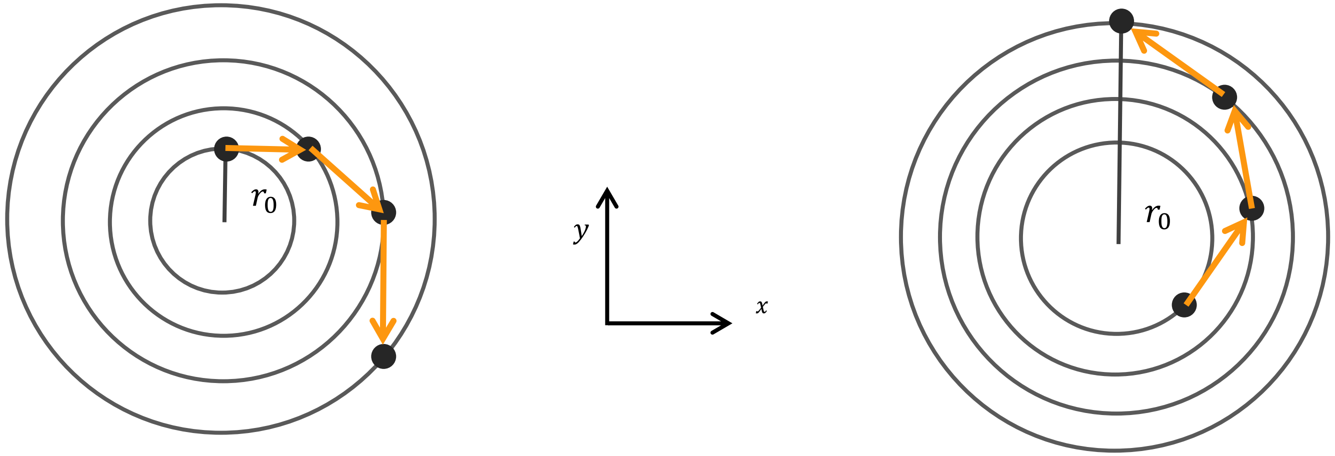
Implicit Euler vs Explicit Euler

- Euler step for test equation $y' = \lambda y, \lambda < 0$
  - Explicit Euler $y_{n+1} = y_n + h\lambda y_n = y_n(1 + h\lambda)$
  - Implicit Euler $y_{n+1} = y_n + h\lambda y_{n+1} = y_n(1 - h\lambda)^{-1}$
- Stability conditions for Euler
  - Explicit $y_n = (1 + h\lambda)^n y_0 < \infty \quad \Leftrightarrow \quad |1 + h\lambda| < 1$
  - Implicit $y_n = (1 - h\lambda)^{-n} y_0 < \infty \quad \Leftrightarrow \quad |1 - h\lambda|^{-1} < 1$

Euler Visually

- Example: 2D circular particle motion, $q(t) = (x(t), y(t))$

Initial condition	State derivative	Analytical solution
$q(0) = \begin{pmatrix} 0 \\ r_0 \end{pmatrix}$	$q'(t) = \begin{pmatrix} y(t) \\ -x(t) \end{pmatrix}$	$q(t) = r_0 \begin{pmatrix} \sin(t) \\ \cos(t) \end{pmatrix}$



Explicit Euler: $q_{n+1} = q_n + hq'(t_n, q_n)$

Solution ($|q|$) unbounded!

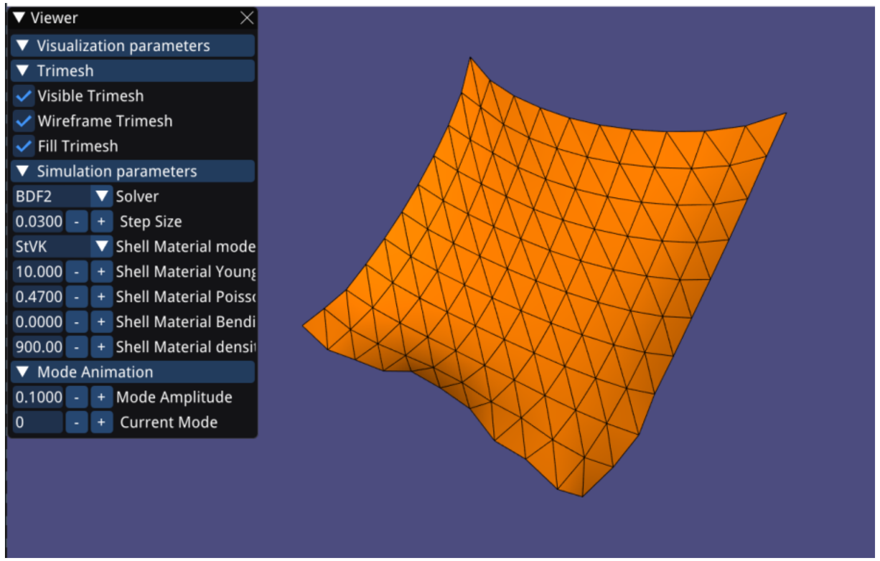
Implicit Euler: $q_{n+1} = q_n + hq'(t_{n+1}, q_{n+1})$

Solution ($|q|$) bounded!

