# NONLINEAR DYNAMICAL SYSTEMS PART III DYNAMICS IN PATTERN-FORMING SYSTEMS TUTORIAL 1

DANIELE AVITABILE*

**Abstract.** Themes of of this tutorial:
1. Approximation of differential operators via differentiation matrices.
2. Numerical time stepping for simple reaction–diffusion PDEs.
3. Computing and visualising spatiotemporal solutions.
4. Exploring dependence on initial conditions and control parameters.
5. Choosing solution measures for steady states of PDEs
6. Bifurcation diagrams for homogeneous steady states of PDEs

**1. Introduction.** In this tutorial we will work with the following Allen–Cahn PDE, subject to homogeneous Neumann (no-flux) boundary conditions

$$\begin{aligned} &\partial_t u = \nu\partial_{xx}u + \lambda u + \alpha u^2 + \beta u^3 - \gamma u^5, && (x,t) \in (-L,L)\times\mathbb{R}_{>0}, \\ &\partial_x u(-L,t) = \partial_x u(L,t) = 0, && t \in \mathbb{R}_{\geq 0}, \\ &u(x,0) = \varphi(x), && x \in [-L,L]. \end{aligned} \tag{1.1}$$

We shall derive and time step a set of $n$ coupled ODEs approximating (1.1),

$$\dot{U} = F(U,p), \qquad U(0) = \{\varphi(x_i)\}_{i=1}^n \tag{1.2}$$

where $U(t) \in \mathbb{R}^n$, contains an approximation to $u(x,t)$ at $n$ grid points $\{x_i\}_{i=1}^n$ in $[-L,L]$, $p = (\nu,\lambda,\alpha,\beta,\gamma) \in \mathbb{R}^5$ is a vector of control parameters, and $F\colon \mathbb{R}^n \times \mathbb{R}^5 \to \mathbb{R}^n$.

**1.1. Differential operator and differentiation matrices.** To make progress with the numerical simulation of (1.1), we should discuss how to approximate the second derivative on the right-hand side of the equation.

Given that the PDE contains a second-order spatial derivative, consider the second-order differential operator $\partial_{xx}$, which associates to a twice differentiable, real-valued function $v(x)$ defined on $[-L,L]$, its second derivative $v''(x)$

$$\begin{aligned} \partial_{xx}\colon C^2([-L,L]) &\to C^0([-L,L]), \\ v &\mapsto v''. \end{aligned}$$

Here are examples of how the operator $\partial_{xx}$ maps functions to their second derivatives

$$\begin{aligned} v(x) = x^3 &\mapsto \partial_{xx}v(x) = 6x, \quad x \in [-L,L] \\ v(x) = e^x &\mapsto \partial_{xx}v(x) = e^x, \quad x \in [-L,L] \end{aligned}$$

The operator $\partial_{xx}$ defined above is a useful gadget to interpret the PDE in (1.1): it is a linear operator from the function space $Y = C^2([-L,L])$ to the function space $X = C^0([-L,L])$. You will be able to make progress in this course possibly without

*Vrije Universiteit Amsterdam, Department of Mathematics, Faculteit der Exacte Wetenschappen, De Boelelaan 1081a, 1081 HV Amsterdam, The Netherlands.
Inria Sophia Antipolis Méditerranée Research Centre, MathNeuro Team, 2004 route des Lucioles-Boîte Postale 93 06902, Sophia Antipolis, Cedex, France.
(d.avitabile@vu.nl, writemywebpage).

any background in Functional Analysis, which covers in detail the operator $\partial_{xx}$, its domain $X$ and codomain $Y$. It suffices to say that one way to make progress in the analysis of the PDE (1.1) is to "include" the boundary conditions in the operators, for instance by updating the definition of the domain $Y$ as follows

$$Y = \{v \in C^2([-L, L])\colon v''(-L) = v''(L) = 0\},$$

and setting

$$\partial_{xx}\colon Y \to X, \quad v \mapsto v'', \tag{1.3}$$

which now defines a linear operator "including boundary conditions": the function $v$ on which $\partial_{xx}$ acts satisfies Neumann boundary conditions because $v \in Y$. In fact, it turns out that there are better choices than $Y = C^2$, and one must introduce Sobolev spaces and weak derivatives to have a satisfactory treatment of the PDE as a dynamical system. While we do not intend to study the functional analytic properties of the operator $\partial_{xx}$, we will be concerned with its approximation via *differentiation matrices*, as we now discuss. For this purpose, it is fine to think of $\partial_{xx}$ as an operator defined on $C^2([-L, L])$.

