Ordinary Differential Equations I

CS 205A:
Mathematical Methods for Robotics, Vision, and Graphics

Doug James (and Justin Solomon)

Announcements

Last Homework (HW7):

- Due Tuesday 6/6 (before midnight)
- NOTE: No late HW7 submissions accepted.

Final Exam:

- Sat June 10 @ 3:30-6:30pm in **(Room: NVIDIA Auditorium)**
- Covers all material–emphasis on 2^{nd} half
- Two cheatsheets–can reuse one from midterm
- Similar format to midterm
- OAE: Tentatively: 2:00pm start (pls confirm time)
- SCPD: Arrange similar time (or slightly later date)

Theme of Last Few Weeks

The unknown is an entire function f.

New Twist

So far:

f (or its derivative/integral) known at isolated points

New Twist

So far:

f (or its derivative/integral) known at isolated points

Instead:

Optimize *properties* of f

Example Problems

- Approximate f_0 with f but make it smoother (or sharper!)

Example Problems

- Approximate f_0 with f but make it smoother (or sharper!)
- Simulate some dynamical or physical relationship as $f(t)$ where t is time

Example Problems

- Approximate f_0 with f but make it smoother (or sharper!)
- Simulate some dynamical or physical relationship as $f(t)$ where t is time
- Approximate f_0 with f but transfer properties of g_0

Today: Initial Value Problems

$$\text{Find } f(t) : \mathbb{R} \to \mathbb{R}^n$$

$$\text{Satisfying } F[t, f(t), f'(t), f''(t), \ldots, f^{(k)}(t)] = 0$$

$$\text{Given } f(0), f'(0), f''(0), \ldots, f^{(k-1)}(0)$$

Today: Initial Value Problems

$$\text{Find } f(t) : \mathbb{R} \to \mathbb{R}^n$$

$$\text{Satisfying } F[t, f(t), f'(t), f''(t), \ldots, f^{(k)}(t)] = 0$$

$$\text{Given } f(0), f'(0), f''(0), \ldots, f^{(k-1)}(0)$$

Example of canonical form (on board):

