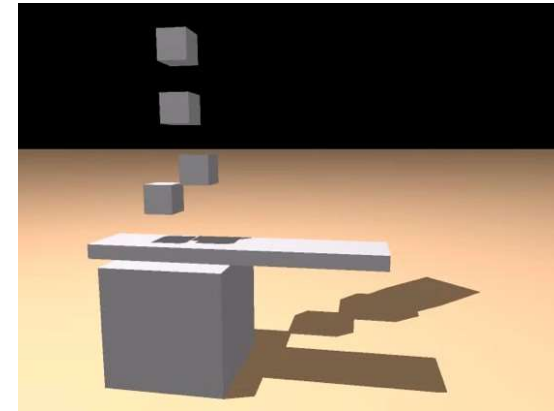
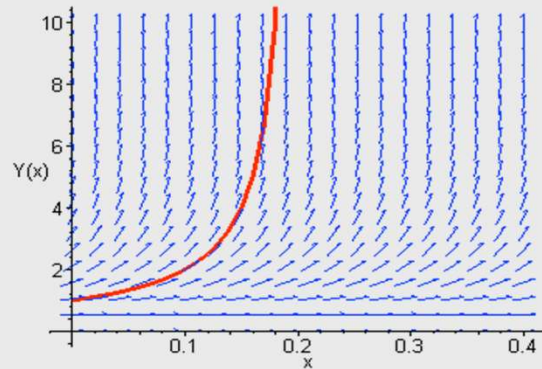


## Differential Equations

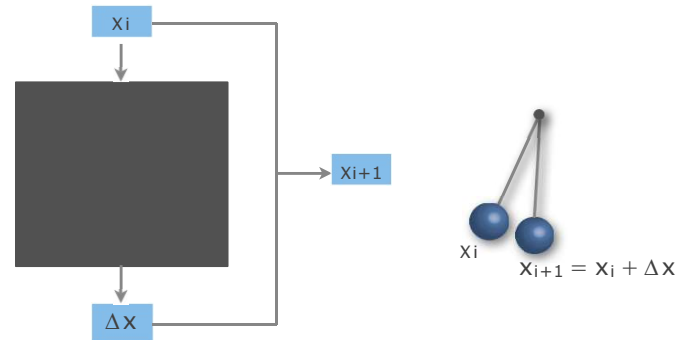


- Overview of differential equation
- Initial value problem
- Explicit numeric methods
- Implicit numeric methods
- Modular implementation

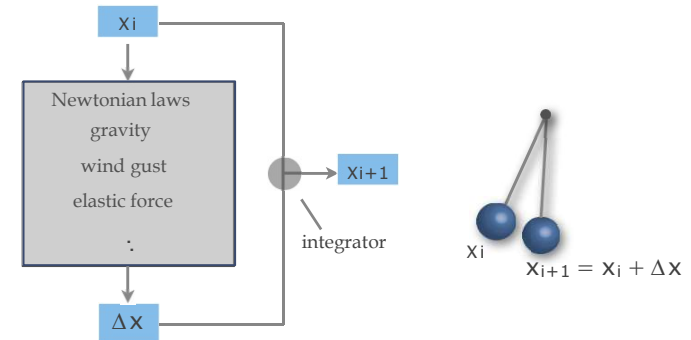
## Physics-based simulation

- It's an algorithm that produces a sequence of states over time under the laws of physics
- What is a state?

## Physics-based simulation



## Physics-based simulation



## Differential equations

- What is a differential equation?
- It describes the relation between an unknown function and its derivatives
- Ordinary differential equation (ODE)
  - is the relation that contains functions of only one independent variable and its derivatives

## Ordinary differential equations

An ODE is an equality equation involving a function and its derivatives

$$\dot{x}(t) = f(x(t))$$

known function  $\downarrow$   
 $\uparrow$  time derivative of the unknown function       $\uparrow$  unknown function that evaluates the state given time

What does it mean to “solve” an ODE?