To fix the ideas, let us consider a first-order differential operator, acting on $2L$-periodic functions on $[-L, L]$ to $\mathbb{R}$:

$$\begin{aligned} \partial_x\colon C^1_{\text{per}}([-L, L]) &\to C^0_{\text{per}}([-L, L]) \\ u &\mapsto u'. \end{aligned} \tag{1.4}$$

Such operator does not arise in (1.1), and would be suitable for periodic (as opposed to Neumann) boundary conditions. We discuss it here as it is instructive to do so, and it helps us jump-start with differentiation matrices.

Let us consider a set of grid points $\{x_j\}$ and the corresponding function values $\{u(x_j)\}$. Differentiation matrices use this data to provide an approximation to derivatives of $u$ at the nodes $\{x_j\}$. For instance, a differentiation matrix for the first derivative approximates $\{u'(x_j)\}$, the first derivative $u'$ of the function $u$ at the grid points.

On a periodic domain, we take gridpoints $x_j = -L + h(j-1)$, $h = 2L/n$, for $j = 1, \dots, n+1$. A common finite difference approximation for $u'(x_j)$ is given by

$$u'(x_j) = \frac{u(x_{j+1}) - u(x_{j-1})}{2h} + O(h^2), \quad \text{as } h \to 0, \qquad u'(x_j) \approx \frac{u(x_{j+1}) - u(x_{j-1})}{2h}.$$

If we collect the $\{u(x_j)\}$ in a vertical vector $U = \{U_j\}_{j=1}^n$, and take into account the periodicity of $u(x)$, we obtain

$$\underbrace{\frac{1}{2h}\begin{pmatrix} 0 & 1 & & & -1 \\ -1 & 0 & 1 & & \\ & \ddots & \ddots & \ddots & \\ & & -1 & 0 & 1 \\ 1 & & & -1 & 0 \end{pmatrix}}_{D_x} \begin{pmatrix} U_1 \\ U_2 \\ \vdots \\ U_{n-1} \\ U_n \end{pmatrix} \approx \begin{pmatrix} u_1' \\ u_2' \\ \vdots \\ u_{n-1}' \\ u_n' \end{pmatrix} \tag{1.5}$$

The matrix $D_x$ on the left-hand side is the differentiation matrix generated by the familiar second order centred finite difference formula to approximate $u'$. Note that

the condition $U_{n+1} = U_1$ (periodicity of the function) is not explicitly present in the discretisation, but the fact that $u(x) = u(x + 2L)$ for all $x$ is implicitly accounted for, in the first and last row of the differentiation matrix. In addition, the vector $D_x U$ returns an approximation to a $2L$-periodic function, by design. This is similar in spirit to what happened for the operator $\partial_x$, which embedded boundary conditions in its domain $Y = C^1_{\text{per}}([-L, L])$. Also, note that the matrix we obtained is sparse. If we fix $L$ and increase $n$, we obtain a larger matrix (more grid points are involved) and standard error bounds on finite difference formulas ensure that we get a more accurate approximation to $u'$.

To understand in which sense the differentiation matrix $D_x$ in (1.5) approximates the differential operator $\partial_x$ in (1.4) compare the mappings

$$u \mapsto \partial_x u, \qquad U \mapsto D_x U.$$

The former is between the infinite-dimensional function spaces $C^1_{\text{per}}([-L, L])$ and $C^0_{\text{per}}([-L, L])$. The latter is between finite-dimensional spaces, from $\mathbb{R}^n$ to $\mathbb{R}^n$. If $U$ stores an $n$-dimensional approximation to $u$, we expect that the matrix-vector multiplication $D_x U$ produces an $n$-dimensional approximation to $\partial_x u$. The larger $n$, the closer $D_x U$ is to $\partial_x u$, in a sense that we will now discuss.

**2. Tutorial questions.**

**Question 1** (Differentiation matrices with periodic BCs). For this question you do not have to write code, but rather understand and run code that is provided to you. Download and familiarise yourself with the Matlab function `PeriodicDiffMat` with interface

```
function [x,Dx,Dxx] = PeriodicDiffMat(xSpan,nx)
```

which returns gridpoints, a differentiation matrix for $\partial_x$, and a differentiation matrix for $\partial_{xx}$ with periodic boundary conditions on the domain $[a, b)$. The function takes as input the boundary of the domain, via the vector `xSpan` with components $a$ and $b$, and the number of gridpoints `nx`.

