Computational Physics – Lecture 13

(30 March 2020)

Housekeeping

Remember Zoom etiquette:

1. You are welcome to turn on your video, but keeping it off will conserve your bandwidth.
2. Mute your microphone unless you are speaking to me or the class.
3. There are two ways of saying something: (a) click on the "Raise Hand" button and start speaking when I address you, or (b) type a message in the chat window to the class or privately to me.
4. I am recording this lecture, so you can catch up if you end up having to leave due to a technical problem.

Where are we?

We have been studying numerical techniques to solve ordinary differential equations.

Where are we?

We have been studying numerical techniques to solve ordinary differential equations.

We have learnt a set of powerful techniques that enable us to solve almost any initial value problem on a computer.

Where are we?

We have been studying numerical techniques to solve ordinary differential equations.

We have learnt a set of powerful techniques that enable us to solve almost any initial value problem on a computer.

But we have exclusively focussed on *initial value problems* alone. In physics, many ordinary differential equations arise as *boundary value problems*.

Today's plan

In today's class, we want to learn how to solve boundary value problems using computers.

An example

Consider the boundary value problem

$$\frac{\mathrm{d}^2 x}{\mathrm{d}t^2} = -g, \tag{1}$$

where $g > 0$ is a constant and the boundary conditions are given as

$$x = 0 \text{ at } t = 0, \tag{2}$$

$$x = 0 \text{ at } t = t_1, \tag{3}$$

where $t_1 > 0$ is a constant.

Cannot solve this using our methods

Let's say we want to solve this problem using the Euler method. How do we proceed?

Cannot solve this using our methods

Let's say we want to solve this problem using the Euler method. How do we proceed?

As we have seen before, we begin by converting our second-order ODE to two simultaneous first-order ODEs, given by

$$\frac{\mathrm{d}v}{\mathrm{d}t} = -g, \text{ and} \tag{4}$$

$$\frac{\mathrm{d}x}{\mathrm{d}t} = v, \tag{5}$$

where we have introduced a new quantity v.

Cannot solve this using our methods

We can cast our system of equations

$$\frac{\mathrm{d}v}{\mathrm{d}t} = -g, \text{ and} \tag{6}$$
$$\frac{\mathrm{d}x}{\mathrm{d}t} = v, \tag{7}$$

as

$$\frac{\mathrm{d}\mathbf{r}}{\mathrm{d}t} = \mathbf{f}(\mathbf{r}, t), \tag{8}$$

where

$$\mathbf{r} = \begin{bmatrix} v \\ x \end{bmatrix}, \tag{9}$$

and

$$\mathbf{f}(\mathbf{r}, t) = \begin{bmatrix} -g \\ v(t) \end{bmatrix}. \tag{10}$$

Cannot solve this using our methods

Euler's method with step size h will now give us

$$\mathbf{w}_0 = ? \tag{11}$$

$$\mathbf{w}_{j+1} = \mathbf{w}_j + h \cdot \mathbf{f}(\mathbf{w}_j, t_j), \tag{12}$$

where we do not have the information that should replace the question mark.

Cannot solve this using our methods

Euler's method with step size h will now give us

$$\mathbf{w}_0 = ? \tag{11}$$

$$\mathbf{w}_{j+1} = \mathbf{w}_j + h \cdot \mathbf{f}(\mathbf{w}_j, t_j), \tag{12}$$

where we do not have the information that should replace the question mark.

So we cannot apply our methods to boundary value problems.

Cannot solve this using our methods

Euler's method with step size h will now give us

$$\mathbf{w}_0 = ? \tag{11}$$

$$\mathbf{w}_{j+1} = \mathbf{w}_j + h \cdot \mathbf{f}(\mathbf{w}_j, t_j), \tag{12}$$

where we do not have the information that should replace the question mark.

So we cannot apply our methods to boundary value problems.

This makes boundary value problems much harder to solve than initial value problems.

How to proceed?

There are various methods to solve boundary value problems.

How to proceed?

