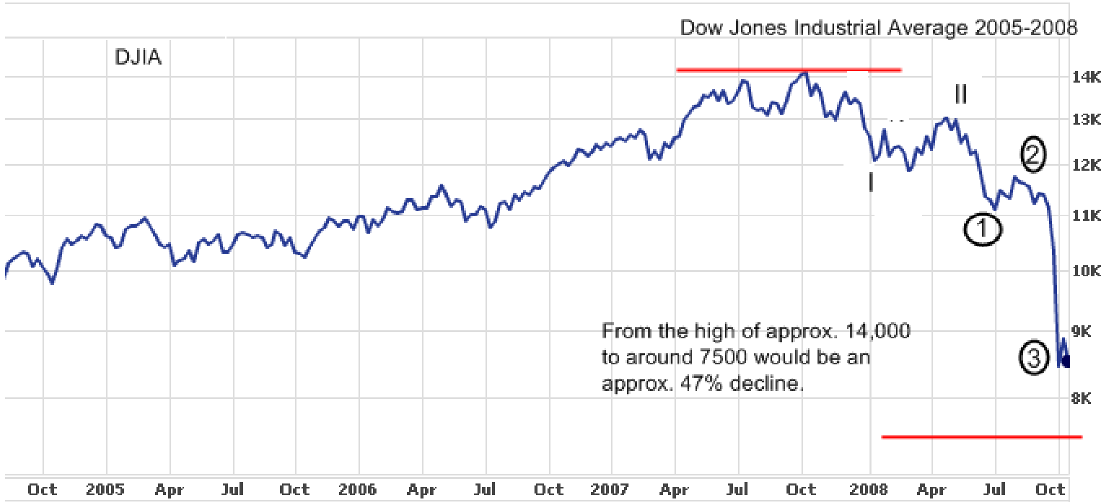
Differential Equations

In many problems, we want to determine or predict the value of some (time-)evolving quantity:

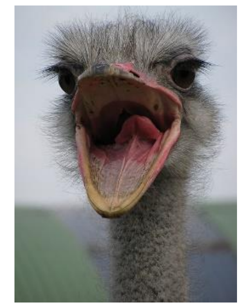
- population of a species
- motion of physical objects
- value of investments
- weather or climate
- ...



e.g., Stock Market Index During Financial Crisis

(Not) Differential Equations

If we had a known closed-form expression for the desired quantity, we could simply evaluate it.



Ex #1: The ostrich population p at time t is given by:

$$p(t) = \sin(t) + 2t^2.$$

Ex #2: The temperature T of the kumquat storage facility at time t is:

$$T(t) = 20 + \cos(\mathrm{t}).$$

Just plug in time t and evaluate to get the result!

Ordinary Differential Equations

A closed-form is often unavailable in practice!

Instead we may know a relationship between the *variable* y, and its *derivative*, y', described by a known function f:

$$y'(t) = f\big(t, y(t)\big)$$

Ex #1: The value, v, of a stock might change over time according to:

$$v'(t) = 15 \cdot \sin(t) \cdot v(t).$$

Ex #2: Some quantity q evolves w.r.t. time according to:

$$q'(t) = e^q + \sin(q(t))\, t^2.$$

Example: A Simple Population Model

Consider a mouse population, $y(t)$, over time. With enough food, we can describe the population change with the ODE

$$y'(t) = a \cdot y(t)$$

where a is some experimentally observed reproduction rate.

i.e., # of mice being born per unit time, $y'(t)$, is a constant, a, times the *current* number of mice, $y(t)$.

$\boldsymbol{y(t)}$ is not given explicitly!

Example: A Simple Population Model

For this specific ODE, $y'(t) = a \cdot y(t)$, with initial population, $y_0 = y(t_0)$, there *does* exist a closed-form solution:

$$y(t) = y_0 e^{a(t-t_0)}.$$

Aside: How can we verify it?

Compute $y'(t)$ by differentiating the closed-form, $y(t)$, and compare:

$$y'(t) = \frac{d}{dt}\left(y_0 e^{a(t-t_0)}\right) = \underbrace{y_0 e^{a(t-t_0)}}_{y(t)} \cdot a = a \cdot y(t).$$