## Symbolic solutions

- Standard introductory differential equation courses focus on finding solutions analytically
- Linear ODEs can be solved by integral transforms
- Use `DSolve[eqn, x, t]` in Mathematica

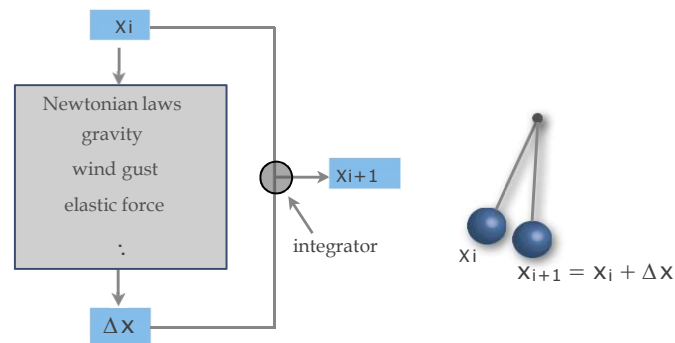
Differential equation:  $\dot{x}(t) = -kx(t)$

Solution:  $x(t) = e^{-kt}$

## Numerical solutions

- In this class, we will be concerned with numerical solutions
- Derivative function  $f$  is regarded as a black box
- Given a numerical value  $x$  and  $t$ , the black box will return the time derivative of  $x$

## Physics-based simulation



- Overview of differential equation
- Initial value problem
- Explicit numeric methods
- Implicit numeric methods
- Modular implementation

## Initial value problems

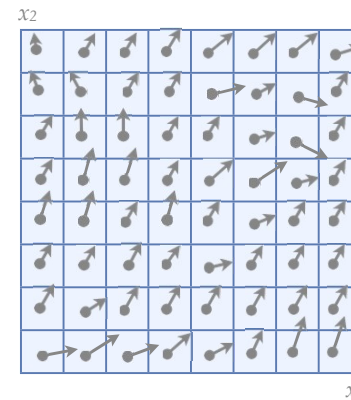
In a canonical initial value problem, the behavior of the system is described by an ODE and its initial condition:

$$\dot{x} = f(x, t)$$

$$x(t_0) = x_0$$

To solve  $x(t)$  numerically, we start out from  $x_0$  and follow the changes defined by  $f$  thereafter

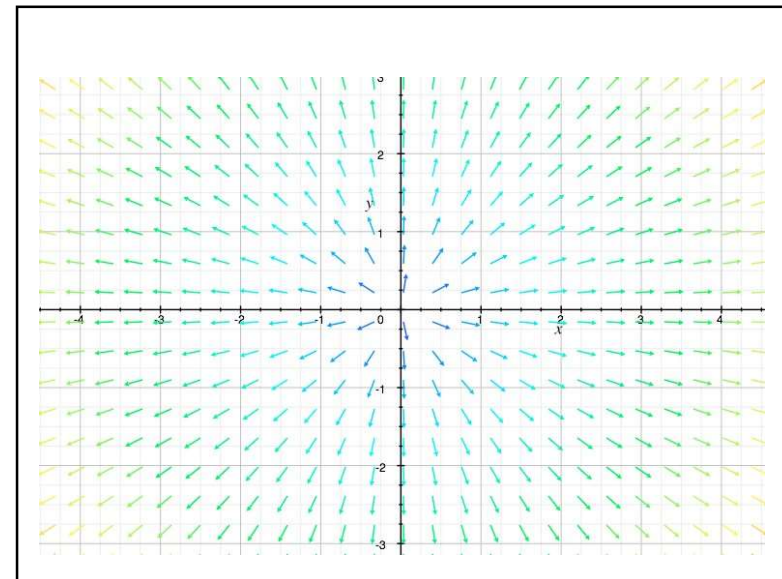
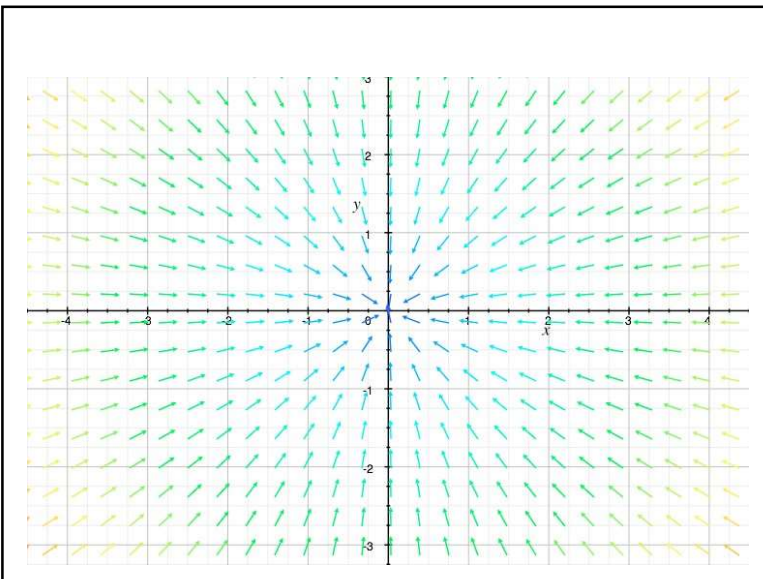
## Vector field