There are various methods to solve boundary value problems.

But these fall into just two classes:

- Shooting methods, and
- Relaxation methods.

How to proceed?

There are various methods to solve boundary value problems.

But these fall into just two classes:

- Shooting methods, and
- Relaxation methods.

Before learning some methods in detail, we should understand the idea behind these two types of method. Fortunately, this is quite straightforward.

How to proceed?

There are various methods to solve boundary value problems.

But these fall into just two classes:

- Shooting methods, and
- Relaxation methods.

Before learning some methods in detail, we should understand the idea behind these two types of method. Fortunately, this is quite straightforward.

Let us understand this in the context of our example problem.

Shooting methods of solving boundary value problems

In the case of our example problem, we happen to know the solution analytically. It is given by

$$x(t) = \frac{-gt^2}{2} + \frac{gt_1 t}{2}, \tag{13}$$

where g and t_1 are our given constants.

Shooting methods of solving boundary value problems

In the case of our example problem, we happen to know the solution analytically. It is given by

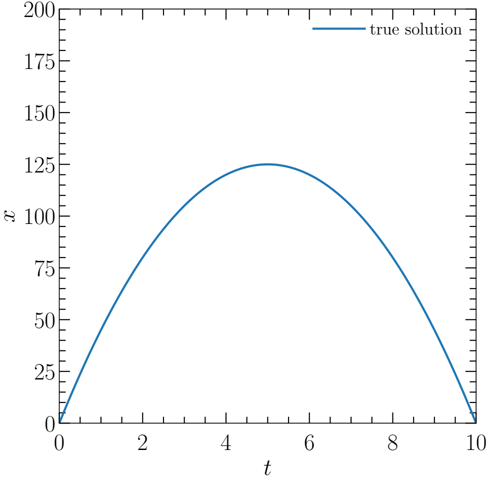
$$x(t) = \frac{-gt^2}{2} + \frac{gt_1 t}{2}, \tag{13}$$

where g and t_1 are our given constants.

(Notice that $\mathrm{d}x/\mathrm{d}t = -gt + gt_1/2$ so $\mathrm{d}^2x/\mathrm{d}t^2 = -g$. Also, $x(0) = 0$ and $x(t_1) = 0$. So this *is* our solution.)

Shooting methods of solving boundary value problems

Here is how the solution looks like for $g = 10$ and $t_1 = 10$:



Shooting methods of solving boundary value problems

We could have easily obtained this solution using Euler or Runge-Kutta methods only if we were given the initial value of v instead of a boundary value of x.

Shooting methods of solving boundary value problems

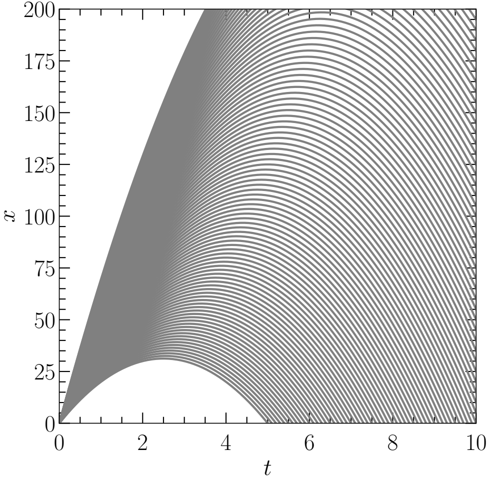
We could have easily obtained this solution using Euler or Runge-Kutta methods only if we were given the initial value of v instead of a boundary value of x.

So shooting methods say:

- just assume various initial values of v,
- solve these *initial value problems* using your usual methods, and then
- choose the solution that satisfies the boundary value of x.

