Midterm Presentations & Boundary Value Problems

Lecture 8, Computational Astrophysics (ASTR660)

Hsiang-Yi Karen Yang, NTHU, 11/03/2022

Date	Time	Name	Title
11/3/22	14:25-14:40	李佳倫	Effects of AGN Quasar Feedback in Galaxy Clusters
	14:40-14:55	石郡翰	Using the Mechanism of AIRES to Simulate the Propagation of Cosmic Rays in the universe
	14:55-15:10	凌志騰	A gravitational SPH simulation of galaxy collision from scratch
	15:30-15:45	張修瑜	Three-body problem in the Moon, Earth, and the Sun.
	15:45-16:00	簡嘉成	Improving my magnetohydrodynamic simulation for non-ideal plasma fluid
	16:00-16:15	龔一桓	Ram pressure striping of galaxy cluster: Can a subcluster be assumed as gasless

For next week's presenters, please fill in your titles in the spreadsheet

Evaluation sheet for the midterm presentations

Name of Presenter:

Title of Presentation:

	Poor				Excellent
CONTENTS OF PROJECT PROPOSAL	**1**	**2**	**3**	**4**	**5**
Were the scientific motivations of the project clearly described?	,	,	,	,	,
Was proper background information on the topic given?	,	,	,	,	,
Was the proposed method suitable for the scientific problem to be tackled?	,	,	,	,	,
Was the numerical method chosen for this project clearly explained?	,	,	,	,	,
Was the motivation for choosing the numerical method clearly described?	,	,	,	,	,
Did the presenter provide sufficient justification for the feasibility of the project? ...	,	,	,	,	,
Did the presenter have a clear understanding of the material presented?	,	,	,	,	,
PRESENTATION SKILLS	**1**	**2**	**3**	**4**	**5**
Were the main ideas presented in an orderly and clear manner?	,	,	,	,	,
Did the presentation fill the time allotted?	,	,	,	,	,
Were the visual aids well prepared with clear and understandable figures/text?	,	,	,	,	,
Did the presentation appropriately cite relevant references?	,	,	,	,	,
Did the speaker make good eye contact and maintain the interest of the audience? ...	,	,	,	,	,
How well did the presenter handle questions from the audience?	,	,	,	,	,

COMMENTS/ENCOURAGEMENTS/CONSTRUCTIVE CRITICISMS/WHAT DO YOU LIKE ABOUT THIS PRESENTATION?

Evaluation forms for today's presentations 11/3/2022

1. Li, Jia-Lun
2. Shih, Jyun-Han
3. Ling, Chih-Teng
4. Chang, Siou-Yu
5. Chien, Chia-Chen
6. Gong, Yi-Huan

Ordinary differential equations -- summary

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

- *Ordinary differential equations (ODEs)* are differential equations that involve only one independent variable (e.g., time) and can be solved by broken down to a system of 1st-order ODEs
- Errors occur mainly due to *truncation errors*. When solving ODEs, one has to care about
  - *accuracy* – choosing a numerical method with desired accuracy to reduce local errors per step
  - *stability* – choosing a step size that satisfies the stability criterion for the numerical method
  - *cost* – choosing a step size as large as allowed to save computational time
- *Stiff ODEs* – systems in which the step size is severely limited by the stability criterion

Ordinary differential equations -- summary

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

- Algorithms for solving ODEs: Euler's (x), backward Euler's, Taylor series, Runge-Kutta methods (RK2, RK4), leapfrog...
- *Order of accuracy for an ODE algorithm:* to what order of *h* does the approximation agrees with the exact function
- *Explicit* methods: use info only at t_k, easier to compute, smaller stability region (requiring smaller step sizes), inefficient for stiff ODEs
- *Implicit* methods: use info also at t_{k+1}, require solving nonlinear equations to obtain next solution, larger stability region (sometimes unconditionally stable), suitable for solving stiff ODEs
- *Symplectic integrators*: algorithm for solving Hamiltonian systems, nearly energy conserving, useful for simulating celestial movements

This lecture…

- Introduction to boundary value problems (BVPs)
- Algorithms for solving BVPs & exercise