Run the code under Question 1 in `driver.m` and see how the function is used to approximate the first and second derivative of the function $u(x) = e^{\cos x}$. The code for Question 1 plots the analytical first and second derivatives, as well as their numerical approximations.

**Question 2** (Convergence test for differentiation matrices). question 1 give some "visual" evidence that the differentiation matrices work as expected. We now work towards a more reliable and rigorous test. From standard numerical analysis results, it is known that the finite-difference approximations used in the differentiation matrices produce an $O(h^2)$ approximation or, equivalently, an $O(n^{-2})$ approximation. To test that our matrices are correct, we now produce numerical evidence of this fact.

Understand and run the code in Question 2 of `driver.m` question. The code produces numerical evidence that the differentiation matrices $D_x$ and $D_{xx}$ satisfy the following error bounds

$$\max_{i=1}^{n} |(D_x U)_i - \partial_x u(x_i)| = O(n^{-2}),$$
$$\max_{i=1}^{n} |(D_{xx} U)_i - \partial_{xx} u(x_i)| = O(n^{-2}),$$

where $U \in \mathbb{R}^n$ is a vector with components $U_i = u(x_i)$, $u(x_i)$ is a function in $C^4_{\text{per}}([-L, L])$, $D_x, D_{xx} \in \mathbb{R}^{n \times n}$ are differentiation matrices. In the previous equations, $(D_x U)_i$ denotes the $i$th component of the vector obtained by multiplying $D_x$

by $U$, and therefore containing an approximation to $\partial_x u(x_i)$. Similar considerations apply to $(D_{xx}U)_i$. Before proceeding, it is important that the solutions to Questions 1 and 2 are clear to you.

**Question 3** (Matrices with Neuman BCs). We now start building code to simulate the PDE (1.1). A first step is coding a differentiation matrix $D_{xx}$ which encapsulates Neumann boundary conditions. Write a function named `NeumannDiffMat` that produces grid points and differentiation matrix for $\partial_{xx}$ on the domain $[a, b]$ with Neumann Boundary conditions $\partial_x u(a) = \partial_x u(b) = 0$. You can take inspiration from `PeriodicDiffMat`. Here are some important differences

1. The $n$ gridpoints in the interval $[a, b]$ for Neumann boundary conditions are given by
$$x_i = a + h(i-1), \qquad h = \frac{b-a}{n-1}, \qquad i = 1, \dots, n,$$
that is, the grid includes both $a$ and $b$, unlike the periodic case.
2. The differentiation matrix with Neumann Boundary conditions now reads
$$D_{xx} = \frac{1}{h^2}\begin{pmatrix} -2 & 2 & & & \\ 1 & -2 & 1 & & \\ & \ddots & \ddots & \ddots & \\ & & 1 & -2 & 1 \\ & & & 2 & -2 \end{pmatrix}.$$
To understand why the first and last rows of $D_{xx}$ encode Neumann boundary conditions, write down the centred finite difference formula for $\partial_{xx}u$ at $x = a$,
$$\partial_{xx}u(a) = \frac{u(a-h) - 2u(a) + u(a+h)}{h^2} + O(h^2),$$
realise that it involves the node $x = a - h$ which lies outside of the domain, and note that this node also occurs when one imposes the Neumann boundary condition
$$0 = \partial_x u(a) = \frac{-u(a-h) + u(a+h)}{2h} + O(h^2).$$
Combining the previous two expressions one obtains
$$\partial_{xx}u(a) = \frac{-2u(a) + 2u(a+h)}{h^2} + O(h^2),$$
which is used in the first row of the matrix $D_{xx}$. A similar reasoning holds for the last row.

Add code in `driver.m` so as to test the function `NeumannDiffMat`, proceeding as in Question 1: approximate the second derivative of a function that satisfies Neumann boundary conditions, for instance
$$u(x) = \cos(\pi x/5), \qquad x \in [-5, 5]$$
Plot on the same graph $\partial_{xx}u$ and its approximation $D_{xx}U$.

**Question 4** (Convergence of differentiation matrices). Test the function you wrote in Question 3 by proceeding as in Question 2: produce numerical evidence that the differentiation matrix $D_{xx}$ satisfies the bound
$$\max_{i=1}^{n} |(D_{xx}U)_i - \partial_{xx}u(x_i)| = O(n^{-2}),$$
for the function $u(x)$ you chose in Question 3.