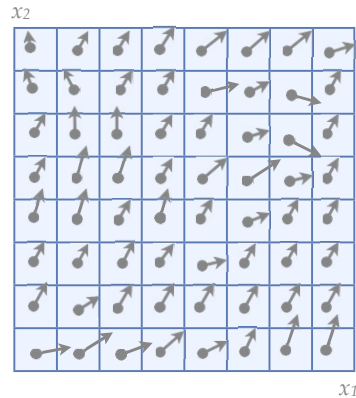
The differential equation can be visualized as a vector field

$$\dot{x} = f(x, t)$$

$x(t)$  : a moving point  
 $f(x, t)$  :  $x$ 's velocity



## Vector field

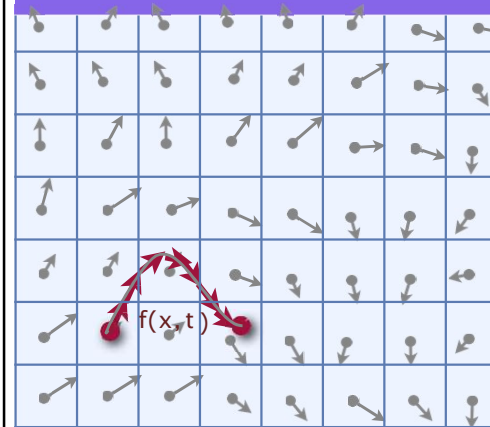


The differential equation can be visualized as a vector field

$$\dot{\mathbf{x}} = f(\mathbf{x}, t)$$

How does the vector field look like if  $f$  depends directly on time?

## Integral curves

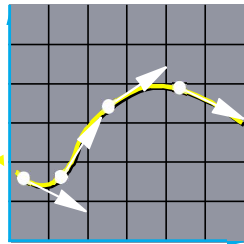


$$\int_{t_0} f(\mathbf{x}, t) dt$$

## Integral Curves

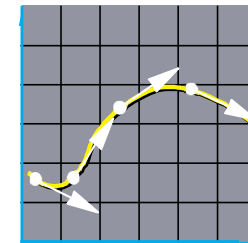
Start Here

Pick any starting point, and follow the vectors.



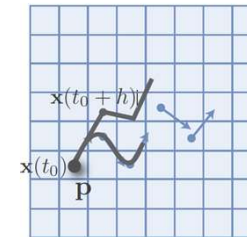
## Initial Value Problem

Given the starting point, follow the integral curve.