Intro to boundary value problems

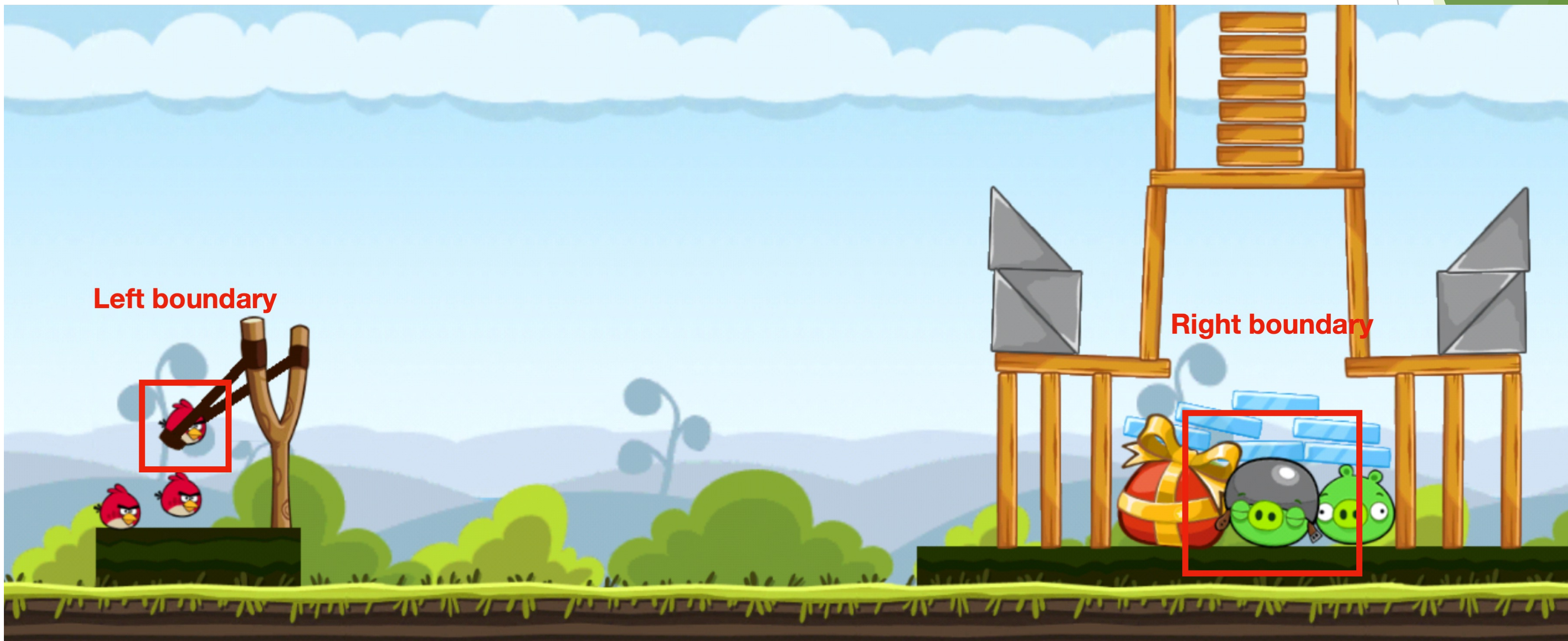
Initial value problems vs. boundary value problems

$$\boxed{y^{(k)}(t) = f(t, y, y', \ldots, y^{(k-1)})}$$

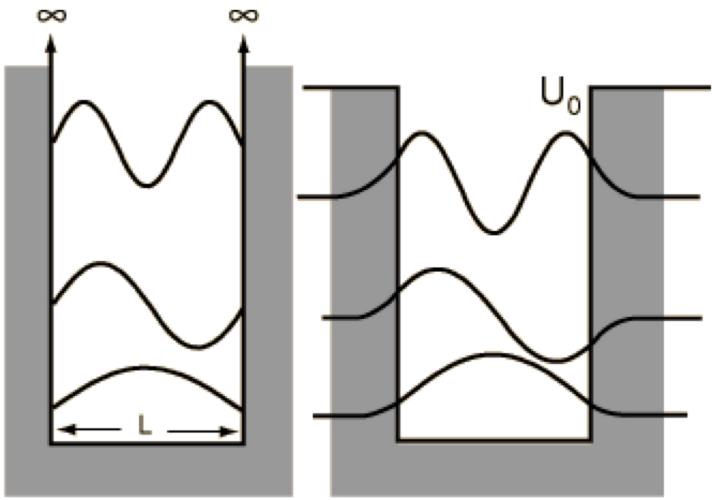
- *Initial value problems (IVPs)* -- when solving *ODEs*, initial values are required to determine a unique solution (last lecture)
- k-th order ODEs require k conditions/constraints
- *Boundary value problems (BVPs)* – specify the *boundary conditions (B.C.)* at end points a and b as the constraints instead
- **Example**: for a 2nd-order ODE $y'' = f(t, y)$