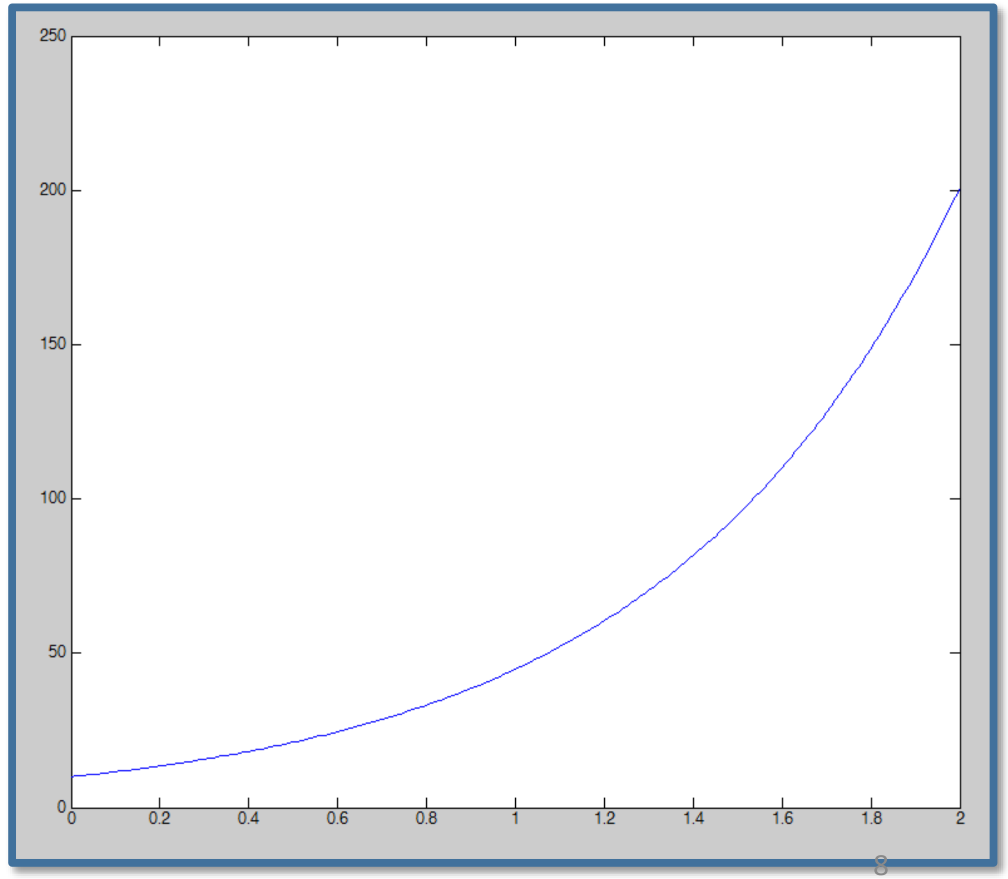
Example: A Simple Population Model

So the population would follow $y(t) = y_0 e^{a(t-t_0)}$.

Plugging in t gives the population at any desired time.

This is *exponential* growth.

But how *realistic* was our population growth model?



Example: A More Complex Model

In reality, food supplies (and space and partners and …) *are* usually limited.

Consider an enhanced model, $y'(t) = y(t) \cdot \big(a - b \cdot y(t)\big)$, where the new b term accounts for resource limits.

Observe:

1) For small $y(t)$, we again have $y'(t) \approx a \cdot y(t)$. (Exponential growth.)

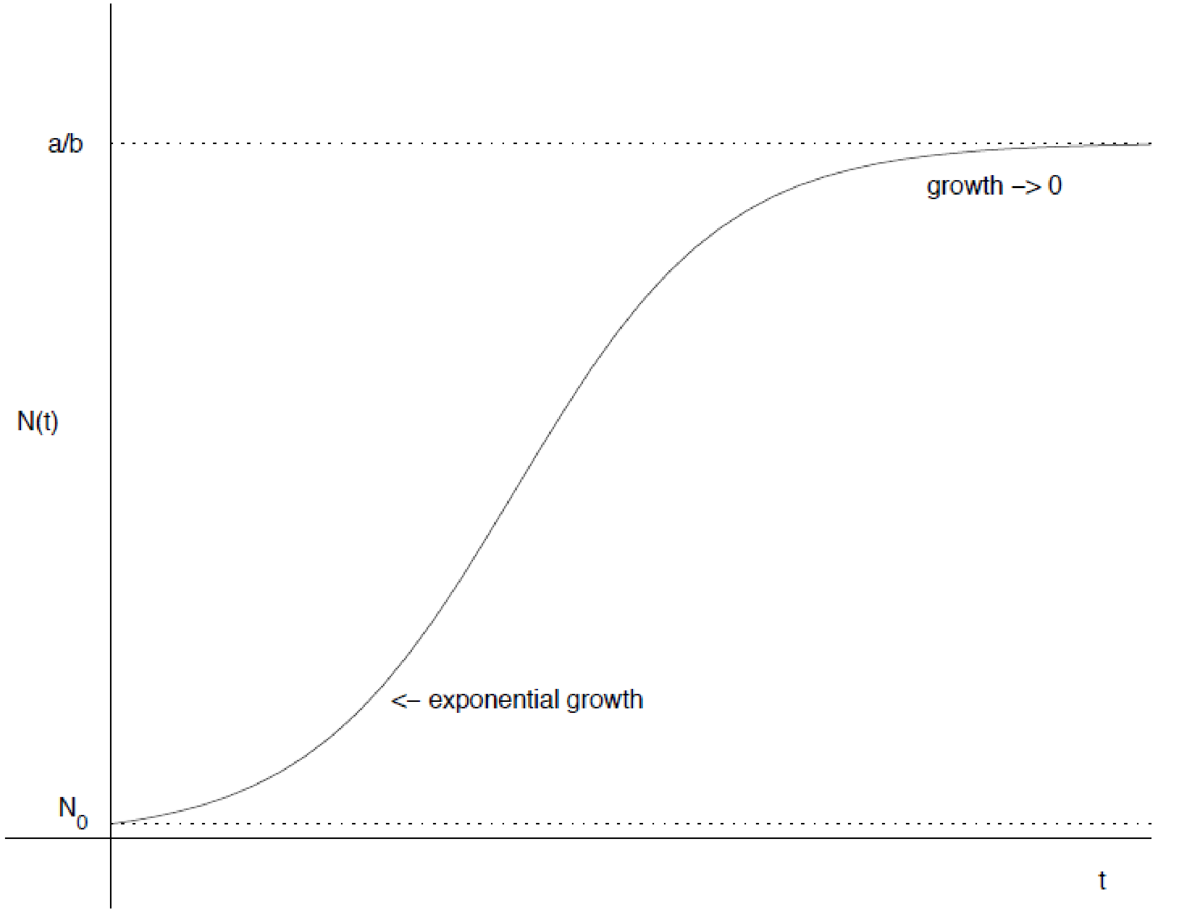
2) For $y(t) \approx \frac{a}{b}$, we have $y'(t) \approx 0$. **Population growth levels off!**

