Practical session 2: Modelling the emergence of seizures in networks

This practical session aims to introduce you to a phenomenological model of seizure transitions, the *theta model* [1]. The theta model is a phase oscillator model, i.e. it describes the dynamics of an oscillator. A single oscillator may be used to represent the brain activity at one brain region. A network of oscillators may then represent the dynamics of a brain network. In this model, large oscillations represent seizure activity, whereas low amplitude fluctuations correspond to a 'normal' brain activity.

This worksheet is split into three parts:

1. Simulate a deterministic phase oscillator
2. Simulate a stochastic phase oscillator
3. Simulate a network of interacting phase oscillators

It is strongly advised to first complete the practical session 1:
https://github.com/lukewtait/intro_to_modelling/blob/master/practical1/Worksheet1.pdf

In particular, in this practical session you will be using the functions `EulerODE.m` and `EulerSDE.m`, which were introduced and developed in the first practical session. **It is important that you familiarize with these functions, so that you can use them below.** They will allow you to solve ordinary and stochastic differential equations.

For a guide on MatLab syntax to help with practical session 1:
https://github.com/lukewtait/intro_to_modelling/blob/master/practical1/SyntaxGuide.pdf

If you have any questions about either practical session 1 or 2, ask here:
https://brainmodelworkshop.freeforums.net (no registration required)

If you missed the first tutorial on "An Introduction to Dynamical Systems", you can watch here: https://www.youtube.com/watch?v=ZxXE2eQ7Y38

If you would like to know more about the theta model in the context of epilepsy:
[1] Lopes, M.A., et al. (2017). PLoS CB, 13(8). https://doi.org/10.1371/journal.pcbi.1005637
[2] Lopes, M.A., et al. (2018). Front Neurol, 9, 98. https://www.frontiersin.org/articles/10.3389/fneur.2018.00098/full
[3] Lopes, M. A., et al. (2019). Front Comput Neurosci, 13, 25. https://doi.org/10.3389/fncom.2019.00025
[4] Junges, L., et al. (2019). Sci Rep, 9(1), 1-12. https://www.nature.com/articles/s41598-019-43871-7
[5] Lopes, M. A., et al. (2019). Sci Rep, 9(1), 1-10. https://www.nature.com/articles/s41598-019-46633-7
[6] Laiou, P., et al. (2019). Front Neurol, 10, 1045. https://www.frontiersin.org/articles/10.3389/fneur.2019.01045/full
[7] Słowiński, P., et al. (2019). eNeuro, 6(4). https://www.eneuro.org/content/6/4/ENEURO.0059-19.2019
[8] Lopes, M. A., et al. (2020). Clin Neurophysiol, 131(1), 225-234. https://doi.org/10.1016/j.clinph.2019.10.027
[9] Lopes, M. A., et al. (2020). Front Neurol, 11, 74. https://www.frontiersin.org/articles/10.3389/fneur.2020.00074/full
[10] Lopes, M.A., et al., (2020). https://www.medrxiv.org/content/10.1101/2020.05.18.20102681v1

Part 1: Simulate a deterministic phase oscillator

In this section you are going to implement the theta model in its simplest form, i.e. a single ordinary differential equation (ODE) that describes the dynamics of a phase oscillator θ:

$$\dot{\theta} = (1 - \cos\theta) + (1 + \cos\theta)I \qquad (1)$$

where I is an input current to the oscillator. As learnt in the first practical session, an ODE can be solved using Euler's Method (find `EulerODE.m` in github). It will be assumed below that you are going to use this function, but feel free to implement it again. Also, do not forget to define MatLab's directory to where you have this function placed.

Tasks

1. Open the script `practical2_part1.m` (available in github).
   To run line 13, `theta=EulerODE(time,theta_0,@(theta) ThetaModel(theta,I))`, you need to implement the function `ThetaModel`. This function corresponds to the right-hand side of Eq. (1) above.