IVP	*BVP*
$y(a)$	$y(a)$
$y'(a)$	$y(b)$

BVPs – example #1



BVPs – example #2



$$-\frac{h^2}{2m}\frac{d^2\varphi}{dx^2} + V(x)\varphi = E\varphi$$

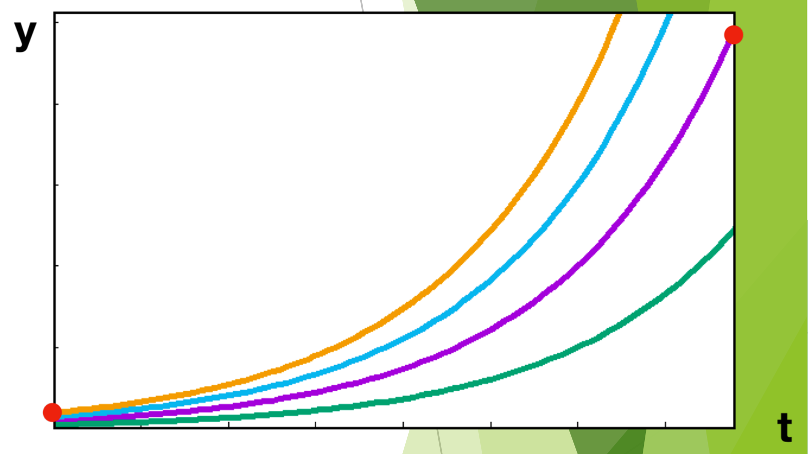
$$u'' = \lambda f(y, u, u')$$

- BVP -- solving for the wave function φ in an infinite potential well using Schrodinger's equation given *V(x)* and *E*
- This becomes an eigenvalue problem when one needs to solve for the eigenvalue λ as well

Algorithms for solving BVPs

Algorithm #1 – shooting method

- If $y'(a)$ is unknown, then the problem is reduced to an ***IVP*** which we know how to solve
- Given some initial guess of $y'(a)$, one could solve the ODE and see whether $y(b)$ agrees with β
- Vary the values of $y'(a)$ until $y(b)$ matches with β

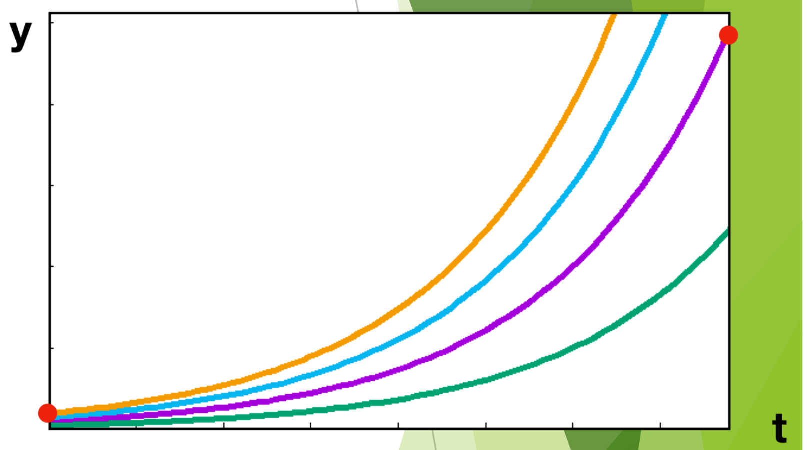


Boundary conditions (BC):

$$y(a) = \alpha$$
$$y(b) = \beta$$

Algorithm #1 – shooting method

- Instead of varying *y'(a)* randomly, one could try to find the range of *y'(a)* that ***brackets*** the solution, e.g., the orange and green lines in the figure
- Then one can use the ***bisection*** method to narrow down the search range for *y'(a)* until *y(b)* matches with β



Boundary conditions (BC):

$$y(a) = \alpha$$
$$y(b) = \beta$$

Algorithm #2 – finite difference method

- Finite difference method converts BVP into system of algebraic equations by *replacing derivatives with finite difference approximations*
- Consider the BVP: $y'' = f(t, y, y') \qquad a < t < b$

with BC: $y(a) = \alpha, \qquad y(b) = \beta$

- **Step 1**: introduce mesh points between *a* and *b*: $t_i = a + ih$
- **Step 2**: replace the derivatives into a system of equations for unknowns y_i:

$$\frac{y_{i+1} - 2y_i + y_{i-1}}{h^2} = f(t_i, y_i, \frac{y_{i+1} - y_{i-1}}{2h})$$