$$y'' = ty' \cos y$$

Most Famous Example

$$\vec{F} = m\vec{a}$$

Newton's second law

Most Famous Example

$$\vec{F} = m\vec{a}$$

Newton's second law

$\vec{F}(t, \vec{x}, \vec{x}')$ usual expression of force

Most Famous Example

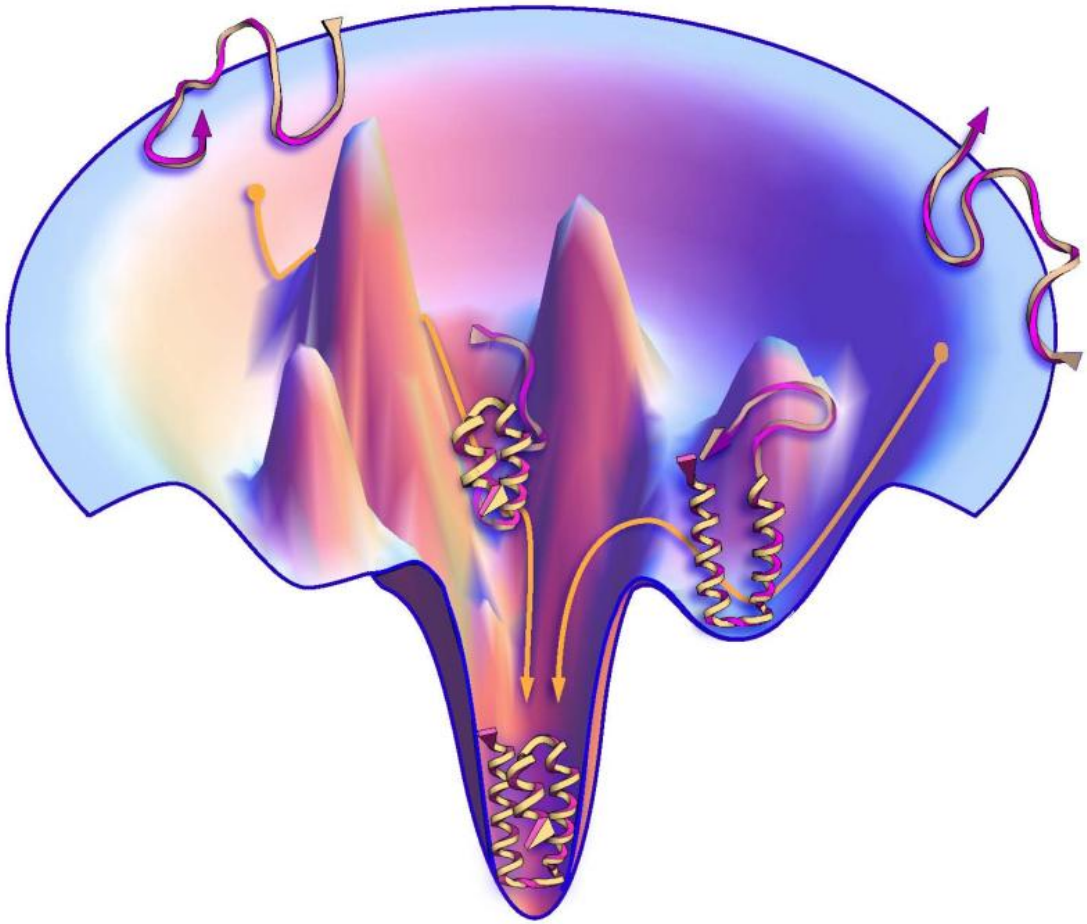
$$\vec{F} = m\vec{a}$$

Newton's second law

$\vec{F}(t, \vec{x}, \vec{x}')$ usual expression of force

n particles $\implies$ simulation in $\mathbb{R}^{3n}$

Protein Folding



http://www.sciencedaily.com/releases/2012/11/121122152910.htm

Gradient Descent

$$\min_{\vec{x}} E(\vec{x})$$

Gradient Descent

$$\min_{\vec{x}} E(\vec{x})$$

$$\implies \vec{x}_{i+1} = \vec{x}_i - h\nabla E(\vec{x}_i)$$

Gradient Descent

$$\min_{\vec{x}} E(\vec{x})$$

$$\implies \vec{x}_{i+1} = \vec{x}_i - h\nabla E(\vec{x}_i)$$

$$\underset{h\to 0}{\implies} \frac{d\vec{x}}{dt} = -\nabla E(\vec{x})$$

Crowd Simulation

http://video.wired.com/watch/building-a-better-zombie-wwz-exclusive

http://gamma.cs.unc.edu/DenseCrowds/

Examples of ODEs

- $y' = 1 + \cos t$: solved by integrating both sides

Examples of ODEs

- $y' = 1 + \cos t$: solved by integrating both sides
- $y' = ay$: linear in y, no dependence on t

Examples of ODEs

- $y' = 1 + \cos t$: solved by integrating both sides
- $y' = ay$: linear in y, no dependence on t
- $y' = ay + e^t$: time and value-dependent

Examples of ODEs

- $y' = 1 + \cos t$: solved by integrating both sides
- $y' = ay$: linear in y, no dependence on t
- $y' = ay + e^t$: time and value-dependent
- $y'' + 3y' - y = t$: multiple derivatives of y

Examples of ODEs

- $y' = 1 + \cos t$: solved by integrating both sides
- $y' = ay$: linear in y, no dependence on t
- $y' = ay + e^t$: time and value-dependent
- $y'' + 3y' - y = t$: multiple derivatives of y
- $y'' \sin y = e^{ty'}$: nonlinear in y and t

Reasonable Assumption

Explicit ODE

An ODE is *explicit* if can be written in the form

$$f^{(k)}(t) = F[t, f(t), f'(t), f''(t), \ldots, f^{(k-1)}(t)].$$

Reasonable Assumption

Explicit ODE

An ODE is *explicit* if can be written in the form

$$f^{(k)}(t) = F[t, f(t), f'(t), f''(t), \ldots, f^{(k-1)}(t)].$$

Otherwise need to do root-finding!

Reduction to First Order

$$f^{(k)}(t) = F[t, f(t), f'(t), f''(t), \ldots, f^{(k-1)}(t)]$$

Reduction to First Order

$$f^{(k)}(t) = F[t, f(t), f'(t), f''(t), \ldots, f^{(k-1)}(t)]$$

$$\frac{d}{dt}\begin{pmatrix} f_0(t) \\ f_1(t) \\ \vdots \\ f_{k-2}(t) \\ f_{k-1}(t) \end{pmatrix} = \begin{pmatrix} f_1(t) \\ f_2(t) \\ \vdots \\ f_{k-1}(t) \\ F[t, f_0(t), f_1(t), \ldots, f_{k-1}(t)] \end{pmatrix}$$