Example: A More Complex Model

This new population growth model,
$y'(t) = y(t) \cdot \big(a - b \cdot y(t)\big)$,
also has a closed-form:

$$y(t) = \frac{a y_0 e^{a(t-t_0)}}{b y_0 e^{a(t-t_0)} + (a - y_0 b)}.$$

This is *logistic growth*.



Example: *Even More* Complex Models

But what other real-world factors might affect population growth?

- Seasonal variation in birth rate.
- Seasonal variation in food supply.
- Ratio of males to females.
- Presence and population of predators.
- Many, **many** others… (and no model can perfectly capture all possible factors!)

A very *slightly* more complex model is:

$$y'(t) = y(t) \cdot (a(t) - b(t) \cdot y(t)^{\alpha}).$$

This already has *no general closed form solution*.

Can we somehow still make meaningful predictions for it?

Ordinary Differential Equations (ODEs)

Central Observation:

Even fairly simple mathematical models often lack closed form solutions, except in very special cases.

(Partial) Solution:

For such problems, we will develop “numerical methods” to find *approximate* solutions.

Form of the **Initial Value Problem (IVP)**

The general form is a differential equation

$$y'(t) = f\big(t, y(t)\big)$$

f is the "Dynamics Function"

where f is specified, and the initial values are

$$y(t_0) = y_0.$$

"Initial Conditions"

Translation:

The rate that y is changing is given by a function f that depends on the current time t and value of y.

We know f and the starting value of y.

IVP Form example

In our population example, we had $y'(t) = a \cdot y(t)$, with some initial pop. at time $t_0 = 0$, of say $y_0 = 100$ mice.

So for this problem…

The **dynamics function** is: $f\left(t, y(t)\right) = a \cdot y(t)$.

The **initial condition** is: $y(t_0) = 100$.

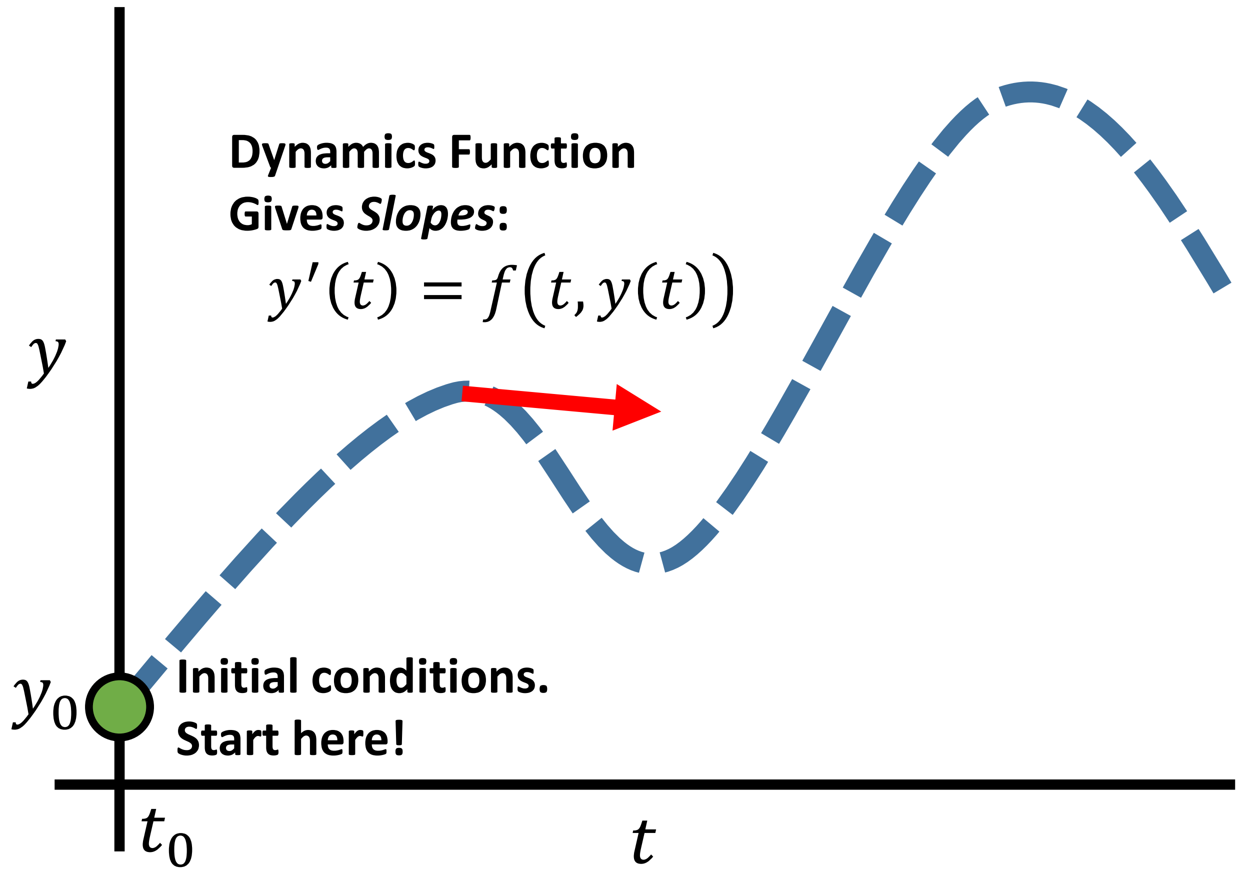
This information characterizes our initial value problem.