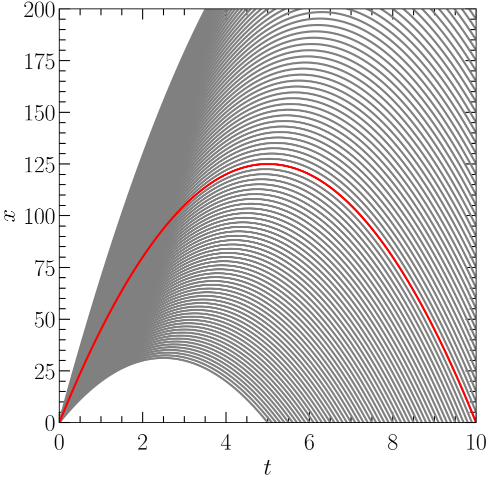
Shooting methods of solving boundary value problems

In our case, the initial value of x is specified. (It is 0.) If we choose a range of initial values on v and solve the resultant *initial value problem* using the fourth-order Runge-Kutta method, we get (for $g = 10$ and $t_1 = 10$)



Shooting methods of solving boundary value problems

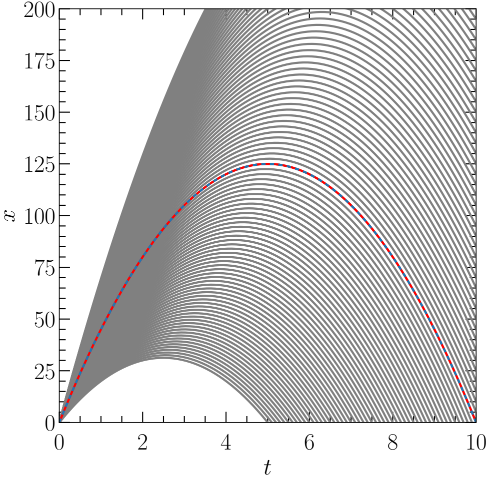
These solutions have different values at $t = t_1$. We now find the solution that is closest to our boundary value $x(t_1) = 0$. We get



(Red curve is the solution reported by the shooting method.)

Shooting methods of solving boundary value problems

Comparing with our true solution tells us that this is a good approximation:



(Red curve is the solution reported by the shooting method. Dashed blue curve is the true solution. They agree.)

Relaxation methods of solving boundary value problems

So shooting methods work by creating initial value problems and then choosing the one that gives the desired solution. The candidate solutions satisfy the ODE but do not necessarily satisty the boundary conditions.

Relaxation methods of solving boundary value problems

So shooting methods work by creating initial value problems and then choosing the one that gives the desired solution. The candidate solutions satisfy the ODE but do not necessarily satisty the boundary conditions.

How do relaxation methods work?

Relaxation methods of solving boundary value problems

So shooting methods work by creating initial value problems and then choosing the one that gives the desired solution. The candidate solutions satisfy the ODE but do not necessarily satisty the boundary conditions.

How do relaxation methods work?

Relaxation methods work by considering a range of curves that satisfy the boundary conditions and choose the curve that satisfies the ODE as our solution.

Relaxation methods of solving boundary value problems

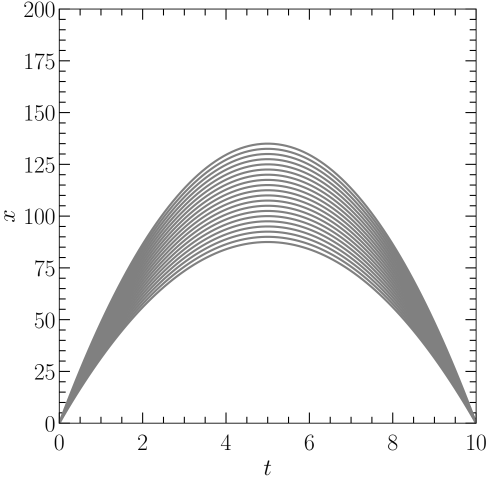
So shooting methods work by creating initial value problems and then choosing the one that gives the desired solution. The candidate solutions satisfy the ODE but do not necessarily satisty the boundary conditions.

How do relaxation methods work?

Relaxation methods work by considering a range of curves that satisfy the boundary conditions and choose the curve that satisfies the ODE as our solution. So the candidate solutions satisfy the boundary conditions but do not necessarily satisfy the ODE.