Example

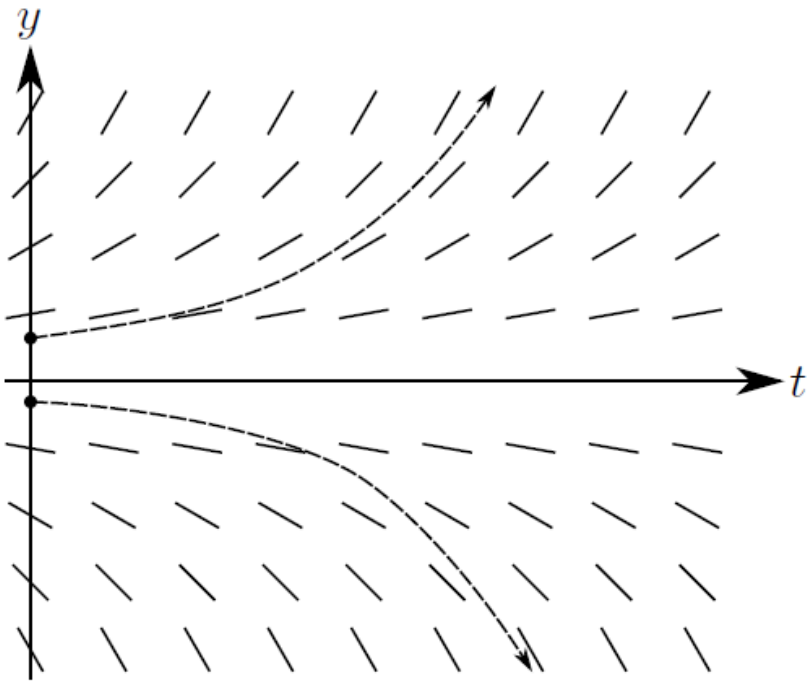
$$y''' = 3y'' - 2y' + y$$

Example

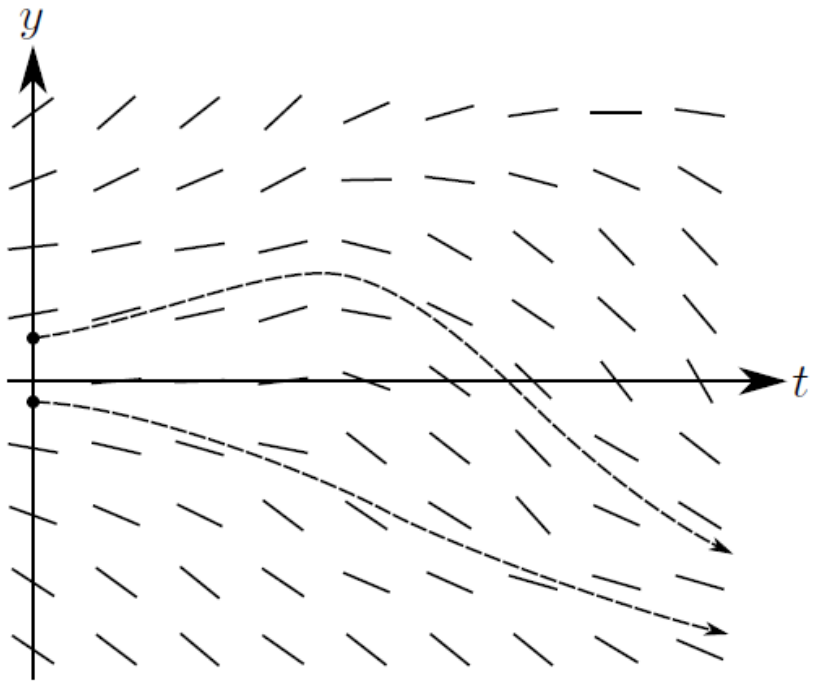
$$y''' = 3y'' - 2y' + y$$

$$\frac{d}{dt}\begin{pmatrix} y \\ z \\ w \end{pmatrix} = \begin{pmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \\ 1 & -2 & 3 \end{pmatrix}\begin{pmatrix} y \\ z \\ w \end{pmatrix}$$