Example: Mass-Spring Systems

Example: Mass-Spring Systems

- A simple model for deformable materials
  - Curves (hair, cables)
  - Sheets (cloth, leather, rubber)
  - Solids (flesh, fat, rubber)



- Application: elastic sheets
  - Discretize sheet as triangle mesh
  - Nodes are mass points with $\boldsymbol{x}_i$ and masses m_i
  - Edges are elastic springs exerting forces when deformed
  - Goal: simulate motion under elastic forces and gravity

Cloth and Hair Simulation for Computer Animation

Disney's *Tangled*

Mass-Spring Systems

- Connect two particles $\boldsymbol{x}_1, \boldsymbol{x}_2$ by a spring of length L_0
- Spring force is given by Hooke's law:

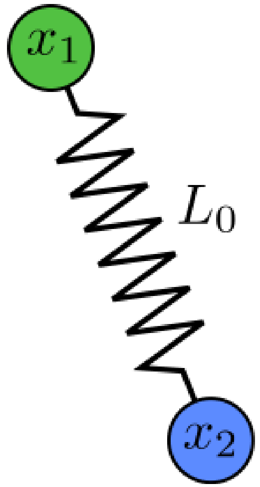
current length

direction

stiffness

$$\boldsymbol{f}_{12} = -k\left(\frac{|\boldsymbol{x}_1 - \boldsymbol{x}_2|}{L_0} - 1\right)\frac{\boldsymbol{x}_1 - \boldsymbol{x}_2}{|\boldsymbol{x}_1 - \boldsymbol{x}_2|}$$

rest length



- Easy to understand, simple to implement
- Limited control of material behavior

Mass-Spring Systems – Implicit Time Integration

- For each node we have

  $m\boldsymbol{a}_i = \boldsymbol{f}_i$ where $\boldsymbol{f}_i = \sum_j \boldsymbol{f}_{ij}^{spring} + m\boldsymbol{g}$

- Convert to system of first-order ODEs
  - $\dot{\boldsymbol{x}} = \boldsymbol{v}$
  - $\dot{\boldsymbol{v}} = \boldsymbol{M}^{-1}\boldsymbol{f}(\boldsymbol{x})$

- Discretize with implicit Euler
  - $\boldsymbol{x}_{n+1} = \boldsymbol{x}_n + h\boldsymbol{v}_{n+1}$
  - $\boldsymbol{v}_{n+1} = \boldsymbol{v}_n + h\boldsymbol{a}_{n+1} = \boldsymbol{v}_n + h\boldsymbol{M}^{-1}\boldsymbol{f}(\boldsymbol{x}_{n+1})$

Implicit Euler

- Implicit Euler update rules
  - $\boldsymbol{x}_{n+1} = \boldsymbol{x}_n + h\boldsymbol{v}_{n+1}$
  - $\boldsymbol{v}_{n+1} = \boldsymbol{v}_n + h\boldsymbol{a}_{n+1} = \boldsymbol{v}_n + h\boldsymbol{M}^{-1}\boldsymbol{f}(\boldsymbol{x}_{n+1})$

- Rewrite with positions as only unknowns

$$\boldsymbol{x}_{n+1} = \boldsymbol{x}_n + h\boldsymbol{v}_{n+1} = \boldsymbol{x}_n + h\,\boldsymbol{v}_n + h^2\boldsymbol{M}^{-1}\boldsymbol{f}(\boldsymbol{x}_{n+1})$$
$$\boldsymbol{g}(\boldsymbol{x}_{n+1}) = \boldsymbol{M}(\boldsymbol{x}_{n+1} - \boldsymbol{x}_n) - h^2\boldsymbol{f}(\boldsymbol{x}_{n+1}) - h\boldsymbol{M}\boldsymbol{v}_n = 0$$

- Solve $\boldsymbol{g}(\boldsymbol{x}_{n+1}) = \boldsymbol{0}$ with Newton’s method

CRL

Newton's Method

Solve $\mathbf{g}(\mathbf{x}_{n+1}) = \mathbf{0}$

Make a guess

$$\mathbf{x}_{n+1} = \mathbf{x}_n + h\mathbf{v}_n$$

Generally

$$\mathbf{g}(\mathbf{x}_{n+1}) \neq \mathbf{0}$$

Correction

$$\mathbf{g}(\mathbf{x}_{n+1} + \Delta\mathbf{x}) = \mathbf{0}$$

Taylor Expansion

$$\mathbf{g}(\mathbf{x}_{n+1}) + \frac{\partial \mathbf{g}}{\partial \mathbf{x}}\Delta\mathbf{x} + O(\Delta\mathbf{x}^2) = \mathbf{0}$$

Linearize

$$\frac{\partial \mathbf{g}}{\partial \mathbf{x}}\Delta\mathbf{x} + \mathbf{g}(\mathbf{x}_{n+1}) \approx \mathbf{0}$$

Solve for $\Delta\mathbf{x}$ → **Update** → **Compute error**

$$\frac{\partial \mathbf{g}}{\partial \mathbf{x}}\Delta\mathbf{x} = -\mathbf{g}(\mathbf{x}_{n+1})$$

$$\mathbf{x}_{n+1} \mathrel{+}= \Delta\mathbf{x}$$

$$g(\mathbf{x}_{n+1})$$

Repeat until $|\mathbf{g}|$ small enough

Newton's Method – Remarks

- Have to solve linear system in each iteration
  - *How do we do that?*
  - Off-the-shelf linear solver depending on matrix type (sparse, symmetric, …)
- Requires derivatives of the forces
  - *How do we compute those?*
  - Manually (easy for springs) or with symbolic differentiation

- More details in lecture on continuum mechanics and finite elements

Further Reading