Tutorial on Turing-like bifurcations for neural fields

Daniele Avitabile*

Abstract. In this tutorial we discuss a basic mechanism for the formation of spatial patterns from homogeneous steady states in cortical models knows as Neural Field Equations. We use a mix of formal nonlinear analysis and numerical simulations. Before starting this tutorial download the latest version of the accompanying dataset. Solutions to the exercises can be browsed here and code can be downloaded from our repository.

1. Introduction.

We will study the emergence of spatially-periodic stationary solutions in the following Neural Field Equation (NFE)

$$\partial_t u(x,t) = -u(x,t) + \int_{\mathbb{R}} w(x-y) f(u(y,t))\, dy \qquad (x,t) \in \mathbb{R} \times \mathbb{R}_{>0}. \tag{1.1}$$

Recall that for functions $u \in L^1(\mathbb{R})$, the space of integrable functions on $\mathbb{R}$, we define its Fourier Transform as

$$\hat{u}(\xi) = \int_{\mathbb{R}} u(x) e^{-i\xi x}\, dx, \qquad \xi \in \mathbb{R}.$$

We henceforth assume that $w \in L^1(\mathbb{R})$.

2. Tutorial questions - nonlinear analysis.

Question 1. Let u_* be a spatially homogeneous equilibrium of the NFE (1.1). Setting $u(x,t) = u_* + v(x,t)$, and using a Taylor expansion of f show formally that small perturbations $v(x,t)$ to the homogeneous steady state u_* evolve according to the linear integro-differential equation

$$\partial_t v(x,t) = -v(x,t) + f'(u_*) \int_{\mathbb{R}} w(x-y) v(y,t)\, dy \qquad (x,t) \in \mathbb{R} \times \mathbb{R}_{>0} \tag{2.1}$$

Question 2. Assuming w is an even function, show that (2.1) admits solutions of the form $v(x,t) = e^{\lambda t} e^{i\xi x}$ for any (ξ, λ) satisfying the *dispersion relation*

$$\lambda(\xi) = -1 + f'(u_*)\hat{w}(\xi), \qquad \xi \in \mathbb{R},$$

and deduce that if $\lambda(\xi) < 0$ for all $\xi \in \mathbb{R}$, then u_* is linearly stable to perturbations $v(x,0) = e^{i\xi x}$.

Question 3. Consider the NFE (1.1) with synaptic kernel and firing rate given by

$$w(x) = AW(x), \qquad W(x) = \frac{1}{\sqrt{\pi}} e^{-x^2} - \frac{1}{\sigma\sqrt{\pi}} e^{-x^2/\sigma^2}, \qquad f(u) = \frac{1}{1+e^{-\mu u + \theta}} - \frac{1}{1+e^{\theta}}, \tag{2.2}$$

*Vrije Universiteit Amsterdam, Department of Mathematics, Faculteit der Exacte Wetenschappen, De Boelelaan 1081a, 1081 HV Amsterdam, The Netherlands. MathNeuro Team, Inria branch of the University of Montpellier, 860 rue Saint-Priest 34095 Montpellier Cedex 5 France. (d.avitabile@vu.nl, www.danieleavitabile.com).

respectively, where $A \in \mathbb{R}_{\geq 0}$, $\sigma \in \mathbb{R}_{>1}$, $\mu \in \mathbb{R}_{>0}$ and $\theta \in \mathbb{R}$. Plot these functions for $A = 1$, $\sigma = 1.5$, $\mu = 10$, and $\theta = 0.5$. Discuss whether the synaptic kernel models excitation, inhibition, or both. Perturb the three parameters to see the effect they have on the functions. In preparation for the upcoming question, show that for arbitrary values A, σ, μ, θ, the kernel is *balanced*, that is,

$$\int_{\mathbb{R}} w(x)\, dx = 0.$$

Question 4. Show that the NFE (1.1) with synaptic kernel and firing rate given by (2.2) admits the trivial steady state $u(x,t) \equiv 0$ for any values of the parameters A, σ, μ, θ.

Plot the Fourier Transform of W

$$\hat{W}(\xi) = e^{-\xi^2/4} - e^{-\sigma^2\xi^2/4}, \xi \in \mathbb{R},$$

for the parameters given in Question 3. Show that $\hat{W}$ admits global maxima at $\xi = \pm\xi_c$, with $\xi_c = \sqrt{8\ln(\sigma)/(\sigma^2-1)}$ and $\hat{W}_c := \hat{W}(\pm\xi_c)$.

Using the result in Question 1 deduce that, if the coupling parameter is sufficiently large,

$$A > \frac{1}{\hat{W}_c f'(0)} =: A_c, \tag{2.3}$$

the trivial steady state $u(x,t) \equiv 0$ is linearly unstable to perturbations $e^{i\xi x}$, for $\pm\xi$ in an interval including ξ_c. At $A = A_c$ the system undergoes a Turing-like bifurcation. In an experiment where A is set slightly higher than A_c, we expect that spatially-periodic patterns with wavelength ξ_c emerge (albeit this may occur transiently).

