

FALL 2025

MEETING #3

Computational Modeling in Engineering and the Sciences

ASHTON COLE

Graduate Student, Computational Hydraulics Group, The University of Texas at Austin

Agenda

- Discussion of Baigent - Lotka-Volterra Dynamics and Runge-Kutta
- Introduction to multidimensional ODEs
- Demo: SEIR and Lotka-Volterra

**Assignment: (Lightly) read Jah - Multiple Object Space Surveillance.
Review Newton's and Kepler's laws. We'll discuss next week.**

Discussion of Baigent - Lotka-Volterra Dynamics

Introduction to Multidimensional ODEs

Introduction to Multidimensional ODEs

- **ordinary differential equation:** a differential equation with only one **independent** variable
- Introducing the IVP:

$$\frac{du}{dt} = f(u, t)$$
$$u(t_0) = u_0$$
$$u(t) = ?$$

Introduction to Multidimensional ODEs

- **system of ordinary differential equations:** a set of differential equations with only one **independent** variable
- Introducing the IVP:

$$\begin{aligned}\frac{d\mathbf{u}}{dt} &= f(\mathbf{u}, t) \\ \mathbf{u}(t_0) &= \mathbf{u}_0 \\ \mathbf{u}(t) &= ?\end{aligned}$$

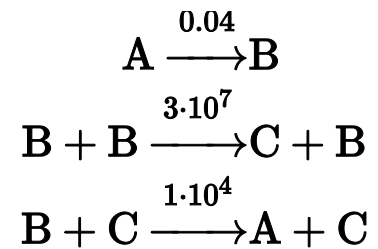
Introduction to Multidimensional ODEs

- **linear system of ordinary differential equations:** writable as a matrix equation
- Exact solution
- Linearization in a small region to understand behavior

$$\begin{aligned}\frac{d\mathbf{u}}{dt} &= \mathbf{A}\mathbf{u} \\ \mathbf{u}(t_0) &= \mathbf{u}_0 \\ \mathbf{u}(t) &= ?\end{aligned}$$

Introduction to Multidimensional ODEs

- **stiff ordinary differential equations:** difficult to solve without a tiny step size



$$\begin{aligned} \dot{x} &= -0.04x + 10^4 y \cdot z \\ \dot{y} &= 0.04x - 10^4 y \cdot z - 3 \cdot 10^7 y^2 \\ \dot{z} &= 3 \cdot 10^7 y^2 \end{aligned}$$

Demo: SEIR and Lotka-Volterra

- Get in groups. Complete the `my_integrators.py` module, and use it to solve the SEIR and Lotka-Volterra systems of ODE's.
- Bonus tasks:
 - Compare against the exact solution for Lotka-Volterra, which is defined implicitly.
 - Create error log-log plots.
 - Generate phase plots.
 - Try using implicit or high-order solvers, e.g. from `scipy`.