- Overview of differential equation
- Initial value problem
- Explicit numeric methods
- Implicit numeric methods
- Modular implementation

## Explicit Euler method



How do we get to the next state from the current state?

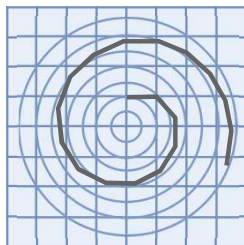
$$\mathbf{x}(t_0 + h) = \mathbf{x}_0 + h\dot{\mathbf{x}}(t_0)$$

Instead of following real integral curve,  $\mathbf{p}$  follows a polygonal path

Discrete time step  $h$  determines the errors

## Problems of Euler method

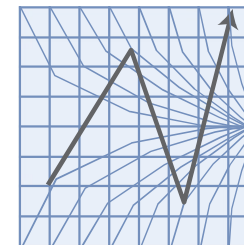
Inaccuracy



The circle turns into a spiral no matter how small the step size is

## Problems of Euler method

Instability



$$\dot{\mathbf{x}} = -k\mathbf{x}$$

Symbolic solution:  $\mathbf{x}(t) = e^{-kt}$

Oscillation:

Divergence:

How small the step size has to be?

## Accuracy of Euler method

- At each step,  $\mathbf{x}(t)$  can be written in the form of Taylor series:
- $$\mathbf{x}(t_0 + h) = \mathbf{x}(t_0) + h\dot{\mathbf{x}}(t_0) + \frac{h^2}{2!}\ddot{\mathbf{x}}(t_0) + \frac{h^3}{3!}\dddot{\mathbf{x}}(t_0) + \dots + \frac{h^n}{n!}\frac{\partial^n \mathbf{x}}{\partial t^n} + \dots$$
- What is the order of the error term in Euler method?
  - The cost per step is determined by the number of evaluations per step

Taylor series is a representation of a function as an infinite sum of terms calculated using the derivatives at a particular point

## Stability of Euler method

- Assume the derivative function is linear

$$\frac{d}{dt}\mathbf{x} = \mathbf{A}\mathbf{x}$$

- Look at  $\mathbf{x}$  parallel to the largest eigenvector of  $\mathbf{A}$

$$\frac{d}{dt}\mathbf{x} = \lambda\mathbf{x}$$

- Note that eigenvalue  $\lambda$  can be complex

## The test equation

- Test equation advances  $\mathbf{x}$  by

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

- Solving gives

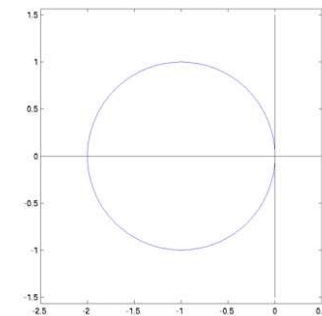
$$\mathbf{x}_n = (1 + h\lambda)^n \mathbf{x}_0$$

- Condition of stability

$$|1 + h\lambda| \leq 1$$

## Stability region

- Plot all the values of  $h\lambda$  on the complex plane where Euler method is stable



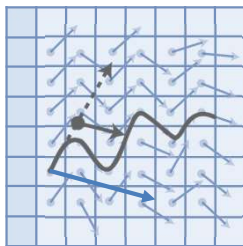
## Real eigenvalue

- If eigenvalue is real and negative, what kind of the motion does  $\mathbf{x}$  correspond to?
  - a damping motion smoothly coming to a halt
- The threshold of time step
 
$$h \leq \frac{2}{|\lambda|}$$
- What about the imaginary axis?