**Question 5** (Approximate the RHS of (1.1)). Using pen and paper, show that an approximating set of $n$ ODEs for the PDE (1.1) is given by

$$\begin{aligned}\dot{U}_1 &= \nu\frac{2U_2 - 2U_1}{h^2} + N(U_1),\\ \dot{U}_i &= \nu\frac{U_{i+1} - 2U_i + U_{i-1}}{h^2} + N(U_i), \qquad i = 2,\dots,n-1\\ \dot{U}_n &= \nu\frac{2U_{n-1} - 2U_n}{h^2} + N(U_n),\end{aligned}$$

with initial condition

$$U_i(0) = \varphi(x_i), \qquad i = 1,\dots,n$$

where $N$ is the 5th order polynomial

$$N(u) = \lambda u + \alpha u^2 + \beta u^3 - \gamma u^5,$$

and convince yourself that, in vectorial form, the system of ODEs can be written as

$$\dot{U} = F(U,p), \qquad U(0) = \{\varphi(x_i)\}_{i=1}^n \tag{2.1}$$

where $U \in \mathbb{R}^n$, $p = (\nu,\lambda,\alpha,\beta,\gamma) \in \mathbb{R}^5$ is a vector containing all control parameters, and $F\colon \mathbb{R}^n \times \mathbb{R}^5 \to \mathbb{R}^n$ can be expressed in terms of $D_{xx}$ and $N$.

**Question 6** (Code the RHS of (1.1)). Create a Matlab function `AllenCahn` using the following prototype

```
function F = AllenCahn(u,p,Dxx)
```

The function accepts as inputs the differentiation matrix $D_{xx} \in \mathbb{R}^{n\times n}$, a column vector $U \in \mathbb{R}^n$, a vector $p \in \mathbb{R}^5$, and returns in `F` a column vector containing $F(U,p) \in \mathbb{R}^n$.

**Question 7** (Test the RHS of (1.1)). Test the function `AllenCahn`. To do this, notice that if $u(x) = \cos(\pi x/5)$, for $x \in [-5,5]$, then $u$ satisfies Neumann boundary conditions at $x = \pm 5$ and the right-hand side of the Allen–Cahn equation can be computed in closed form,

$$\nu\partial_{xx}u + \lambda u + \alpha u^2 + \beta u^3 - \gamma u^5 = v$$

where

$$v(x) = -(\pi/5)^2\nu\cos(\pi x/5) + \lambda\cos(\pi x/5) + \alpha\cos^2(\pi x/5) + \beta\cos^3(\pi x/5) - \gamma\cos^5(\pi x/5).$$

If `AllenCahn` is implemented correctly, then it should return an approximation to the vector $V$ with components $V_i = v(x_i)$, provided its input vector $U$ has components $U_i = \cos(\pi x_i/5)$. Verify that, if $n = 100$, and $p = (1,-0.2,0,1,1)$ then

$$\|F(U,p) - V\|_\infty = \max_{i=1}^n |F_i(U,p) - V_i| \le 2\times 10^{-4}.$$

**Question 8** (Time step the PDE (1.1)). We are now ready to integrate the Allen–Cahn PDE (1.1). Time step the set of ODEs (2.1) with $n = 100$ for $(x,t) \in [-5,5]\times[0,50]$ with parameters $\nu = 1$, $\lambda = -0.2$, $\alpha = 0$, $\beta = 1$, $\gamma = 1$, and initial condition[1] $\varphi(x) = (\cosh(x))^{-2}$. You can use one of Matlab's in-built time steppers,

[1]Strictly speaking, the initial condition should satisfy Neumann boundary conditions, and this one does not, because $\varphi'$ at the boundary is small, but not 0, $\varphi'(\pm 5) \approx 7\times 10^{-4}$. One can check numerically that this initial "imperfection" is quickly cared for by the differentiation matrix, that enforces the boundary conditions $\partial_x u(\pm 5,t) = O(h^2)$ for all $t > 0$.

for instance `ode15s`. Plot the approximation $u(x,t)$ for $(x,t) \in [-5,5] \times [0,50]$. Observe the solution dynamics: you should see that the initial "peak" in the solution dissipates, and the solution approaches an equilibrium. Is the equilibrium spatially homogeneous or heterogeneous?

