# MATH 3341: Introduction to Scientific Computing Lab

Libao Jin

University of Wyoming

May 13, 2020

# Lab 14: Built-in ODE Solvers in MATLAB

# Built-in ODE Solvers for Stiff/Nonstiff ODEs

# Stiff ODEs

**Definition**

A stiff equation is a differential equation for which certain numerical methods for solving the equation are numerically unstable, unless the step size is taken to be extremely small. It has proven difficult to formulate a precise definition of stiffness, but the main idea is that the equation includes some terms that can lead to rapid variation in the solution.

# Choose an ODE Solver

Some ODE problems exhibit *stiffiness*, or difficulty in evaluation. For example, if an ODE has two solution components that vary on drastically different time scales, then the equation might be stiff. You can identify a problem as stiff if nonstiff solvers (such as `ode45`) are unable to solve the problem or are extremly slow. If you observe that a nonstiff solver is very slow, try using a stiff solver such as `ode15s` instead. When using a stiff solver, you can improve reliability and efficiency by supplying the Jacobian matrix or its sparsity pattern.

# Nonstiff ODEs Solvers - ode45, ode23, and ode113

# ode45: Solve non-stiff ODEs, medium order method

- `[TOUT,YOUT] = ode45(ODEFUN,TSPAN,Y0)` with `TSPAN = [T0 TFINAL]` integrates the system of differential equations y' = f(t,y) from time `T0` to `TFINAL` with initial conditions `Y0`. `ODEFUN` is a function handle. To obtain solutions at specific times `T0,T1,...,TFINAL` (all increasing or all decreasing), use `TSPAN = [T0 T1 ... TFINAL]`.
- `[TOUT,YOUT] = ode45(ODEFUN,TSPAN,Y0,OPTS)` solves as above with default integration properties replaced by values in `OPTS`, an argument created with the `odeset` function.
- `SOL = ode45(ODEFUN,[T0 TFINAL],Y0...)` returns a structure that can be used with `deval` to evaluate the solution or its first derivative at any point between `T0` and `TFINAL`. The steps chosen by `ode45` are returned in a row vector `SOL.x`. For each `I`, the column `SOL.y(:,I)` contains the solution at `SOL.x(I)`.

# ode23: Solve non-stiff ODEs, low order method

- `[TOUT,YOUT] = ode23(ODEFUN,TSPAN,Y0)` with `TSPAN = [T0 TFINAL]` integrates the system of differential equations y' = f(t,y) from time `T0` to `TFINAL` with initial conditions `Y0`. `ODEFUN` is a function handle. To obtain solutions at specific times `T0,T1,...,TFINAL` (all increasing or all decreasing), use `TSPAN = [T0 T1 ... TFINAL]`.
- `[TOUT,YOUT] = ode23(ODEFUN,TSPAN,Y0,OPTS)` solves as above with default integration properties replaced by values in `OPTS`, an argument created with the `odeset` function.
- `SOL = ode23(ODEFUN,[T0 TFINAL],Y0...)` returns a structure that can be used with `deval` to evaluate the solution or its first derivative at any point between `T0` and `TFINAL`. The steps chosen by `ode23` are returned in a row vector `SOL.x`. For each `I`, the column `SOL.y(:,I)` contains the solution at `SOL.x(I)`.

# ode113: Solve non-stiff ODEs, variable order method

- `[TOUT,YOUT] = ode113(ODEFUN,TSPAN,Y0)` with `TSPAN = [T0 TFINAL]` integrates the system of differential equations y' = f(t,y) from time `T0` to `TFINAL` with initial conditions `Y0`. `ODEFUN` is a function handle. To obtain solutions at specific times `T0,T1,...,TFINAL` (all increasing or all decreasing), use `TSPAN = [T0 T1 ... TFINAL]`.
- `[TOUT,YOUT] = ode113(ODEFUN,TSPAN,Y0,OPTS)` solves as above with default integration properties replaced by values in `OPTS`, an argument created with the `odeset` function.
- `SOL = ode113(ODEFUN,[T0 TFINAL],Y0...)` returns a structure that can be used with `deval` to evaluate the solution or its first derivative at any point between `T0` and `TFINAL`. The steps chosen by `ode113` are returned in a row vector `SOL.x`. For each `I`, the column `SOL.y(:,I)` contains the solution at `SOL.x(I)`.

# Stiff ODEs Solvers - ode15s, ode23s,ode23t, and ode23tb

# Fully Implicit ODEs Solvers - `ode15i`