## Imaginary eigenvalue

- If eigenvalue is pure imaginary, Euler method is unconditionally unstable
- What motion does  $\mathbf{x}$  look like if the eigenvalue is pure imaginary?
  - an oscillatory or circular motion
- We need to look at other methods

## The midpoint method



1. Compute an Euler step

$$\Delta \mathbf{x} = h \mathbf{f}(\mathbf{x}(t_0))$$

2. Evaluate  $\mathbf{f}$  at the midpoint

$$\mathbf{f}_{mid} = \mathbf{f}(\mathbf{x}(t_0) + \frac{\Delta \mathbf{x}}{2})$$

3. Take a step using  $\mathbf{f}_{mid}$

$$\mathbf{x}(t_0 + h) = \mathbf{x}(t_0) + h \mathbf{f}_{mid}$$

$$\mathbf{x}(t + h) = \mathbf{x}_0 + h \mathbf{f}(\mathbf{x}_0 + \frac{h}{2} \mathbf{f}(\mathbf{x}_0))$$

## Accuracy of midpoint

Prove that the midpoint method is correct within  $O(h^3)$

$$\mathbf{x}(t + h) = \mathbf{x}_0 + h \mathbf{f}(\mathbf{x}_0) + \frac{h^2}{2} \mathbf{f}'(\mathbf{x}_0)$$

$$\Delta \mathbf{x} = \frac{h}{2} \mathbf{f}(\mathbf{x}_0)$$

$$\mathbf{f}(\mathbf{x}_0 + \Delta \mathbf{x}) = \mathbf{f}(\mathbf{x}_0) + \Delta \mathbf{x} \frac{\partial \mathbf{f}(\mathbf{x}_0)}{\partial \mathbf{x}} + O(h^2)$$

$$\mathbf{x}(t + h) = \mathbf{x}_0 + h \mathbf{f}(\mathbf{x}_0) + \frac{h^2}{2} \mathbf{f}(\mathbf{x}_0) \frac{\partial \mathbf{f}(\mathbf{x}_0)}{\partial \mathbf{x}} + h O(h^2)$$

$$\mathbf{x}(t + h) = \mathbf{x}_0 + h \dot{\mathbf{x}}_0 + \frac{h^2}{2} \ddot{\mathbf{x}}_0 + O(h^3)$$



## Stability region

$$\mathbf{x}_{n+1} = \mathbf{x}_n + h\lambda\mathbf{x}_{n+\frac{1}{2}} = \mathbf{x}_n + h\lambda(\mathbf{x}_n + \frac{1}{2}h\lambda\mathbf{x}_n)$$

$$\mathbf{x}_{n+1} = \mathbf{x}_n(1 + h\lambda + \frac{1}{2}(h\lambda)^2)$$

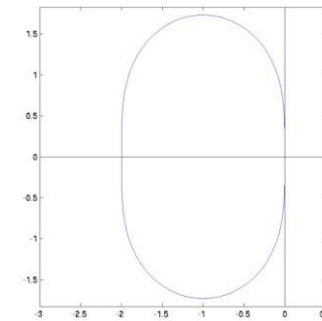
$$h\lambda = x + iy$$

$$\left| \begin{bmatrix} 1 \\ 0 \end{bmatrix} + \begin{bmatrix} x \\ y \end{bmatrix} + \frac{1}{2} \begin{bmatrix} x^2 - y^2 \\ 2xy \end{bmatrix} \right| \leq 1$$

$$\left| \begin{bmatrix} 1 + x + \frac{x^2 - y^2}{2} \\ y + xy \end{bmatrix} \right| \leq 1$$

## Stability of midpoint

- Midpoint method has larger stability region, but still unstable on the imaginary axis



## Runge-Kutta method

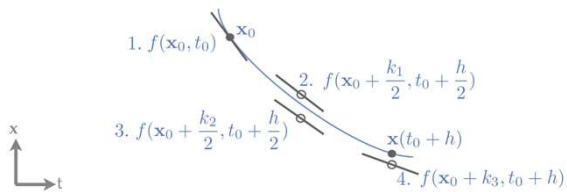
- Runge-Kutta is a numeric method of integrating ODEs by evaluating the derivatives at a few locations to cancel out lower-order error terms
- Also an explicit method:  $\mathbf{x}_{n+1}$  is an explicit function of  $\mathbf{x}_n$

## Runge-Kutta method

- $q$ -stage  $p$ -order Runge-Kutta evaluates the derivative function  $q$  times in each iteration and its approximation of the next state is correct within  $O(h^{p+1})$
- What order of Runge-Kutta does midpoint method correspond to?