Visualizing the Initial Value Problem (IVP)

Given:

- Initial value, y_0.
- Dynamics function, $y' = f(t, y(t))$

Problem: Can we (approximately) find the value at later times?



Two Possible Complications

There are two common ways that ODE problems, $y'(t) = f\big(t, y(t)\big)$, can become more complicated.

1. By involving more than one unknown variable (not just y):
   "*Systems* of differential equations"

2. By involving second, third, or higher derivatives (not just y'):
   "*Higher order* differential equations"

1. *Systems* of Differential Equations

Consider a model with *multiple* variables of interest.

E.g. *x* and *y* coordinates of a moving object.

This gives a *system* of differential equations, such as

$$\underbrace{\begin{aligned} x'(t) &= f_x\big(t, x(t), y(t)\big) \\ y'(t) &= f_y\big(t, x(t), y(t)\big) \end{aligned}}_{\text{Two dynamics functions.}} \underbrace{\begin{aligned} &, \text{with } x(t_0) = x_0, \\ &, \text{with } y(t_0) = y_0. \end{aligned}}_{\text{Two initial conditions.}}$$

1. *Systems* of Differential Equations

The system…

$$x'(t) = f_x\big(t, x(t), y(t)\big), \text{with } x(t_0) = x_0$$
$$y'(t) = f_y\big(t, x(t), y(t)\big), \text{with } y(t_0) = y_0$$

can be written in *vector form* just by stacking:

$$\begin{bmatrix} x(t) \\ y(t) \end{bmatrix}' = \begin{bmatrix} f_x\big(t, x(t), y(t)\big) \\ f_y\big(t, x(t), y(t)\big) \end{bmatrix} \text{ with } \begin{bmatrix} x(t_0) \\ y(t_0) \end{bmatrix} = \begin{bmatrix} x_0 \\ y_0 \end{bmatrix}$$

or in vector notation:

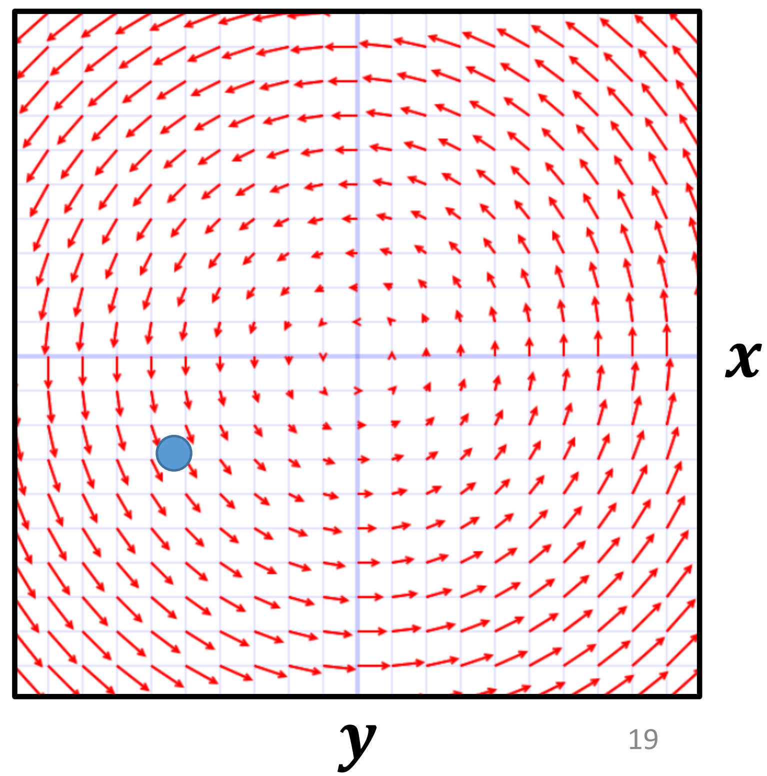
$$\vec{\boldsymbol{x}}'(t) = \vec{\boldsymbol{f}}(t, \vec{\boldsymbol{x}}(t)) \text{ with } \vec{\boldsymbol{x}}(t_0) = \vec{\boldsymbol{x}}_0.$$

Example: Dust Particle in Wind Field

The 2D position of a particle of dust is $\vec{P}(t) = (x(t), y(t))$.