Relaxation methods of solving boundary value problems

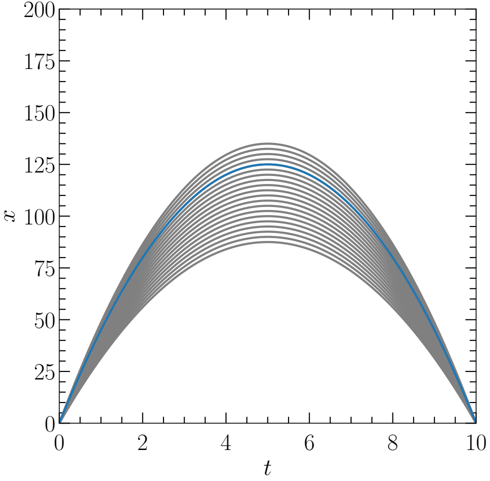
For example, here is a family of functions satisfying our given boundary conditions:



These functions do *not* satisfy our ODE!

Relaxation methods of solving boundary value problems

The function that is closest to satisfying our ODE is our result:



Relaxation methods of solving boundary value problems

The `solve_bvp` function in SciPy's `scipy.integrate` module solves boundary value problems using a type of relaxation method.

Shooting method

Consider the boundary value problem

$$y'' = f(x, y, y'), \tag{14}$$

with $a \geq x \geq b$, $y(a) = \alpha$, and $y(b) = \beta$.

Shooting method

Consider the boundary value problem

$$y'' = f(x, y, y'), \tag{14}$$

with $a \geq x \geq b$, $y(a) = \alpha$, and $y(b) = \beta$.

To solve this problem, we construct a family of initial value problems

$$y'' = f(x, y, y'), \tag{15}$$

with $a \geq x \geq b$, $y(a) = \alpha$, and $y'(a) = t$, and choose $t = t_k$ such that

$$\lim_{k \to \infty} y(b, t_k) = y(b) = \beta \tag{16}$$

Shooting method

How to we determine the t_k?

Shooting method

How to we determine the t_k?

Well these are just the sequence of values of t in an iterative solution to the nonlinear equation

$$y(b,t) - \beta = 0 \tag{17}$$

Shooting method

How to we determine the t_k?

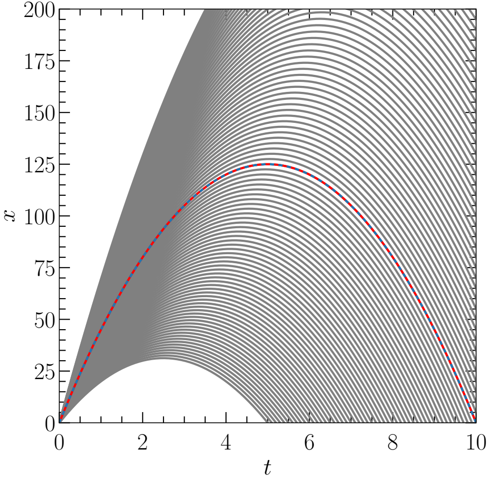
Well these are just the sequence of values of t in an iterative solution to the nonlinear equation

$$y(b, t) - \beta = 0 \tag{17}$$

Examples of such iterative procedures are the bisection method, the secant method, or the Newton-Raphson method.

Shooting method

The result will look something like:



Shooting method

Write a Python code to implement our shooting method for the boundary value problem we considered before:

$$\frac{\mathrm{d}^2 x}{\mathrm{d}t^2} = -g, \tag{18}$$

where $g = 10$ and the boundary conditions are given as

$$x = 0 \text{ at } t = 0, \tag{19}$$

$$x = 0 \text{ at } t = t_1, \tag{20}$$

where $t_1 = 10$. Make a figure to show your intermediate solutions, the final solution, and the exact solution.

Relaxation method

There is a variety of relaxation methods.

Relaxation method

There is a variety of relaxation methods.