3. Tutorial questions - numerical experiment.

In this section we will produce numerical evidence of the Turing-like bifurcation discussed in section 2, and perform a time simulation of a neural field equation. Instead of posing the neural field an unbounded cortex, we shall consider a neural field equation posed on a large but finite cortex D. More specifically, we will take $D = \mathbb{R} \setminus 2L\mathbb{Z}$, that is, a ring of width $2L$ with L large, much larger than the characteristic timescale σ of the synaptic kernel. To fix the ideas, one could set $D = [-L, L)$, and identify L and $-L$. For PDEs, one would naturally speak about *periodic boundary conditions*, but this is misleading for neural field equations, for which boundary conditions need not be specified. The system reads

$$\begin{aligned} &\partial_t u(x,t) = -u(x,t) + \int_D w_p(x-y) f(u(y,t))\, dy, \qquad (x,t) \in D \times [0,T] \\ &u(x,0) = \varphi(x), \end{aligned} \tag{3.1}$$

where f is given in (2.2), and w_p is the $2L$-periodic extension of the function w in (2.2), that is, a $2L$-periodic function such that $w_p(x) = w(x)$ for all $x \in [-L, L)$.

We discretise the cortex D using n evenly spaced points given by

$$x_j = -L + (j-1)h, \qquad j = 1, \dots, n, \qquad h = 2L/n.$$

Henceforth we shall assume that the integer n is *even*. The grid size h is such that the closed interval $[-L, L]$ is split into n equal strips and $n+1$ evenly-spaced points, of which the first

n are taken to discretise $[-L, L)$. This slightly awkward indexing is common when dealing with periodic functions: we approximate $u(x,t)$ only for $x \in [-L, L)$ and $u(L,t)$ is redundant, because it is equal to $u(-L,t)$.

A numerical scheme for approximating (3.1) is heuristically obtained as follows: we evaluate the system at the node set $\{x_i\}_{i=1}^n$, to obtain

$$\begin{aligned} &\partial_t u(x_i,t) = -u(x_i,t) + \int_{-L}^{L} w_p(x_i - y) f(u(y,t))\,dy, \quad i = 1,\dots,n, \quad t \in [0,T], \\ &u(x_i,0) = \varphi(x_i), \qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\quad i = 1,\dots,n \end{aligned}$$

and approximate the integral in the variable y using the composite trapezium rule with n strips. For a $2L$-periodic function g the rule is given by

$$\int_{-L}^{L} g(y)\,dy = g(x_1)\frac{h}{2} + \sum_{j=2}^{n} g(x_j)h + g(x_{n+1})\frac{h}{2} = \sum_{j=1}^{n} g(x_j)h,$$

from which we get confirmation that only function evaluations at nodes $\{x_i\}_{i=1}^n$ are required.

One can show that the scheme above approximates (3.1) via the system n ODEs

$$U'(t) = -U(t) + AMF(U), \qquad t \in [0,T], \qquad U(0) = \Phi, \tag{3.2}$$

where $U(t) \in \mathbb{R}^n$ for all $t \in [0,T]$ is a vector with components $U_i(t) \approx u(x_i,t)$, the vector $\Phi \in \mathbb{R}^n$ has components $\Phi_i = \varphi(x_i)$, and the nonlinear function $F\colon \mathbb{R}^n \to \mathbb{R}^n$, has components $(F(U))_i = f(u_i)$. Finally the matrix $M \in \mathbb{R}^{n\times n}$ is expressed by setting $W_i = W(x_i)$ and

$$\begin{array}{c} \\ 1 \\ 2 \\ \vdots \\ n/2 \\ n/2{+}1 \\ n/2{+}2 \\ \vdots \\ n \end{array}
\begin{array}{c}
\begin{array}{cccccccc} 1 & & 2 & & \cdots & n/2{+}1 & \cdots & n \end{array} \\
\left[\begin{array}{cccccc}
hW_{n/2+1} & hW_{n/2+2} & \cdots & hW_1 & \cdots & hW_{n/2} \\
hW_{n/2} & hW_{n/2+1} & \cdots & hW_n & \cdots & hW_{n/2-1} \\
\vdots & \vdots & \cdots & \vdots & \cdots & \vdots \\
hW_2 & hW_3 & \cdots & hW_{n/2+2} & \cdots & hW_1 \\
hW_1 & hW_2 & \cdots & hW_{n/2+1} & \cdots & hW_n \\
hW_n & hW_1 & \cdots & hW_{n/2+3} & \cdots & hW_{n-1} \\
\vdots & \vdots & \cdots & \vdots & \cdots & \vdots \\
hW_{n/2+2} & hW_{n/2+3} & & hW_2 & \cdots & hW_{n/2+1}
\end{array}\right]
\end{array} = M.$$

