

Due Thursday, May 27th at 11:59pm

PROBLEM 1

The FitzHugh-Nagumo model is a system of ordinary differential equations used to describe the excitation of a neuron membrane:

$$\begin{aligned}\dot{v} &= v - \frac{1}{3}v^3 - w + I(t), \\ \dot{w} &= \frac{a + v - bw}{\tau}.\end{aligned}$$

In this model, $v(t)$ is the membrane voltage at time t and $w(t)$ is a variable representing the activity of several types of membrane channel proteins. The function $I(t)$ represents an external electrical current, and the parameters a , b and τ are constants controlling the channel protein activity.

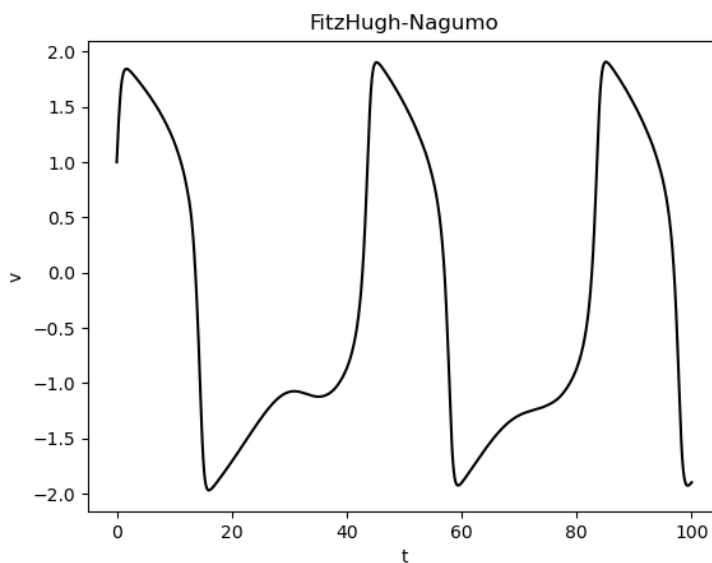
In this problem, we will assume that $a = 0.7$, $b = 1$ and $\tau = 12$ and that

$$I(t) = \frac{1}{10} \left(5 + \sin \left(\frac{\pi t}{10} \right) \right).$$

Throughout this problem, we will assume that the initial voltage is 1 and the initial channel activity is 0. That is,

$$v(0) = 1 \text{ and } w(0) = 0.$$

You should find that the voltage looks approximately like this:



Notice that this solution is roughly periodic. It is very important in neuroscience to be able to predict the firing rate of a neuron. The firing rate is

$$\text{firing rate} = \frac{1}{\text{the firing period}},$$

where the firing period is the distance between two adjacent peaks. In this problem, we will estimate the period by finding the distance between the first two peaks in our approximate solutions. From the diagram above, you can see that the firing period is roughly 45, and so the firing rate is roughly $1/45 \approx 0.02$.

Use `ode45` (for MATLAB) or `solve_ivp` (for Python) to solve the system described above from time $t = 0$ to $t = 100$ with $\Delta t = 0.5$. Save your approximations for the voltage at each time in a 201×1 column vector named `A1`.

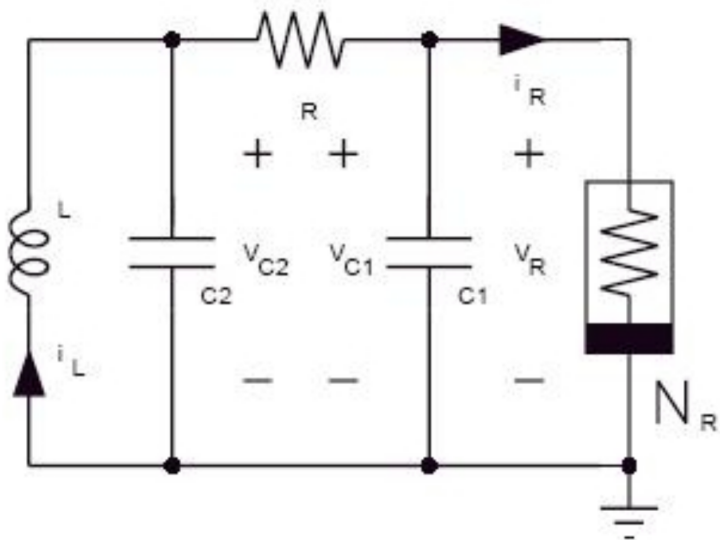
Find the time when the voltage that you saved as **A1** first reaches a maximum. (You can see from the diagram above that this time must be somewhere between 0 and 10.) Save this time in a variable named **A2**.

Find the time when the voltage reaches a the second peak. (You can see from the diagram above that this time must be somewhere between 40 and 50.) Save this time in a variable named **A3**.

Use your previous two answers to calculate the firing rate and save your answer in a variable named **A4**.