Before we look at ODEs, consider the simple non-linear equation

$$x = 2 - e^{-x} \tag{21}$$

that we would like to solve.

Relaxation method

There is a variety of relaxation methods.

Before we look at ODEs, consider the simple non-linear equation

$$x = 2 - e^{-x} \tag{21}$$

that we would like to solve.

We could solve this using bisection method or Newton-Raphson method. But we can also solve this using a relaxation method.

Relaxation method

There is a variety of relaxation methods.

Before we look at ODEs, consider the simple non-linear equation

$$x = 2 - e^{-x} \tag{21}$$

that we would like to solve.

We could solve this using bisection method or Newton-Raphson method. But we can also solve this using a relaxation method.

How does that work?

Relaxation method

We start with a guess solution $x_0 = 1$ and plug it into the RHS of our equation to get

$$x_1 = 2 - e^{-1} \approx 1.632. \tag{22}$$

Relaxation method

We start with a guess solution $x_0 = 1$ and plug it into the RHS of our equation to get

$$x_1 = 2 - e^{-1} \approx 1.632. \tag{22}$$

Now we repeat the process and get

$$x_2 = 2 - e^{-1.632} \approx 1.804, \tag{23}$$

and so on to get a sequence of x's.

Relaxation method

If I quickly do a few more steps in Python, I get:

1.6321205588285577
1.8044854658474119
1.8354408939220457
1.8404568553435368
1.8412551139114340
1.8413817828128696
1.8414018735357267
1.8414050598547234
1.8414055651879888
1.8414056453310121
So we are converging to the true solution.

Relaxation method

What is happening here is that for a general equation $x = f(x)$ with a true solution x^*, if our guess is x then our updated value is

$$x' = f(x) = f(x^*) + (x - x^*)f'(x^*) + \dots . \tag{24}$$

Relaxation method

What is happening here is that for a general equation $x = f(x)$ with a true solution x^*, if our guess is x then our updated value is

$$x' = f(x) = f(x^*) + (x - x^*)f'(x^*) + \ldots. \tag{24}$$

But $x^* = f(x^*)$ by definition, so

$$x' - x^* = (x - x^*)f'(x^*), \tag{25}$$

and the relaxation will converge to x^* if $|f'(x^*) < 1|$.

Relaxation method

What is happening here is that for a general equation $x = f(x)$ with a true solution x^*, if our guess is x then our updated value is

$$x' = f(x) = f(x^*) + (x - x^*)f'(x^*) + \ldots. \tag{24}$$

But $x^* = f(x^*)$ by definition, so

$$x' - x^* = (x - x^*)f'(x^*), \tag{25}$$

and the relaxation will converge to x^* if $|f'(x^*) < 1|$.

We have also used similar relaxation ideas in linear algebra in the context of the Jacobi and Gauss-Seidel techniques. (There, we also manipulated the relaxation concept to achieve over-relaxation.)

Relaxation method

Now, suppose we have a boundary value problem

$$y'' = f(x, y, y') \text{ for } a \geq x \geq b, \tag{26}$$

with $y(a) = \alpha$ and $y(b) = \beta$.

Relaxation method

Now, suppose we have a boundary value problem

$$y'' = f(x, y, y') \text{ for } a \geq x \geq b, \tag{26}$$

with $y(a) = \alpha$ and $y(b) = \beta$.

We then we begin by setting up mesh points $x_i = a + ih$ in our domain $[a, b]$ with step-size h and $i = 1, 2, \ldots, N + 1$.

Relaxation method

Now, suppose we have a boundary value problem

$$y'' = f(x, y, y') \text{ for } a \geq x \geq b, \tag{26}$$

with $y(a) = \alpha$ and $y(b) = \beta$.

We then we begin by setting up mesh points $x_i = a + ih$ in our domain $[a, b]$ with step-size h and $i = 1, 2, \ldots, N + 1$.