We are given a wind vector function, $\vec{P}'(\mathrm{t}) = \big(-y(t), x(t)\big)$, along with some initial particle position, $\vec{P}_0$.

Goal: Determine the **path** of the dust particle over time, by solving the IVP for future values of $\vec{P}$.



2. *Higher Order* Differential Equations

We will focus on *first order* differential equations, with only a first derivative:

$$y'(t) = f\big(t, y(t)\big).$$

More generally, the *order* is the highest derivative appearing in the equation:

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

We can often transform them into first order equations (more later!)

Example - 2nd order vs. 1st order system

Consider a **2nd order** differential equation

$$y''(t) = ty'(t) - ay(t) + \sin(t)$$
$$\text{with } y(0) = 2 \text{ and } y'(0) = 3.$$

An equivalent **system** of **1st order** differential equations, in terms of variables $y(t)$ and $z(t) = y'(t)$, is:

$$y'(t) = z(t)$$
$$z'(t) = t \cdot z(t) - a \cdot y(t) + \sin(t)$$
$$\text{with } y(0) = 2 \text{ and } z(0) = 3.$$

The Story So Far

Many important phenomena are well-described by (systems of) ***ordinary differential equations***.

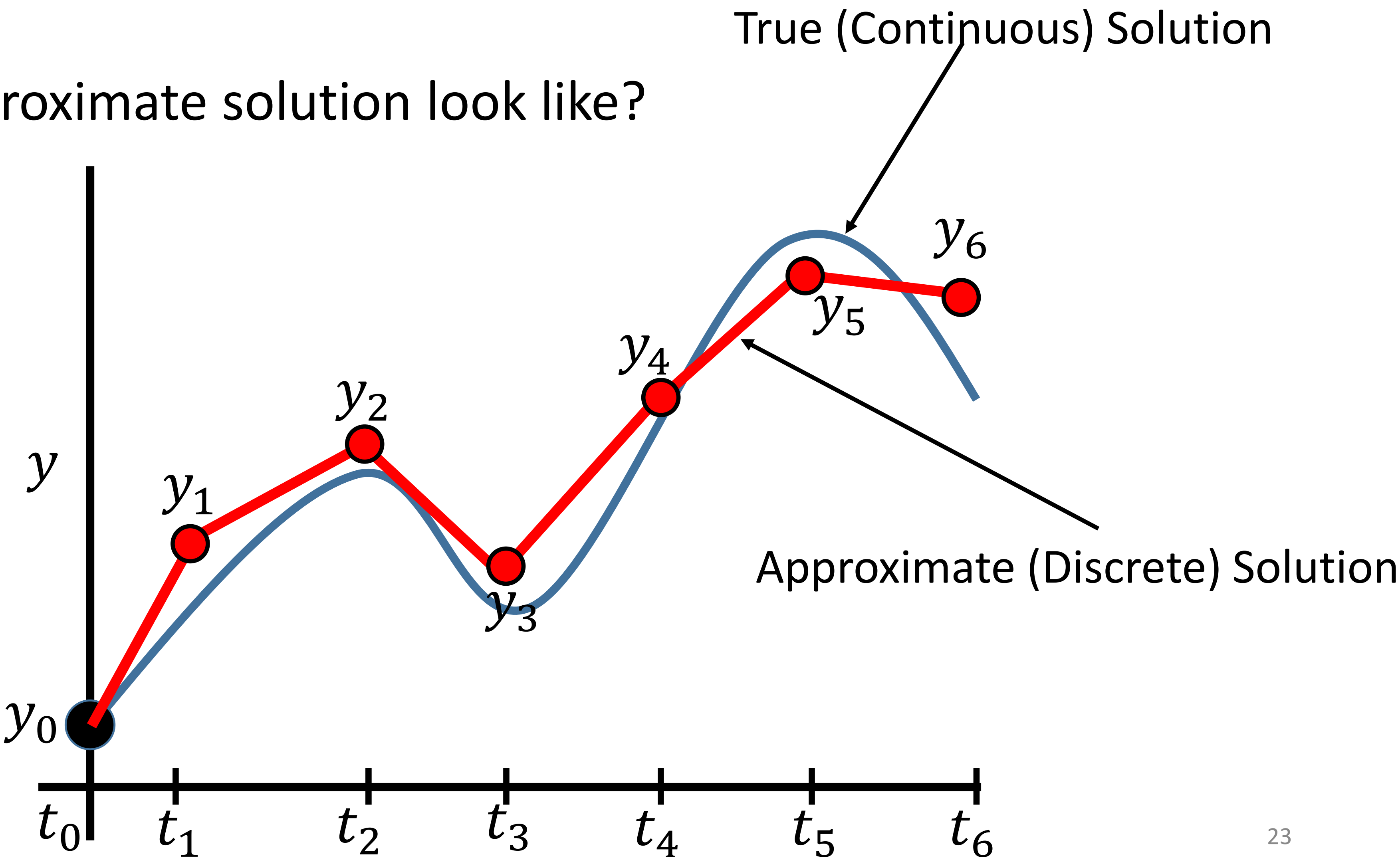
We are given an *initial value problem*: we have the starting point and the derivative(s), but not the unknown function itself.

