SESSION 1: SOLVING DIFFERENTIAL EQUATIONS ON A COMPUTER

This practical session aims to introduce you to simulating brain dynamics on a computer. The practical is split into three parts, which take you through the steps required to simulate cortical gamma oscillations using the Wilson-Cowan model. The parts are outlined below:

1. The Euler method: You will implement a dynamical systems solver on a computer
2. The Wilson-Cowan model: You will implement the Wilson-Cowan model + simulate using the Euler method, to observe the generation of gamma rhythms.
3. Stochastic differential equations: You will learn how to add noise to dynamical systems, and study noise-induced gamma oscillations in your Wilson-Cowan model.

Part 1: The Euler Method

In this section you will implement an "Euler solver", which is an algorithm to solve differential equations on a computer. You will use this solver to simulate the one population firing rate equation. Because this equation has an exact solution, you will be able to compare the Euler solution to the real solution, and see errors in the Euler solution. The algorithm, using python syntax, is outlined below.

Algorithm: The Euler Method

1. Make an array of t time points at which you wish to solve and initial conditions x0, where x0 are the values of x at time t(1).
2. Initialize an array x with size [N,T] where N is the dimensionality of the system and T is the number of time points.
3. Set x(:,1)=x0
4. Loop over i in the range 1 to T-1, finding h=t(i+1)-t(i) and then solving x(:,i+1)=x(:,i)+h*f(x(:,i))

Tasks

Implement a function x=EulerODE(t,x0,f)which takes as input an array of time points (t: size 1xT), an array of initial conditions (x0: size Nx1) and an ODE function handle f, which outputs the solution to the ODE solved using the Euler method. It should perform steps 2-4 of the algorithm above (step 1 are the inputs to the function).

2. Runing the script, to check your

EulerODE function works. The black line (Euler solution) and dotted blue line (real solution) should be very similar.

3. Do you think the Euler solver will be more accurate with smaller or larger time steps? Vary the time step h to test your hypothesis. Do you notice any downsides to smaller time steps?

Part 2: The Wilson-Cowan Equations

Parts 2-3 of this workshop focus on simulating gamma oscillations in the cortex using the Wilson Cowan model

Parameter	Value	Parameter	Value	Meaning
τ_E	3.2	τ_I	3.2	Population time constants
c_{EE}	2.4	c_{IE}	2	Connectivity onto *E* population
c_{EI}	2	c_{II}	0	Connectivity onto *I* population
k_E	4	k_I	4	Slope of sigmoid
θ_E	1	θ_I	1	"Threshold" (input at which firing rate = 0.5)

Tasks

4. Make a function xdot=WilsonCowan(x,P), which takes in a 2x1 matrix x=[E;I] and a value P (the input current) and outputs a vector of derivatives xdot=[Edot;Idot]. The function FiringRateModel() is a good start point, as the Wilson-Cowan model is a two population version of the firing rate model.

5. Run the script SimulateWilsonCowan() to simulate the Wilson-Cowan system. This script uses your EulerODE function. You should find a steady state at x=[0.0181;0.0207].

6. Increase the value of P (input to the system) on line 8 for values between 0 and 2 and run the script again. What happens as this input to the excitatory population increases?

7. Set P=0.5. You should see a gamma frequency (~55 Hz) oscillation. Set cEI = 0 and simulate. Does the system still oscillate? Now return cEI to it's original value and do the same for cIE = 0. Can you think of an explanation for these results?

Remember to return these values to their original values for the next tasks.

Part 3: Stochastic differential equations (SDEs)

In nature, systems often have inherent or extrinsic sources of noise. A noisy ODE is known as a stochastic differential equation (SDE).

8. Implement a function x=EulerSDE(t,x0,f,stdnoise)which, in addition to the same inputs as EulerODE, takes in the standard deviation of noise (stdnoise: size Nx1) and outputs the solution to an SDE solved using the stochastic Euler method. The file EulerODE is a good start point.

9. Now solve the equation using EulerSDE instead of EulerODE. By default, I have set the standard deviation of noise for both populations to zero. Run the script keeping these defaults, and check you get the same results as when EulerODE was used.

10. Set P=0.39, and simulate. The system should be at a steady state. Now set stdEnoise=0.01 on line 15, and simulate again. What do you notice? Is this oscillation also at a gamma frequency (around 30-100 Hz)?

What happens as you decrease P? Why do you think you see these results?