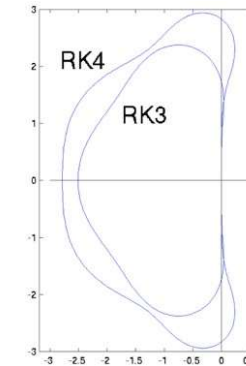
## 4-stage 4<sup>th</sup> order Runge-Kutta

$$\begin{aligned}
 k_1 &= hf(\mathbf{x}_0, t_0) \\
 k_2 &= hf(\mathbf{x}_0 + \frac{k_1}{2}, t_0 + \frac{h}{2}) \\
 k_3 &= hf(\mathbf{x}_0 + \frac{k_2}{2}, t_0 + \frac{h}{2}) \\
 k_4 &= hf(\mathbf{x}_0 + k_3, t_0 + h) \\
 \mathbf{x}(t_0 + h) &= \mathbf{x}_0 + \frac{1}{6}k_1 + \frac{1}{3}k_2 + \frac{1}{3}k_3 + \frac{1}{6}k_4
 \end{aligned}$$



## High order Runge-Kutta

- RK3 and up include part of the imaginary axis



## Stage vs. order

$p$	1	2	3	4	5	6	7	8	9	10
$q_{min}(p)$	1	2	3	4	6	7	9	11	12-17	13-17

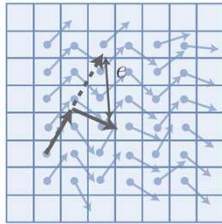
The minimum number of stages necessary for an explicit method to attain order  $p$  is still an open problem

Why is fourth order the most popular Runge Kutta method?

## Adaptive step size

- Ideally, we want to choose  $h$  as large as possible, but not so large as to give us big error or instability
- We can vary  $h$  as we march forward in time
  - Step doubling
  - Embedding estimate
  - Variable step, variable order

## Step doubling



Estimate  $\mathbf{x}_a$  by taking a full Euler step

$$\mathbf{x}_a = \mathbf{x}_0 + hf(\mathbf{x}_0, t_0)$$

Estimate  $\mathbf{x}_b$  by taking two half Euler steps

$$\mathbf{x}_{temp} = \mathbf{x}_0 + \frac{h}{2}f(\mathbf{x}_0, t_0)$$

$$\mathbf{x}_b = \mathbf{x}_{temp} + \frac{h}{2}f(\mathbf{x}_{temp}, t_0 + \frac{h}{2})$$

$$e = |\mathbf{x}_a - \mathbf{x}_b| \text{ is bound by } O(h^2)$$

Given error tolerance  $\epsilon$ , what is the optimal step size?  $\left(\frac{\epsilon}{e}\right)^{\frac{1}{2}} h$

## Embedding estimate

- Also called Runge-Kutta-Fehlberg
- Compare two estimates of  $\mathbf{x}(t_0 + h)$ 
  - Fifth order Runge-Kutta with 6 stages
  - Forth order Runge-Kutta with 6 stages

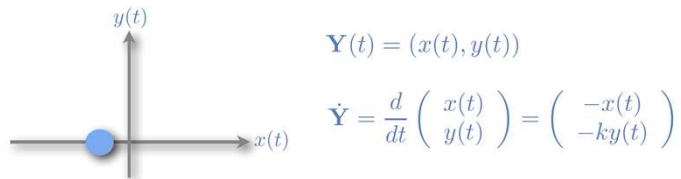
## Variable step, variable order

- Change between methods of different order as well as step based on obtained error estimates
- These methods are currently the last work in numerical integration