Hint: This task is similar to task 4 in part 2 in the first practical session. Instead of changing the `FiringRateModel` into the `WilsonCowan`, here you change it into the `ThetaModel`. In other words, you can use the `FiringRateModel` as your starting point and change the function so that it computes the theta model instead.

2. To observe the output, run the script `practical2_part1.m`. The output corresponds to a simple transformation of θ into $1 - \cos\theta$ (this transformation is convenient to make clear how the time dependence of θ may translate into oscillations). Play with different values of the input current I (e.g. $I = -0.5$, $I = 0.1$, and $I = 0.5$).

   What is the condition to observe oscillations? How does the frequency of oscillations depend on the input current?

3. As in the first practical session (part 1, task 3), observe what happens as the time step is varied. Assess the computational time and accuracy of your numerical simulations.

Hint: uncomment lines 25 to 31 in the script `practical2_part1.m` to compare your numerical solution (Euler's method) to the exact solution.

4. For negative input current (e.g. $I = -0.1$), assess the role of the initial condition (`theta_0`) by changing it. What do you observe and why? Note that since θ is a phase, you can simply assess the interval from 0 to 2π.

Part 2: Simulate a stochastic phase oscillator

Now let us consider a modification to Eq. (1) so that the phase oscillator becomes stochastic. One way of doing this is to simply add a noise term ϵ:

$$\dot{\theta} = (1 - \cos\theta) + (1 + \cos\theta)I + \sigma\epsilon, \qquad (2)$$

where ϵ is normally distributed, with zero mean and standard deviation σ.

Side note: This stochastic implementation of the theta model differs a little bit from the one mentioned in the presentation and in the papers [1-10] referred above. The difference is that here we consider *additive noise* for simplicity (i.e. we add a noise term), whereas in the presentation the noise was introduced in definition of the current I, i.e. $I = I_0 + \sigma\epsilon$. Since I is multiplying by $(1 + \cos\theta)$, this noise is called *multiplicative noise*. The difference is that the additive noise produces a perturbation on the phase θ which is independent of the specific value of θ at any given time, whereas the effect of the multiplicative noise on the dynamics depends on θ. For example, if at time t_1 we have $1 + \cos\theta(t_1) = 0$, then the multiplicative noise at this moment in time won't affect the phase, because the whole term $(1 + \cos\theta\,(t_1))I$ is zero, whereas the additive noise would affect the phase.

Tasks

5. Modify the script `practical2_part1.m` (copy it and change its name to `practical2_part2.m`) to simulate the stochastic theta model. Use the function `EulerSDE` developed in the first practical session (part 3, task 8) to integrate the theta model. Consider $\sigma = 0.5$.

Hint: You will have to change line 13 to call `EulerSDE` instead of `EulerODE`. Note also that `EulerSDE` has one additional fourth input relative to `EulerODE`, which is the standard deviation of the noise. Note that now lines 25-31 do not provide an exact solution for the stochastic theta model, though the comparison may still be useful in this task and in the next.

6. Play with the values of I (and optionally with σ). How does the addition of noise change the results observed in task 2?

   Does a given level of noise σ produce similar effects on the steady state at different values of I (e.g. $I = -0.5$ and $I = -0.1$)? Why? What about in the oscillatory regime (e.g. $I = 0.1$ and $I = 0.5$)??

Hint 1: Note that each time you run your script, you get a new noise realisation, and hence a different time series. Run several times the script for the same parameters to obtain intuition. It may also be helpful to increase the number of time steps to consider longer signals (e.g. define `time = 0:dt:300;`).

Hint 2: To build intuition, you may start with $\sigma = 0$ so that you recover the deterministic dynamics. Then increase σ in increments of 0.1 up to 0.5, to see how the dynamics are changing.