We then discretise the derivatives to write

$$\begin{aligned} &\frac{y(x_{i+1}) - 2y(x_i) + y(x_{i-1})}{h^2} + \mathcal{O}(h^4) \\ &\qquad = f\left(x_i, y(x_i), \frac{y(x_{i+1}) - y(x_{i-1})}{2h} + \mathcal{O}(h^2)\right) \end{aligned} \tag{27}$$

Relaxation method

Now we can forget the higher-order terms and write $w_0 = \alpha$, $w_{N+1} = \beta$, and

$$\frac{w_{i+1} - 2w_i + w_{i-1}}{h^2} - f\left(x_i, w_i, \frac{w_{i+1} - w_{i-1}}{2h}\right) = 0. \qquad (28)$$

Relaxation method

Now we can forget the higher-order terms and write $w_0 = \alpha$, $w_{N+1} = \beta$, and

$$\frac{w_{i+1} - 2w_i + w_{i-1}}{h^2} - f\left(x_i, w_i, \frac{w_{i+1} - w_{i-1}}{2h}\right) = 0. \qquad (28)$$

These are now N coupled non-linear equations that we can solve using the kind of relaxation method we saw above for $x = 2 - e^{-x}$.

Relaxation method

Now we can forget the higher-order terms and write $w_0 = \alpha$, $w_{N+1} = \beta$, and

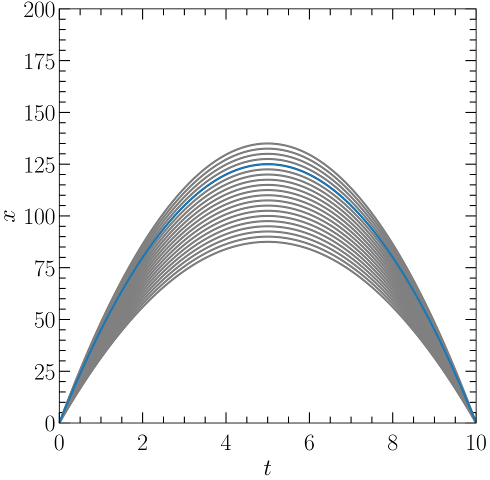
$$\frac{w_{i+1} - 2w_i + w_{i-1}}{h^2} - f\left(x_i, w_i, \frac{w_{i+1} - w_{i-1}}{2h}\right) = 0. \tag{28}$$

These are now N coupled non-linear equations that we can solve using the kind of relaxation method we saw above for $x = 2 - e^{-x}$.

If f is linear, this is a system of linear equations that you can solve using the Jacobi or Gauss-Seidel methods.

Relaxation method

This will result in a sequence of solutions that will converge to the true solution:



Relaxation method

You can implement a relaxation method in your assignment. For the moment, try solving our projectile motion problem using `solve_bvp`.

ODE Wrap-up

We have studied numerical techniques to solve ordinary differential equations.

ODE Wrap-up

We have studied numerical techniques to solve ordinary differential equations.

Here is what we have learnt:

1. Euler's Method
2. Error analysis and optimum step size for Euler's Method
3. Taylor and Runge-Kutta Methods
4. ODEs over infinite domains
5. Simultaneous differential equations
6. Higher-order differential equations
7. Adaptive step-size control
8. Backward integration for stiff problems
9. Boundary value problems

ODE Wrap-up

There are other methods available for solving ODEs.

ODE Wrap-up

There are other methods available for solving ODEs.

For example:

- Leapfrog method
- Verlet method
- Modified Midpoint method
- Bulirsch-Stoer method

ODE Wrap-up

There are other methods available for solving ODEs.

For example:

- Leapfrog method
- Verlet method
- Modified Midpoint method
- Bulirsch-Stoer method

But these methods can be often complex and unnecessary. You can go far with what we learnt.

ODE Wrap-up

There are other methods available for solving ODEs.

For example:

- Leapfrog method
- Verlet method
- Modified Midpoint method
- Bulirsch-Stoer method

But these methods can be often complex and unnecessary. You can go far with what we learnt.

Still: Keep an open mind. Keep adding methods to your toolbox. Use good libraries.