## Problems of explicit methods

- Do not work well with stiff ODEs
  - Simulation blows up if the step size is too big
  - Simulation progresses slowly if the step size is too small

## Example: a bead on the wire



Explicit Euler's method:

$$\mathbf{Y}_{new} = \mathbf{Y}_0 + h\dot{\mathbf{Y}}(t_0) = \begin{pmatrix} x(t) \\ y(t) \end{pmatrix} + h \begin{pmatrix} -x(t) \\ -ky(t) \end{pmatrix}$$

$$\mathbf{Y}_{new} = \begin{pmatrix} (1-h)x(t) \\ (1-kh)y(t) \end{pmatrix} \quad h \leq \frac{2}{|\lambda|}$$

## Stiff equations

- Stiffness constant:  $k$
- Step size is limited by the largest  $k$
- Systems that has some big  $k$ 's mixed in are called "**stiff system**"

- Overview of differential equation
- Initial value problem
- Explicit numeric methods
- Implicit numeric methods
- Modular implementation

## Implicit methods

Explicit Euler:  $\mathbf{Y}_{new} = \mathbf{Y}_0 + hf(\mathbf{Y}_0)$

Implicit Euler:  $\mathbf{Y}_{new} = \mathbf{Y}_0 + hf(\mathbf{Y}_{new})$

Solving for  $\mathbf{Y}_{new}$  such that  $\mathbf{f}$ , at time  $t_0 + h$ , points directly back at  $\mathbf{Y}_0$

## Implicit methods

Our goal is to solve for  $\mathbf{Y}_{new}$  such that

$$\mathbf{Y}_{new} = \mathbf{Y}_0 + hf(\mathbf{Y}_{new})$$

Approximating  $f(\mathbf{Y}_{new})$  by linearizing  $f(\mathbf{Y})$

$$f(\mathbf{Y}_{new}) = f(\mathbf{Y}_0) + \Delta \mathbf{Y} f'(\mathbf{Y}_0), \text{ where } \Delta \mathbf{Y} = \mathbf{Y}_{new} - \mathbf{Y}_0$$

$$\mathbf{Y}_{new} = \mathbf{Y}_0 + hf(\mathbf{Y}_0) + h\Delta \mathbf{Y} f'(\mathbf{Y}_0)$$

$$\Delta \mathbf{Y} = \left( \frac{1}{h} \mathbf{I} - f'(\mathbf{Y}_0) \right)^{-1} f(\mathbf{Y}_0)$$

$$\begin{aligned} f(\mathbf{Y}, t) &= \dot{\mathbf{Y}}(t) \\ f(\mathbf{Y}, t)' &= \frac{\partial f}{\partial \mathbf{Y}} \end{aligned}$$

## Example: A bead on the wire

Apply the implicit Euler method to the bead-on-wire example

$$\Delta \mathbf{Y} = \left( \frac{1}{h} \mathbf{I} - f'(\mathbf{Y}_0) \right)^{-1} f(\mathbf{Y}_0)$$

$$f(\mathbf{Y}(t)) = \begin{bmatrix} -x(t) \\ -ky(t) \end{bmatrix}$$

$$f'(\mathbf{Y}(t)) = \frac{\partial f(\mathbf{Y}(t))}{\partial \mathbf{Y}} = \begin{bmatrix} -1 & 0 \\ 0 & -k \end{bmatrix}$$

$$\begin{aligned} \Delta \mathbf{Y} &= \begin{bmatrix} \frac{1+h}{h} & 0 \\ 0 & \frac{1+kh}{h} \end{bmatrix}^{-1} \begin{bmatrix} -x_0 \\ -ky_0 \end{bmatrix} \\ &= \begin{bmatrix} \frac{h}{h+1} & 0 \\ 0 & \frac{h}{1+kh} \end{bmatrix} \begin{bmatrix} -x_0 \\ -ky_0 \end{bmatrix} \\ &= - \begin{bmatrix} \frac{h}{h+1} x_0 \\ \frac{h}{1+kh} ky_0 \end{bmatrix} \end{aligned}$$