Time Dependence

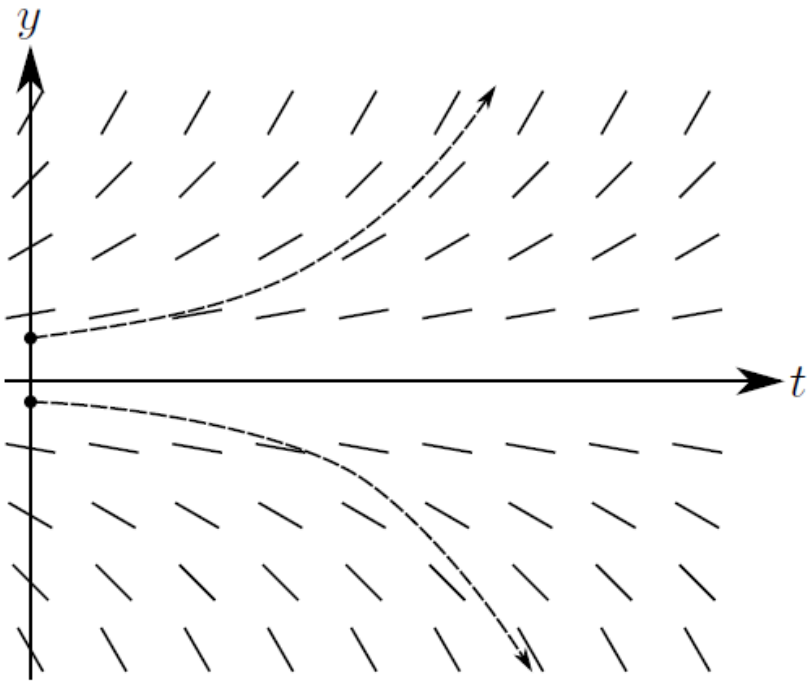


Time-independent

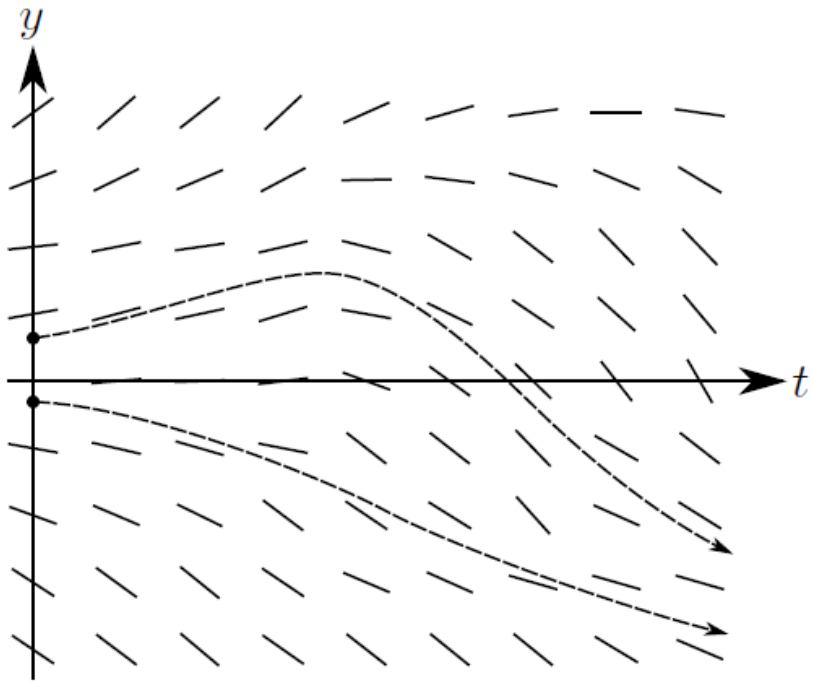


Time-dependent

Time Dependence



Time-independent



Time-dependent

Visualization: Slope field

Autonomous ODE

$$\vec{y}' = F[\vec{y}]$$

No dependence of F on t

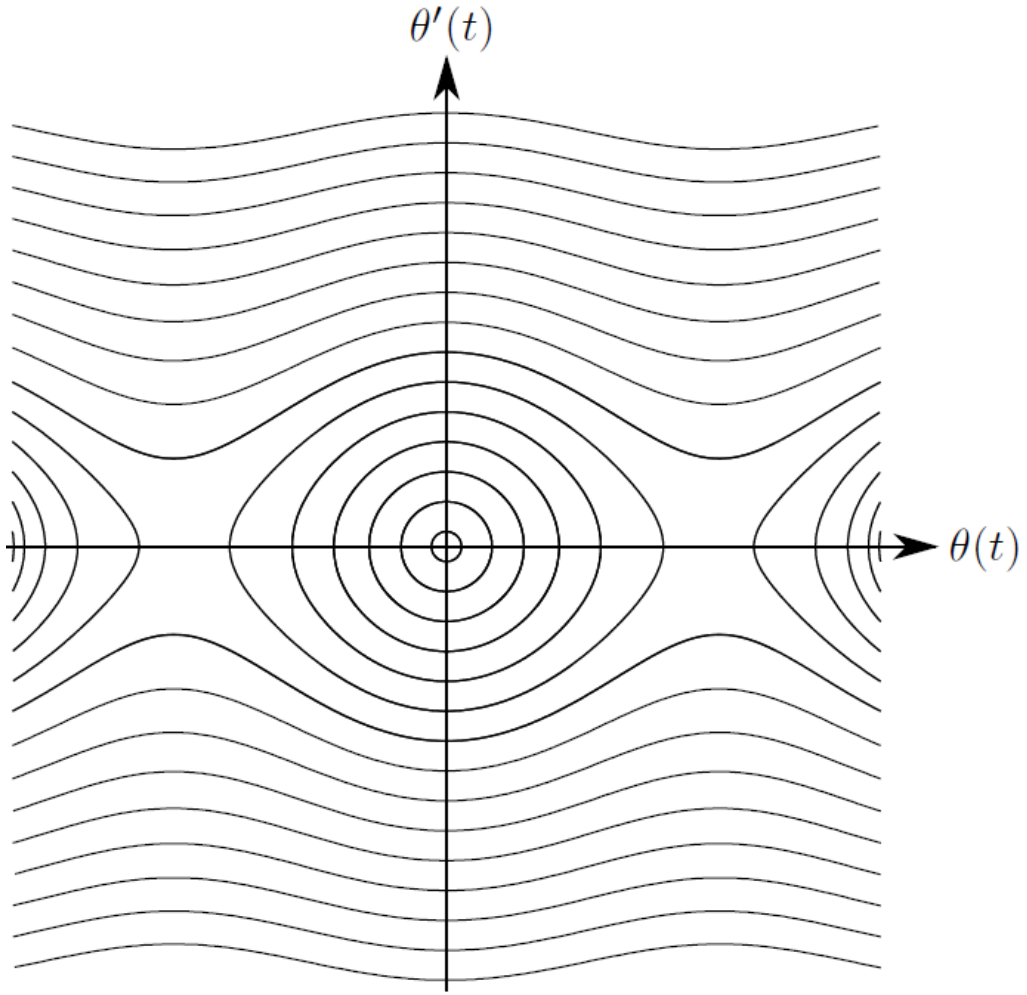
Autonomous ODE

$$\vec{y}' = F[\vec{y}]$$

No dependence of F on t

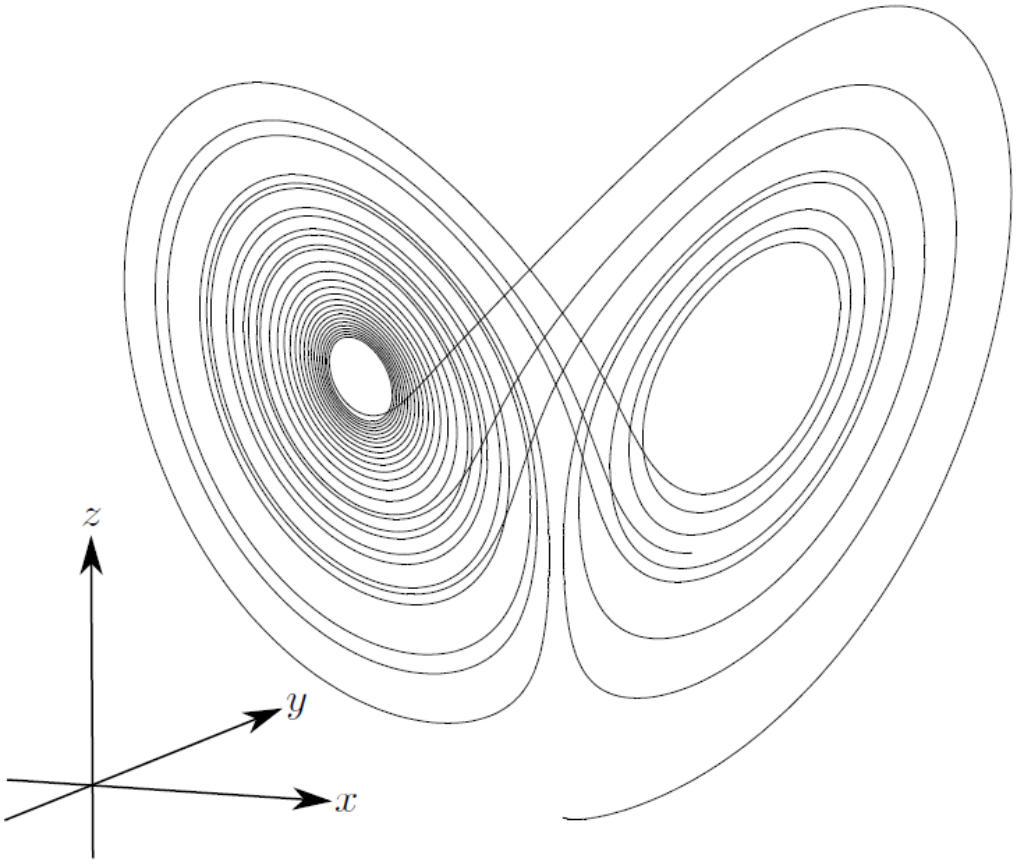
$$g'(t) = \begin{pmatrix} f'(t) \\ \bar{g}'(t) \end{pmatrix} = \begin{pmatrix} F[f(t), \bar{g}(t)] \\ 1 \end{pmatrix}$$

Visualization: Phase Space



$$\theta'' = -\sin\theta$$

Visualization: Traces



$$x' = \sigma(y - x),\ y' = x(\rho - z) - y,\ z' = xy - \beta z$$

Existence and Uniqueness

$$\frac{dy}{dt} = \frac{2y}{t}$$

Two cases:

$$y(0) = 0,\ y(0) \neq 0$$

Existence and Uniqueness

Theorem: Local existence and uniqueness

Suppose F is continuous and Lipschitz, that is, $\|F[\vec{y}] - F[\vec{x}]\|_2 \leq L\|\vec{y} - \vec{x}\|_2$ for some fixed $L \geq 0$. Then, the ODE $f'(t) = F[f(t)]$ admits exactly one solution for all $t \geq 0$ regardless of initial conditions.