PROBLEM 2

Consider Chua's circuit



This circuit can be chaotic for the correct parameters. We can model this using Kirchoff's laws. After nondimensionalizing the model, it simplifies to

$$\begin{aligned}\dot{x} &= \alpha(y + \frac{1}{6}x - \frac{1}{16}x^3) \\ \dot{y} &= x - y + z \\ \dot{z} &= -\beta y\end{aligned}\tag{1}$$

- (1) Use `ode45` (for MATLAB) or `solve_ivp` (for Python) to solve the system described above with parameters `alpha = 16` and `beta = 30`, and initial condition $x(0) = 0.1$, $y(0) = 0.2$, $z(0) = 0.3$ from time $t = 0$ to $t = 100$ with $\Delta t = 0.05$. First plot the phase space and comment on whether or not it is chaotic; if it is not chaotic save it as **A5** = 0 and if it is chaotic save it as **A5** = 1. Save your approximations for the entire solution at each time in a 3×2001 matrix named **A6**.
- (2) Repeat the process with $x(0) = 0.1$, $y(0) = 0.2 + 10^{-5}$, $z(0) = 0.3$. Calculate the maximum difference in absolute value between **A6** and the new solution. Since we have a matrix, you have to do `max` twice: `max(max(abs()))` Save this value as **A7**.
- (3) Repeat the process with `beta = 100` and initial condition $x(0) = 0.1$, $y(0) = 0.2$, $z(0) = 0.3$. Plot the phase space and comment on whether or not it is chaotic; if it is not chaotic save it as **A8** = 0 and if it is chaotic save it as **A8** = 1.

PROBLEM 3

Consider the boundary value problem

$$\ddot{x} + x = 5 \cos(4t), \quad \text{with } x(0) = 1 \quad \text{and} \quad x(6) = 0.5.$$

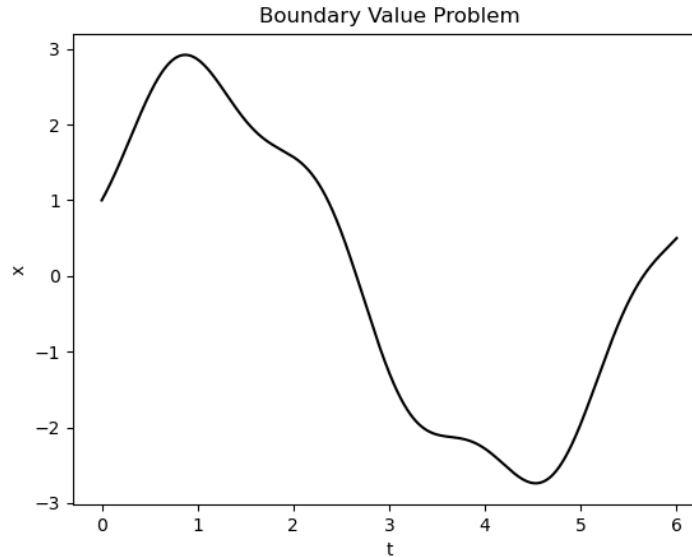
The true solution to this boundary value problem is

$$x(t) = C_1 \sin(t) + C_2 \cos(t) - \frac{1}{3} \cos(4t),$$

where

$$C_1 = \frac{\frac{1}{2} + \frac{1}{3} \cos(24) - \frac{4}{3} \cos(6)}{\sin(6)} \quad \text{and} \quad C_2 = \frac{4}{3}.$$

The solution looks like this:



- (1) Following the finite difference process we used in lecture, use a second order difference scheme to approximate \ddot{x} and rewrite this initial value problem as a linear system of equations $A\mathbf{x} = \mathbf{b}$. Use $\Delta t = 0.1$. Note that there will be 61 total t -values, but we are only trying to approximate x at the *interior times*. The matrix A should be $N_{int} \times N_{int}$, where N_{int} is the number of interior times, and the vector \mathbf{b} should be $N_{int} \times 1$. Save a copy of A in a variable named **A9**. Save a copy of \mathbf{b} in a variable named **A10**.

Solve this linear system using Gaussian elimination (the backslash operator in MATLAB or the `solve` function in python). Make a 61×1 column vector containing the x values at *all times* (including the two boundary values) and save a copy of this vector in a variable named **A11**.

Find the maximum (in absolute value) error between this approximation and the true solution. Save this error in a variable named **A12**.

- (2) Now solve the problem using the shooting method. Remember, you need to convert this second order ODE into a system of two first order ODEs. Apply a bisection scheme to the usual IVP solvers `ode45` (MATLAB) or `solve_ivp` (Python) using $\Delta t = 0.1$. For the second initial conditions, you can use the plot to help you test out values that will get you on different sides of the right boundary condition. Break out of the loop for bisection when the difference between your approximation at the right boundary point and the exact value at the right boundary point is less than $1\text{e-}8$; i.e., $|x_{\text{approx}}(6) - x(6)| = |x_{\text{approx}}(6) - 0.5| < 10^{-8}$. Save the numerical solution as a 61×1 column vector **A13** and the maximum (in absolute value) error $\max |x_{\text{approx}} - x|$ as **A14**. Further save the maximum difference in absolute value between the solution from part 1 and this solution as **A15** = `max(abs(A13 - A11))`.