## Example: A bead on the wire

What is the largest step size the implicit Euler method can take?

$$\begin{aligned} \lim_{h \rightarrow \infty} \Delta \mathbf{Y} &= \lim_{h \rightarrow \infty} - \begin{bmatrix} \frac{h}{h+1} x_0 \\ \frac{h}{1+kh} ky_0 \end{bmatrix} \\ &= - \begin{bmatrix} x_0 \\ \frac{1}{k} ky_0 \end{bmatrix} = - \begin{bmatrix} x_0 \\ y_0 \end{bmatrix} \end{aligned}$$

$$\mathbf{Y}_{new} = \mathbf{Y}_0 + (-\mathbf{Y}_0) = \mathbf{0}$$

## Stability of implicit Euler

- Test equation shows stable when

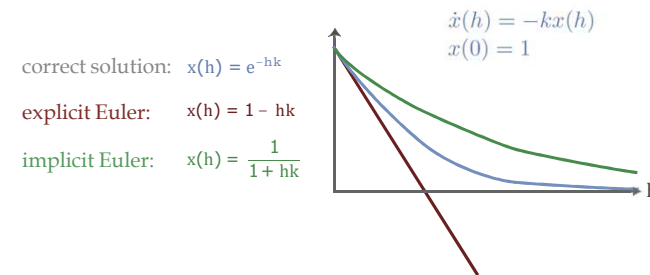
$$|1 - h\lambda| \geq 1$$

- How does the stability region look like?

## Problems of implicit Euler

- Implicit Euler could be stable even when physics is not!
- Implicit Euler damps out motion unrealistically

## Implicit vs. explicit



## Trapezoidal rule

- Take a half step of explicit Euler and a half step of implicit Euler

$$\mathbf{x}_{n+1} = \mathbf{x}_n + h\left(\frac{1}{2}f(\mathbf{x}_n) + \frac{1}{2}f(\mathbf{x}_{n+1})\right)$$

- Explicit Euler is under-stable, implicit Euler is over-stable, the combination is just right

## Stability of Trapezoidal

- What is the test equation for Trapezoidal?

$$h\lambda \leq 0$$

- Where is the stability region?
  - negative half-plane
- Stability region is consistent with physics
- Good for pure rotation

## Terminology

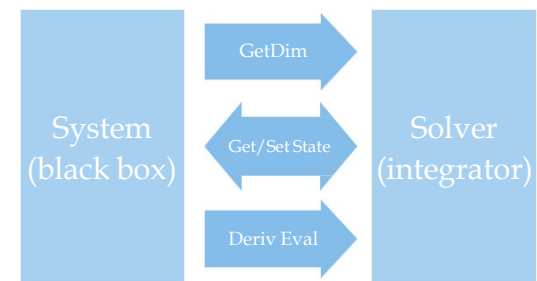
- Explicit Euler is also called **forward Euler**
- Implicit Euler is also called **backward Euler**

- Overview of differential equation
- Initial value problem
- Explicit numeric methods
- Implicit numeric methods
- Modular implementation

## Modular implementation

- Write integrator in terms of
  - Reusable code
  - Simple system implementation
- Generic operations:
  - Get dim(x)
  - Get/Set x and t
  - Derivative evaluation at current (x, t)

## Solver interface



## Summary

- Explicit Euler is simple, but might not be stable; modified Euler may be a cheap alternative
- RK4 allows for larger time step, but requires much more computation
- Use implicit Euler for better stability, but beware of over-damp