- **Step 3**: solve for the unknowns $y_i(t_i)$ to obtain the approximate solution function at the mesh points

Example – finite difference method

- Let's solve $y'' = 6t$

  with BC: $y(0) = 1, \quad y(1) = 1$
- Replacing derivatives with finite difference approximations:

$$\frac{y_{i+1} - 2y_i + y_{i-1}}{h^2} = 6t_i$$

- Assume *h* = 0.5, we have three unknowns to solve between the boundaries: y_0, y_1 and y_2. We have one equation and two BCs:

$$\frac{1 - 2y_1 + 1}{(0.5)^2} = 6t_1 = 3$$

Finite difference method

- In practice, much smaller step sizes and many more mesh points would be required to obtain good approximations to the solution function
- In general, for *N* mesh points, we would have a system of *N-2* equations and 2 BCs to solve for $y(t_i)$, where $i = 1, 2 \ldots N$
- This system of equations is typically ***tridiagonal*** (involving only neighboring points) and relatively easy to solve
- **Example** -- using h = 0.2 we have:

$$\begin{bmatrix} -2 & 1 & 0 & 0 \\ 1 & -2 & 1 & 0 \\ 0 & 1 & -2 & 1 \\ 0 & 0 & 1 & -2 \end{bmatrix} \begin{bmatrix} y_4 \\ y_3 \\ y_2 \\ y_1 \end{bmatrix} = \begin{bmatrix} 4.8h^2 - 1 \\ 3.6h^2 \\ 2.4h^2 \\ 1.2h^2 - 1 \end{bmatrix}$$

- This is a linear system which we know how to solve for y_i

In-class exercise

Exercise #1 – solve the BVP using the shooting method

- Let's solve the BVP for the 2nd-order ODE: $y'' = 6t$ $\Rightarrow$ $\boxed{\begin{array}{l} y' = y_2 \\ y_2{}' = 6t \end{array}}$

$$\boldsymbol{y}'(t) = \begin{bmatrix} y_2 \\ 6t \end{bmatrix} = \boldsymbol{f}(t, \boldsymbol{y})$$

(this is the array `k(:)` defined in the `my_func` subroutine in the exercises)

- BC: $y(0) = 1, \qquad y(1) = 1$

- Start from ***initial guess $y_2(0)=1$***. Solve this IVP to get the value of y(1)
- ***Vary $y_2(0)$*** until y(1)=1

Exercise #1 – solve the BVP using the shooting method

- Connect to CICA and load the Fortran compiler:

```
module load pgi
```

- Go to your exercise directory on CICA:

```
cd astr660/exercise
```

- Copy the template Fortran files to the current directory:

```
cp -r /data/hyang/shared/astro660CompAstro/L8exercise ./
```

- Enter the directory and see what files are there:

```
cd L8exercise
ls
```

Exercise #1 – solve the BVP using the shooting method

- Take a look at the files and make sure you understand how they work
- Fill in subroutine `my_func` in `shooting1.f90`
- Copy over the `rk2` subroutine developed in the last lecture to `solvers.f90`
- Compile and run the codes by

```
make
```

```
./shooting1
```

- To get the bonus credit, please submit your code and screen shot of the output to the TAs by end of today (11/3/2022)

Exercise #2 – solve the BVP using the shooting+bisection method

- Copy shooting1.f90 to shooting2.f90
- Implement a bisection search for the best value of $y_2(0)$
- You could set the search region to be [-1.5, 2.0]
- Modify Makefile to include compilation of shooting2.f90
- Compile and run the codes by

```
make
./shooting2
```

- Observe how many iterations are required until the solution is found

References & acknowledgements

- Course materials of Computational Astrophysics from Prof. Kuo-Chuan Pan (NTHU)
- Course materials of Computational Astrophysics from Prof. Hsi-Yu Schive (NTU)
- Course materials of Computational Astrophysics and Cosmology from Prof. Paul Ricker (UIUC)
- “Computational Physics” by Rubin H. Landau, Manuel Jose Paez and Cristian C. Bordeianu
- “Scientific Computing – An Introductory Survey” by Michael T. Heath