Analytical closed-form solutions exist rarely!

We will use numerical methods to give approximate solutions.

Numerical Schemes for ODEs

What will an approximate solution look like?



Numerical Schemes for ODEs

The numerical solution will be a discrete set of time/value pairs, (t_i, y_i).

At each time instant, t_i, the value y_i should *approximate* the true solution, $y(t_i)$.

(Given these points, you *could* smoothly approximate intermediate times using our interpolation techniques! [Piecewise polynomials, Hermite curves, splines, etc.])

Aside: Interpolation v.s. ODEs

One way to relate ODEs and Interpolation:

Interpolation: Given a finite discrete set of data points, find an approximate continuous/smooth function that matches them.

(DISCRETE->CONTINUOUS)

ODEs: Given a differential equation *describing* the true/continuous solution, find a set of discrete samples that approximately matches it.

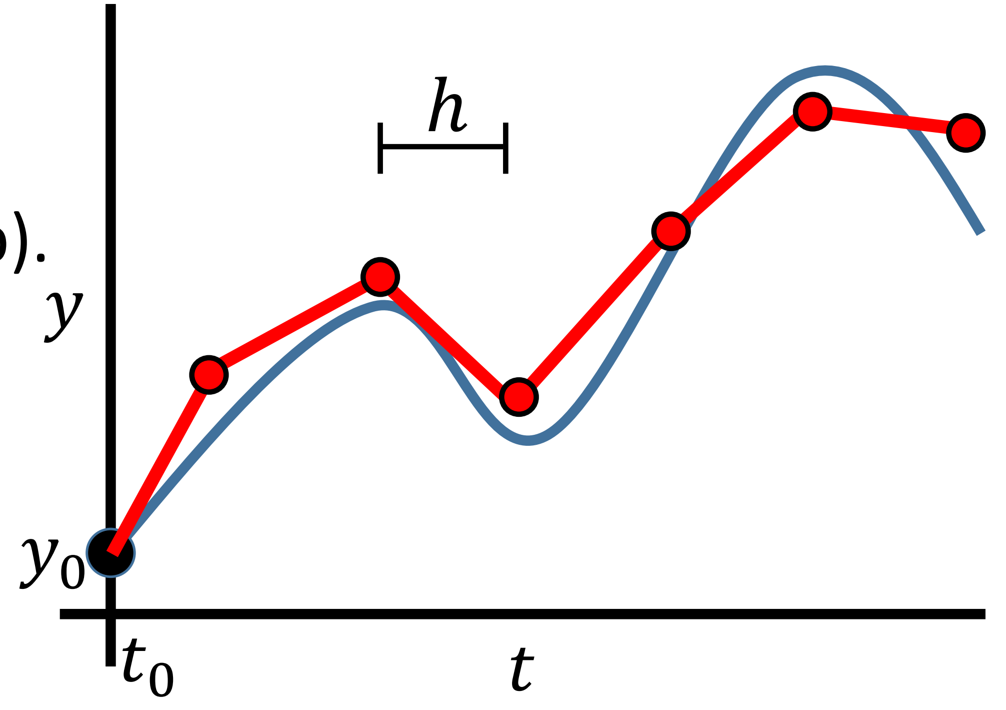
(CONTINUOUS->DISCRETE)

Time-Stepping

Given initial conditions, we repeatedly *step* sequentially forward to the next *time* instant, using the derivative info, y', and a timestep, h.

Set $n = 0, t = t_0, y = y_0, h = h_0$.

1. Compute h_n and y_{n+1} (the key step).
2. Increment time, $t_{n+1} = t_n + h_n$.
3. Advance, $n = n + 1$.
4. Repeat.



Time-Stepping - Variations

There are several varieties/categories of time-stepping methods:

- Single-step: use dynamics function f and current info, (t_n, y_n).
- Multi-step: use dynamics function f and info from both current *and previous timesteps*.

- One may use a constant or varying time size step, h.

- Explicit: y_{n+1} is an explicit function of known data.
- Implicit: y_{n+1} is given implicitly; must solve an algebraic equation.

Forward Euler - Idea

Forward Euler is a simple (explicit, single-step) scheme.