**Question 9** (Equilibria depend on the parameter $\lambda$). Repeat Question 8 with $\lambda = 0.1$. This experiment shows that the system attains a homogeneous equilibrium different from the one in Question 8, hence equilibria depend on the parameter $\lambda$. Henceforth we will use $\lambda$ as our principal control parameter.

**Question 10** (Patterned equilibria). Set $\lambda = 0.3$ and $t \in [0, 200]$. Time step the problem with initial conditions $\varphi(x) = 0.2\cos(\pi x/5)$ and $\varphi(x) = 0.4\cos(\pi x/5)$. Produce numerical evidence that when $\lambda = 0.3$ and all other parameters are as in Question 8, the Allen–Cahn PDE (1.1) supports 2 co-existing stable equilibria, one homogeneous, and one heterogeneous (patterned state). Such coexisting steady states are selected by slight variations in initial conditions.

**Question 11** (Solution measures). We have seen that the Allen–Cahn PDE supports both homogeneous and patterned steady states, and we aim to plot all of them on the same bifurcation diagram in the parameter $\lambda$. At this stage, we must choose a so-called *solution measure*, that is, a scalar variable to put on the ordinates of the bifurcation diagram.

Let $u_*$ denote a steady state of the PDE defined on $[-L, L]$. When $u_*(x) \equiv C$, for some $C \in \mathbb{R}$ (homogeneous steady state) then a possible solution measure is

$$S_1(u_*) = C,$$

which is, however, not useful when $u_*$ depends on $x$ (patterned state). A popular solution measure applicable to both homogeneous and heterogeneous steady states is the scaled $L_2$-norm of $u_*$

$$S_2(u_*) = \|u_*\|_{L^2} = \left( \frac{1}{2L} \int_{-L}^{L} |u_*(x)|^2 \, dx \right)^{1/2},$$

which returns $|C|$ for homogeneous steady states $u_*(x) \equiv C$, and a positive real number for patterned states. Bifurcation diagrams are then drawn on the $(\lambda, S_1)$-plane, or the $(\lambda, S_2)$-plane (and sometimes the two diagrams are superimposed).

Create a function with the following prototype

```
function S = ComputeL2Norm(u,x)
```

which takes a vector `x` containing the gridpoints $x_i$, a vector `U` with components $U_i \approx u(x_i)$, and returns an approximation to the scaled $L^2$-norm of $u$. Test your function in `driver.m`. For instance, compute $S$ for $u_*(x) \equiv \pm 4$, and for a heterogeneous $u_*(x)$.

**Question 12** (Branches of homogeneous steady states). We can now study bifurcation diagrams of homogeneous steady states, in the parameter $\lambda$, with solution measure $S_1$.

The PDE (1.1) depends on parameters $p = (\nu, \lambda, \alpha, \beta, \gamma)$ and we aim to vary $\lambda$ and fix all the others. Henceforth you should fix $\mu = 1$, $\alpha = 0$, $\beta = 1$, $\gamma = 1$, unless otherwise stated.

Use pen and paper to show that homogeneous steady states $u_*(x) \equiv C$ of the Allen–Cahn PDE (1.1) are determined by an implicit equation

$$G(\lambda, C) = 0,$$

where $G\colon \mathbb{R}^2 \to \mathbb{R}$ is a quintic polynomial in the variable C. We can now use the fucntion $G$ in two ways:

1. Fix $\lambda = \lambda_*$, and find $C$ such that $G(\lambda_*, C) = 0$ to determine homogeneous equilibria of the Allen–Cahn PDE for $\lambda = \lambda_*$. Show that, depending on the value $\lambda_*$, the Allen–Cahn PDE admits 1, 3, or 5 homogeneous equilibria. Do the equilibria have any special property? Any symmetry?
2. Explain to a colleague (or convince yourself of) the following statement: *the locus of points* $(\lambda, C)$ *satisfying* $G(\lambda, C) = 0$ *is a bifurcation diagram of homogeneous equilibria in the parameter* $\lambda$*, with solution measure* $S_1$ *(excluding solutions stability).* Use Matlab's command `fimplicit` to plot a bifurcation diagram of homogeneous steady states in $(\lambda, S_1)$-plane. How many solution branches do you see? Do you see any bifurcation? If yes, can you guess their type? Discuss this with colleagues. Also, seek confirmation of what you found in Question 12.1.