One way to navigate the *circulant* matrix M and to relate (3.2) to (3.1) is to recall that the grid has nodes $x_1 = -L$, $x_{n/2+1} = 0$, and $x_{n+1} = L$ (the latter is not used in calculations). The equation

$$\partial_t u(0,t) = -u(0,t) + \int_{-L}^{L} w_p(0 - y) f(u(y,t))\,dy,$$

for instance, is approximated by the ODE

$$U'_{n/2+1}(t) = -U_{n/2+2} + h\sum_{j=1}^{n} w(-x_j) f(U_j) h = -U_{n/2+1} + Ah\sum_{j=1}^{n} W_j f(U_j) h,$$

and the corresponding row of the matrix M is highlighted in red. Similarly

$$\partial_t u(h,t) = -u(h,t) + \int_{-L}^{L} w_p(h-y) f(u(y,t))\, dy,$$

is approximated by

$$U'_{n/2+2}(t) = -U_{n/2+2} + h\sum_{j=1}^{n} w(h-x_j) f(U_j) h = -U_{n/2+2} + h\sum_{j=1}^{n} w(x_{j-1}) f(U_j) h,$$

which, upon recalling that $x_0 = x_n$ by periodicity, gives row $n/2+2$ of the matrix. The fact that M is circulant descends from the integral in (3.1) being a circular convolution, a convolution between periodic functions with the same period.

Question 5. We will now work towards the construction of code to timestep the neural field equation. Assume the following setup: $A = 1$, $\sigma = 1.5$, $\mu = 10$, $\theta = 0.5$, $L = 10\pi$, $n = 2^{10}$.

Write a code that, for generic number of nodes n, stores the nodes $\{x_j\}_{j=1}^n$ in a vector, and forms the n-by-n matrix M. You may want to test the matrix M for low n, and known values of w. Ultimately you can download a file containing the matrix for the parameter above at this link, in the folder `Ring/`.

Question 6. Use the matrix M in Question 5 to time step the set of ODEs (3.2). You can use any off-the-shelf timestepper available on your platform, or write your own time stepper. In the solutions I have used Matlab's in-built `ode45` which is an explicit 4th order, time-adaptive scheme.

Question 7. Produce numerical evidence that, with the parameters given in Question 5, the trivial steady state $u(x,t) \equiv 0$ is linearly stable. To do this, you can set the initial condition Φ to be a random vector with small norm (which models small random perturbations around 0), and observe the initial perturbations decay.

Question 8. From section 2 we know that, upon increasing A above a critical value A_c, we should reach a Turing-like bifurcation. Calculate A_c from (2.3), set $A > A_c$ and repeat the numerical simulation: you should now observe the formation of patterns. If you plot initial and final condition, they should have similar wavelength (they may even have the same roots).

Question 9. From section 2 we know that emerging patterns should have a specific wavelength ξ_c. Verify that, when $A > A_c$, patterns are formed at the predicted wavelength ξ_c. One way to do that is to set a deterministic initial condition with wavelength ξ_c, for instance $\varphi(x) = \varepsilon \cos(\xi_c\, x)$ for $\varepsilon \ll 1$, instead of a random one, and observe the perturbations decay or amplify when A is above or below the critical value, respectively.

Question 10. As for ODEs, bifurcations of stationary states in a neural field equation can be super- or sub-critical. The analysis of the previous section does not address the criticality of the Turing-like bifurcation, but we can do that numerically. One possibility is to time-step the system for various values $\{A_k\}$ in the interval $[1,3] \ni A_c$, keeping an identical initial condition in the form of a small perturbation of the trivial state, say $\varphi(x) = \varepsilon \cos(\xi_c\, x)$.

One observation is that T must be large enough that each simulation gets close to an equilibrium, either the trivial state (when $A < A_c$) or the patterned state (when $A > A_c$).

At the end of the simulation for $A = A_k$, we record a solution measure of the corresponding final state $u_k(x, T)$, for instance the infinity norm $\|u_k(\cdot, T)\|_\infty = \max_{x\in[-L,L]} |u_k(x, T)|$.

We plot the set of points with coordinates $(A_k, \|u_k(\cdot, T)\|_\infty)$ to obtain a crude bifurcation diagram, displaying branches of stable steady states as A varies in $[1, 3]$. Is the Turing-like bifurcation sub- or super-critical?

Question 11. The most expensive operation when timestepping a neural field is the matrix-vector multiplication featuring in the right-hand side of (3.2). The n-by-n matrix M is full, and multiplying it to the right by an n-vector takes $O(n^2)$ operations. This becomes expensive on large-scale computations.

When the matrix M is circulant, one can use Fast Fourier Transforms (FFTs) to perform matrix-vector multiplications in $O(n \log n)$ operations. Amend your code to carry out the matrix-vector multiplication using FFTs, and compare its performance with the old code.