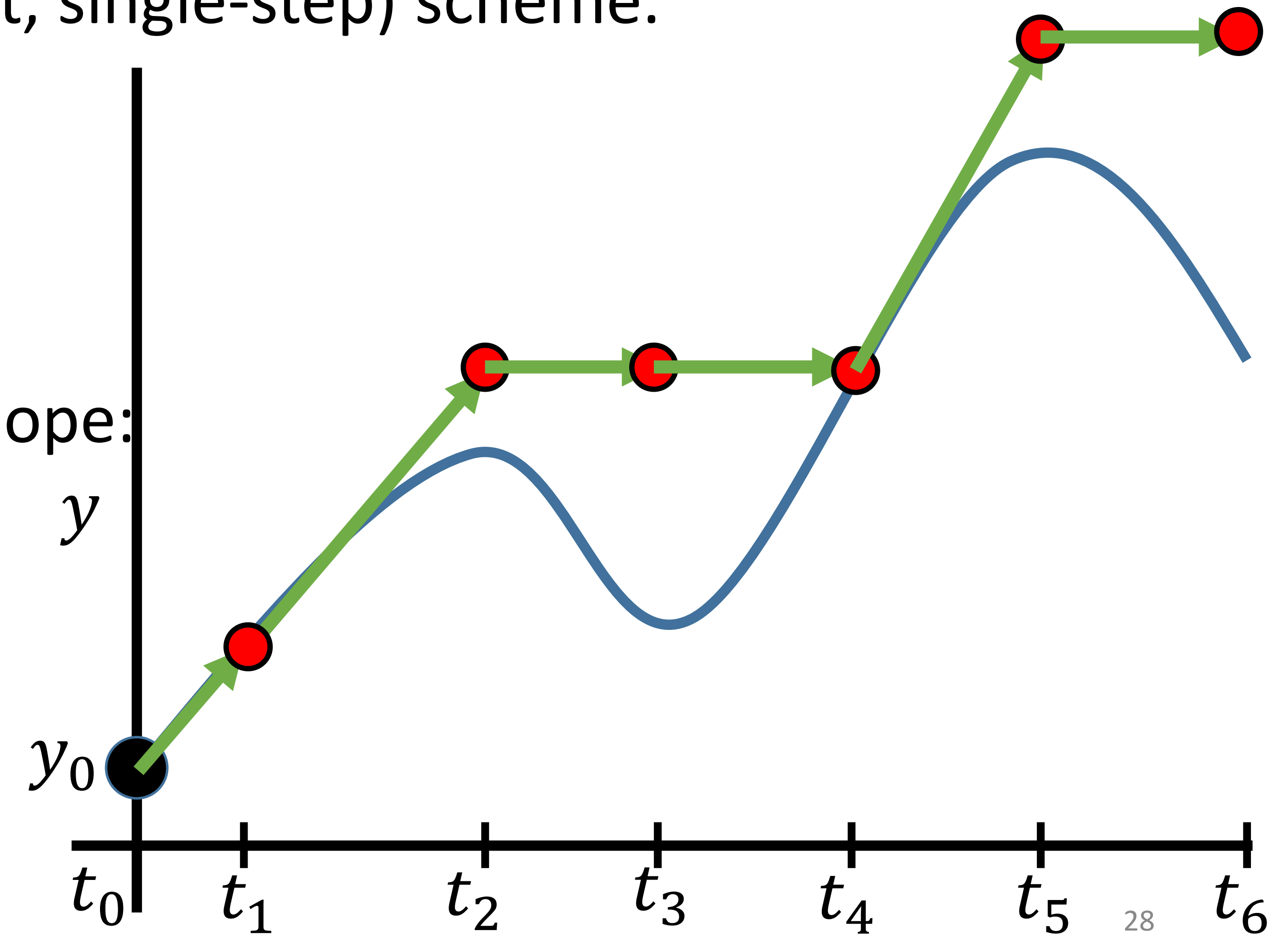
Compute the current slope:

$$y_n' = f(t_n, y_n)$$

Step in a straight line with that slope:

$$y_{n+1} = y_n + h \cdot y_n'$$

Repeat.



Forward Euler - Summary

The Forward Euler scheme is

$$y_{n+1} = y_n + hf(t_n, y_n)$$

At a given step, we have the value y_n, the time t_n, and the time step h.
The slope is given by the dynamics function f, so we evaluate it.

This directly gives a new value, y_{n+1}, at the new time, $t_{n+1} = t_n + h$.

Time-Stepping as a Recurrence

Time-stepping will amount to using a **recurrence relation** to gradually approximate the function value at later and later times.