Part 3: Simulate a network of interacting phase oscillators

Now let us generalize Eq. (2) to N interacting phase oscillators:

$$\dot{\theta}_i = (1 - \cos\theta_i) + (1 + \cos\theta_i)I + \sigma\epsilon_i + \frac{K}{N}\sum_{j\neq i} a_{ji}\left[1 - \cos\theta_j\right]. \qquad (3)$$

We have a stochastic differential equation (SDE) for each oscillator θ_i, where $i = 1, \ldots, N$. The new term on the right accounts for the interaction with other oscillators. K is a global scaling coupling, a convenient parameter to define how strong the interactions are (at $K = 0$ there is no interaction and each oscillator behaves independently from the others). a_{ji} defines whether an oscillator j is connected to i ($a_{ji} = 1$ if there is a connection from j to i, otherwise $a_{ji} = 0$). The type of interaction is given by the multiplicative factor $[1 - \cos\theta_j]$. Depending on its phase, oscillator j may have a stronger or weaker effect on oscillator i (e.g. if the oscillator θ_j is at a steady state $\theta_j = 0$, then its impact on other oscillators is null; if instead it is oscillating, it will produce an oscillating input to other oscillators). The sum over all j different from i ensures that we consider all oscillators j that are connected to i.

Note: As in Part 2, this implementation of the theta model differs a bit from that mentioned in the presentation and papers [1-10]. Namely, both the noise and interaction may be included in the current I. For our purposes, this difference is not relevant, though the model output is certainly different.

Tasks

7. Given that `EulerSDE` is already set up to deal with N differential equations (see the first practical section, part 3), all you have to do is to update the function `ThetaModel` so that it works for N oscillators and accounts for their interactions following Eq. (3). Copy the m-file `ThetaModel.m`, name it `ThetaModel_N.m`, and modify it.

Hints: you need to add additional inputs to the function, namely K and the adjacency matrix $\boldsymbol{A} = (a_{ji})$, i.e. the first line of your function should look like:

```
theta_dot = ThetaModel_N(theta,I,K,A)
```

You will also need to define N, which is the dimension of the vector $\boldsymbol{\theta} = (\theta_1, \theta_2, \ldots, \theta_N)$. In other words, the input `theta` is now an array of length `N`, where the `i`'th element of the array is the phase of oscillator `i` (e.g. `theta(1)` is the phase of oscillator `1`). Since N shows up in Eq. (3), you need to calculate it inside this function.

Finally, to add the interaction term, you may either do it by using a `for` cycle or a multiplication of the matrix ($\boldsymbol{A}$) by the vector ($1 - \cos\boldsymbol{\theta}$), i.e. `A*(1-cos(theta))`. Note that the second alternative needs to make sure that the diagonal of matrix $\boldsymbol{A}$ is zero.

The noise term $\sigma\epsilon_i$ is handled by `EulerSDE`, so it should not be included in `ThetaModel_N`.

8. Use the script `practical2_part3.m` to check whether your function is working. By default, the script uses a network with 5 nodes which are all interconnected (this is called a fully connected network or all-to-all network), $K = 10$, $\sigma = 0.1$, and $I = -1$. Note that this script uses the function `plot_signals.m` to plot the results, so make sure this function is in your MatLab directory (you can find the function in github).

   Run the script several times – different noise realisations will give you different results. Compare the network dynamics when using $K = 10$ with using $K = 1$ and $K = 20$. How do you interpret the results?

9. Set again $K = 10$ (and all parameters to the default values as in the previous task). Increase the size of the network N (for example $N = 10$). What do you observe and why?

10. Set $I = -1.2$, and all other parameters to the default values as in task 8. Replace the fully connected network by a random network. If these two networks would represent a brain network from a healthy individual and a brain network from someone with epilepsy, which one would be the healthy and which one would be the epileptic? Why?

Hint 1: To build a random network you may use the function `rand`, and then set values below 0.5 to 1, and values above 0.5 to 0.

Hint 2: Spiky activity represents seizure activity.