Linearization of 1D ODEs

$$y' = F[y]$$

Linearization of 1D ODEs

$$y' = F[y]$$

$$\longrightarrow y' = ay + b$$

Linearization of 1D ODEs

$$y' = F[y]$$

$$\longrightarrow y' = ay + b$$

$$\longrightarrow \bar{y}' = a\bar{y}$$

Model Equation

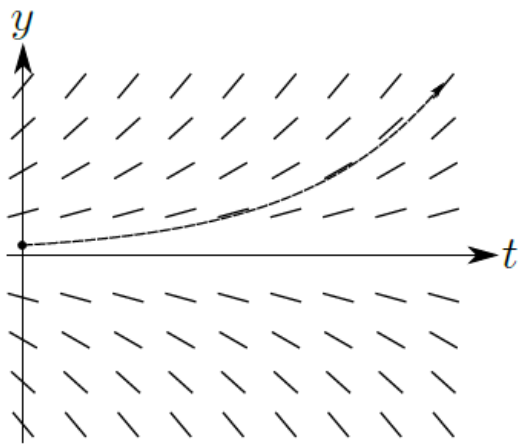
$$y' = ay$$

Model Equation

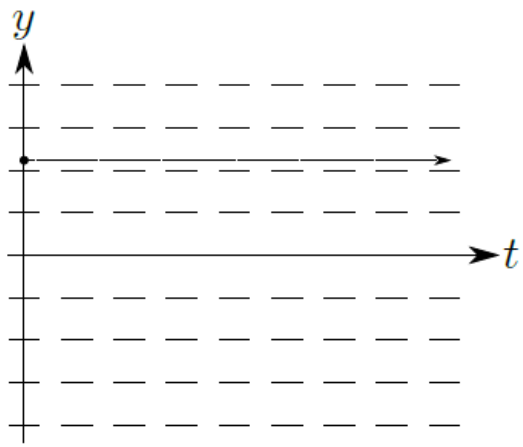
$$y' = ay$$

$$\implies y(t) = Ce^{at}$$

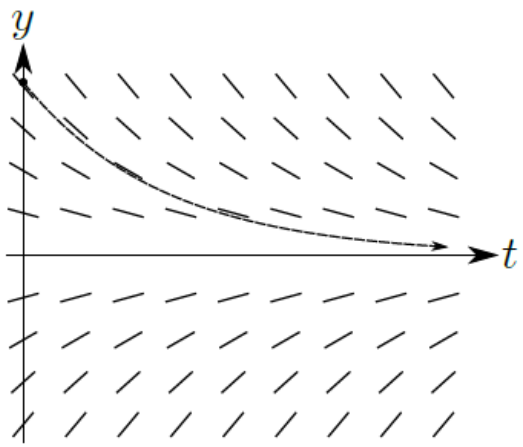
Stability: Visualization



$a > 0$



$a = 0$



$a < 0$

$$\boxed{y' = ay}$$

Three Cases

$$y' = ay, y(t) = Ce^{at}$$

1. $a = 0$: Stable

Three Cases

$$y' = ay, y(t) = Ce^{at}$$

1. $a = 0$: Stable
2. $a < 0$: Stable; solutions get closer

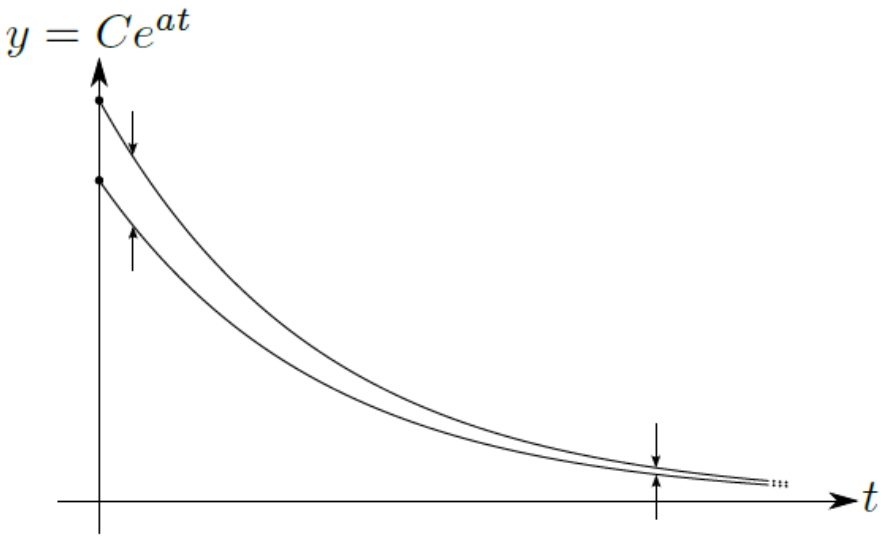
Three Cases

$$y' = ay, y(t) = Ce^{at}$$

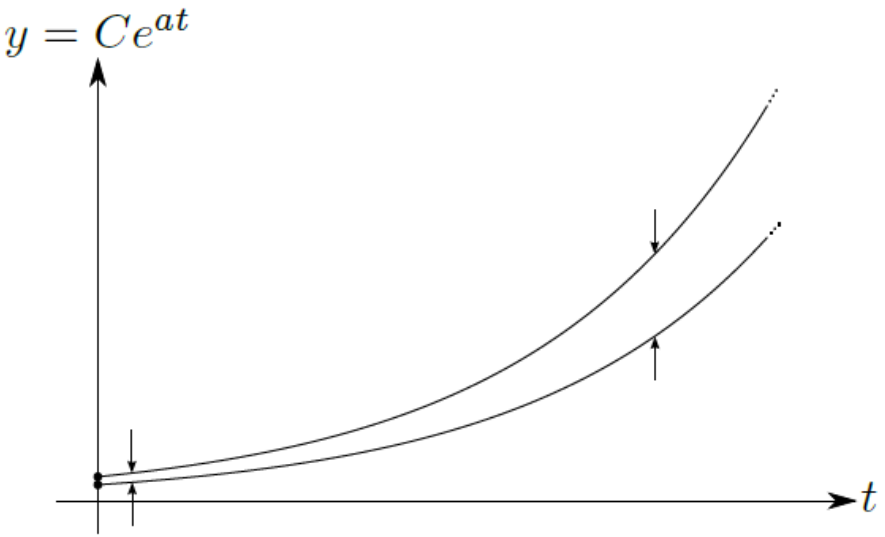
1. $a = 0$: Stable
2. $a < 0$: Stable; solutions get closer
3. $a > 0$: Unstable; mistakes in initial data amplified

