Spring 2021
PHYS 377 Advanced Computational Physics
HW # 7b

Problem 1 (100 points): The relaxation method for ordinary differential equations

There is no reason why the relaxation method must be restricted to the solution of differential equations with two or more independent variables. It can also be applied to those with one independent variable, *i.e.*, to ordinary differential equations. In this context, as with partial differential equations, it is a technique for solving boundary value problems, which are less common with ordinary differential equations but do occur.

Consider the example problem we looked at in **Lecture 5**, in which a ball of mass m = 1 kg is thrown from height x = 0 into the air and lands back at x = 0 ten seconds later. The problem is to calculate the trajectory of the ball, but we cannot do it using initial value methods like the ordinary Runge–Kutta method because we are not told the initial velocity of the ball. One approach to finding a solution is the shooting method. Another is the relaxation method.

Ignoring friction effects, the trajectory is the solution of the ordinary differential equation

$$\frac{d^2x}{dt^2} = -g$$

where g is the acceleration due to gravity.

(a) Replacing the second derivative in this equation with its finite-difference approximation (as given below), **derive a relaxation-method** equation for solving this problem on a time-like "***grid***" of points with separation h.

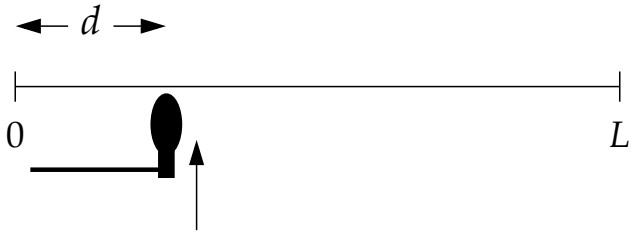
$$f''(x) = \frac{f(x+h) - 2f(x) + f(x-h)}{h^2}$$

(b) Taking the **boundary conditions** to be that x = 0 at t = 0 and t = 10, **write a program** to solve for the height of the ball as a function of time using the **relaxation method** with 100 points and make a plot of the result from t = 0 to t = 10. Run the relaxation method until the answers change by 10^{-6} or less at every point on each step.

Note that, unlike the shooting method, the relaxation method does not give us the initial value of the velocity needed to achieve the required solution. It gives us only the solution itself, although one could get an approximation to the initial velocity by calculating a numerical derivative of the solution at time t = 0. On balance, however, the relaxation method for ordinary differential equations is most useful when one wants to know the details of the solution itself, but not the initial conditions needed to achieve it.

Extra Credit Problem 1 (50 points): FTCS solution of the wave equation

Consider a piano string of length L, initially at rest. At time $t = 0$ the string is struck by the piano hammer a distance d from the end of the string:



The string vibrates as a result of being struck, except at the ends, $x = 0$ and $x = L$, where it is held fixed.

(a) Write a program that uses the **FTCS method** to solve the complete set of *simultaneous first-order equations*,

$$\frac{d\phi}{dt} = \psi(x,t), \qquad \frac{d\psi}{dt} = \frac{v^2}{a^2}[\phi(x+a,t) + \phi(x-a,t) - 2\phi(x,t)]$$

for the case v = 100 ms^{-1}, with the initial condition that $\phi(x) = 0$ everywhere but the velocity $\psi(x)$ is nonzero, with profile

$$\psi(x) = C\frac{x(L-x)}{L^2}exp\left[-\frac{(x-d)^2}{2\sigma^2}\right]$$

where L = 1 m, d = 10 cm, C = 1 ms^{-1}, and σ = 0.3 m. You will also need to choose a value for the time-step h. A reasonable choice is $h = 10^{-6}$ s.

(b) Make an animation of the motion of the piano string using the facilities provided by the **visual** package. There are various ways you could do this. A simple one would be to just place a small sphere at the location of each grid point on the string. A more sophisticated approach would be to use the curve object in the **visual** package—see the on-line documentation at www.vpython.org for details. A convenient feature of the curve object is that you can specify its set of x positions and y positions separately as arrays. In this exercise the x positions only need to specified once, since they never change, while the y positions will need to be specified anew each time you take a time-step. Also, since the vertical displacement of the string is much less than its horizontal length, you will probably need to multiply the vertical displacement by a fairly large factor to make it visible on the screen.

Allow your animation to run for some time, until *numerical instabilities* start to appear.