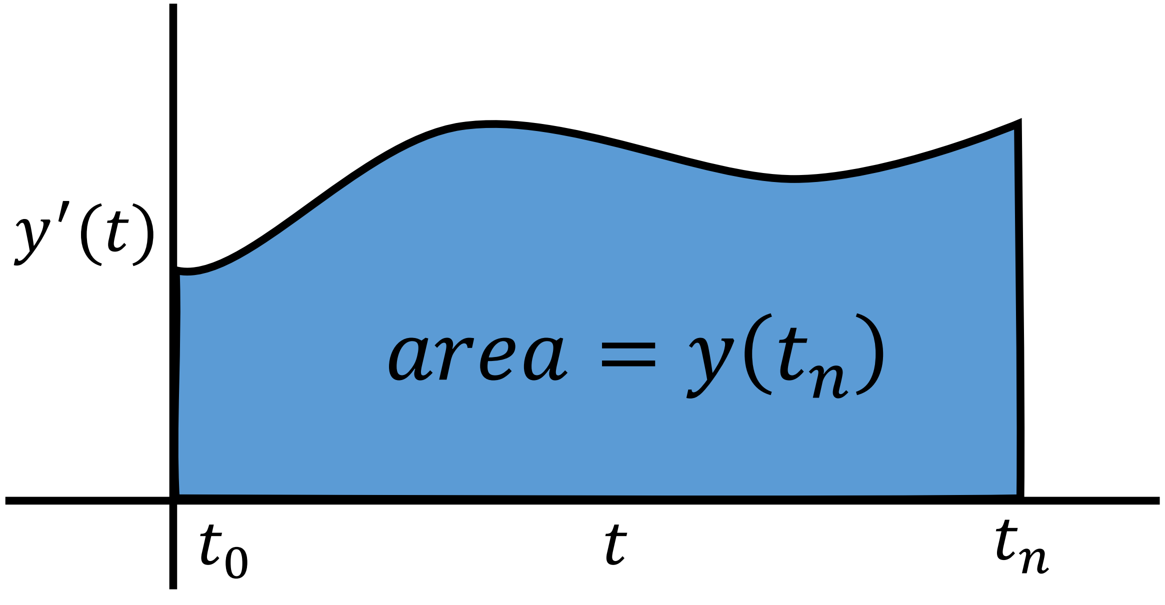
Given y_0, approximate y_1. Given y_1, approximate y_2. etc.

Later we'll analyze stability of such schemes, building on the stability analysis ideas we've seen before.

Time Stepping As "Time Integration"

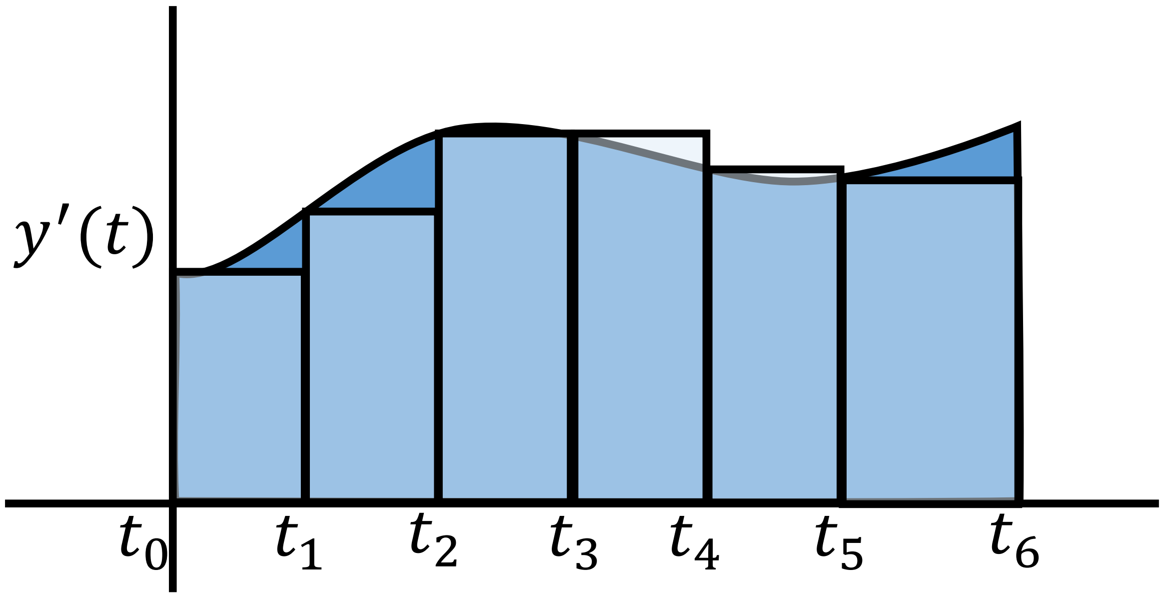
Such methods are also called time-integration schemes: we are *integrating over time* to find the value of y, using its derivative(s).

If we plot $y'(t) = f(t, y(t))$, *time-stepping* amounts to accumulating the *area under the curve* to estimate $y(t)$ at a given time t.



Forward Euler - Time Integration View

So forward Euler, $y_{n+1} = y_n + hf(t_n, y_n)$, is equivalent to summing rectangles of size $h_i \cdot y'(t_i)$, where the "height" $y'(t) = f(t, y(t))$ is evaluated at the "left side" (current time).



Better time-stepping schemes generally correspond to better approximations of the integral!

Forward Euler - Example

Consider the simple IVP $y'(t) = 2y(t)$, with initial conditions at $t_0 = 1$ of $y(t_0) = 3$.

a) Write down the recurrence for Forward Euler on this problem.

b) Use forward Euler to estimate y at time $t = 5$, w/ step size of $h = 1$.

c) Compare against the true solution, $y(t) = 3e^{2(t-t_0)}$.

See sample Matlab code too.