Intuition for Stability

An *unstable* ODE magnifies mistakes in the initial conditions $y(0)$.



Stable ($a < 0$)



Unstable ($a > 0$)

Multidimensional Case

$$\vec{y}' = A\vec{y}, \;\; A\vec{y}_i = \lambda_i \vec{y}_i \; (\text{where } A = A^\top)$$

$$\vec{y}(0) = \sum_i c_i \vec{y}_i$$

Multidimensional Case

$$\vec{y}' = A\vec{y},\ A\vec{y}_i = \lambda_i \vec{y}_i \text{ (where } A = A^\top)$$

$$\vec{y}(0) = \sum_i c_i \vec{y}_i$$

$$\implies \vec{y}(t) = \sum_i c_i e^{\lambda_i t} \vec{y}_i$$

Multidimensional Case

$$\vec{y}' = A\vec{y},\ A\vec{y}_i = \lambda_i \vec{y}_i \text{ (where } A = A^\top)$$

$$\vec{y}(0) = \sum_i c_i \vec{y}_i$$

$$\implies \vec{y}(t) = \sum_i c_i e^{\lambda_i t} \vec{y}_i$$

Stability depends on $\max_i \lambda_i$.

Integration Strategies

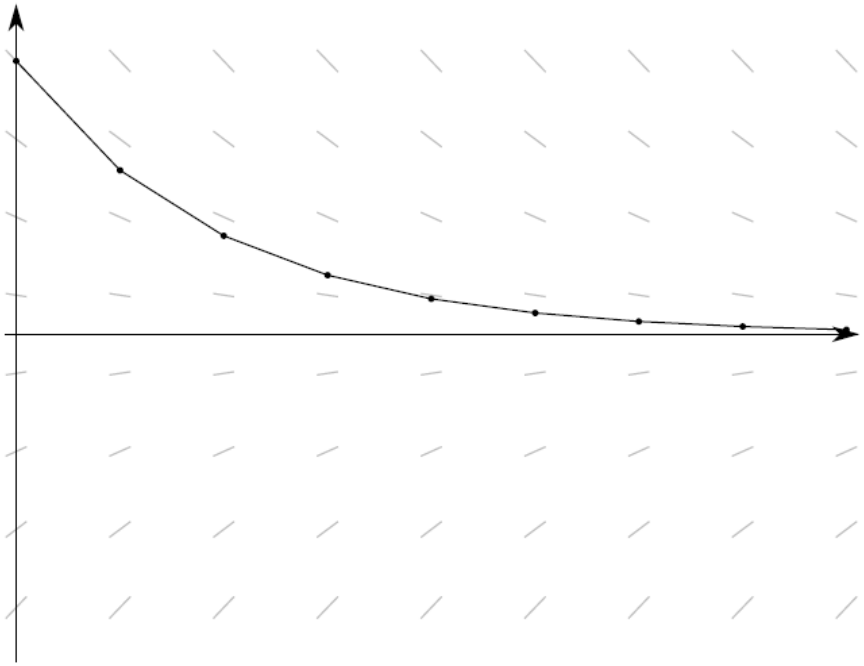
Given $\vec{y}_k$ at time t_k, generate $\vec{y}_{k+1}$ assuming $\vec{y}' = F[\vec{y}]$.

Forward Euler

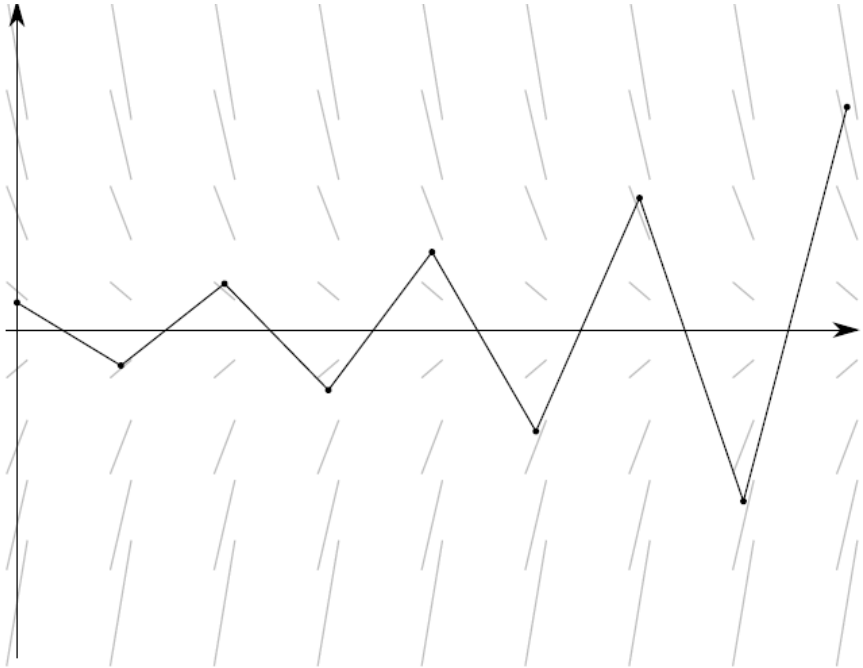
$$\vec{y}_{k+1} = \vec{y}_k + hF[\vec{y}_k]$$

- Explicit method
- $O(h^2)$ localized truncation error
- $O(h)$ global truncation error;
  "first order accurate"

Forward Euler: Stability



Stable ($a = -0.4$)



Unstable ($a = -2.3$)

Model Equation

$$y' = ay \longrightarrow y_{k+1} = (1 + ah)y_k$$

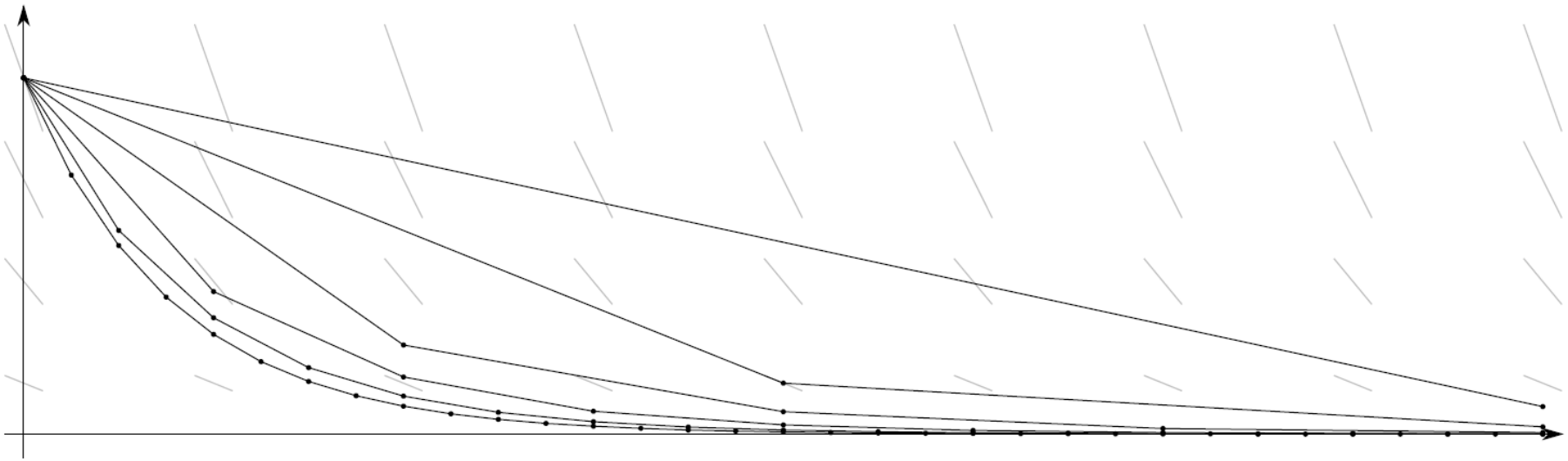
For $a < 0$, stable when $h < \frac{2}{|a|}$.

Backward Euler

$$\vec{y}_{k+1} = \vec{y}_k + hF[\vec{y}_{k+1}]$$

- Implicit method
- $O(h^2)$ localized truncation error
- $O(h)$ global truncation error;
  "first order accurate"

Backward Euler: Stability



Model Equation

$$y' = ay \longrightarrow y_{k+1} = \frac{1}{1-ah} y_k$$

Unconditionally stable!

Model Equation

$$y' = ay \longrightarrow y_{k+1} = \frac{1}{1-ah} y_k$$

Unconditionally stable!

But this has nothing to do with accuracy.

Model Equation

$$y' = ay \longrightarrow y_{k+1} = \frac{1}{1-ah} y_k$$

Unconditionally stable!

But this has nothing to do with accuracy.

Good for *stiff* equations.

Numerical Stiffness for IVP ODEs

An IVP is said to be numerically "stiff" if stability requirements dictate a much smaller time step size than is needed to satisfy the approximation requirements alone.

[Ascher & Petzold 1998]

Forward and Backward Euler on Linear ODE

$$\vec{y}' = A\vec{y}$$

- Forward Euler: $\vec{y}_{k+1} = (I + hA)\vec{y}_k$
- Backward Euler: $\vec{y}_{k+1} = (I - hA)^{-1}\vec{y}_k$

Next