ANALYSING PATTERN FORMATION IN THE GRAY-SCOTT MODEL: AN XPPAUT TUTORIAL

DEMI L. GANDY[*] AND MARTIN R. NELSON[†]

Abstract. The Gray-Scott model is a widely-studied autocatalytic model that exhibits a range of interesting pattern formation behaviour, as well as a rich structure of dynamics that includes many ideas from a typical undergraduate dynamical systems course, and some from beyond. Understanding the solutions to this model is arguably most easily conducted via a bifurcation analysis of corresponding ODE problems within the software XPPAUT. In this paper, we provide an introductory XPPAUT tutorial, through which we begin to expose the range of intricate patterns that the Gray–Scott model emits.

Key words. Reaction-Diffusion, Gray-Scott model, PDE, XPPAUT

1. Introduction.

The spontaneous formation of spatial (and spatio-temporal) patterns has long been of interest to Mathematicians, with previous literature describing several models that show how a pair of reacting and diffusing chemicals can give rise to patterns such as spots, stripes, spirals and more. (See *e.g.* [1, 6, 9, 13].) These models have applications including both experimental studies and pattern formation in biological settings such as animal hides, amongst others. (See Figure 1 for some examples of these.) Since the concentrations of the chemicals involved evolve as functions of both the spatial position and time, models largely comprise of two or more coupled partial differential equations (PDEs). Their analysis involves the application of the key ideas that feature in most undergraduate PDE or dynamical systems courses, with the latter being particularly useful in understanding how the rich array of patterns that models permit depends on related model parameters. In many cases, it is helpful to perform a bifurcation analysis to fully understand the models' dependence on parameter values. For complicated models, this often requires a numerical approach that uses appropriate bifurcation software, with XPP-Auto (XPPAUT; available from [3]) being a common choice. In this article, we describe the analysis of one such reaction–diffusion PDE, the Gray–Scott model, and describe how to perform the necessary analysis in XPPAUT – software which most undergraduate students won't study in class. We include code examples that could be readily adapted to analyse other related models.

In 1984, Gray and Scott [4] introduced a reaction-diffusion (PDE) model of the following irreversible, autocatalytic reaction:

$$U + 2V \to 3V, \tag{1.1a}$$

$$V \to P, \tag{1.1b}$$

in which two generic chemical species U and V react to produce a product P. Their model assumes that U is continuously supplied, and the inert product P is continuously removed. After a suitable re-scaling, the Gray–Scott model can be expressed

[*] Department of Physics & Mathematics, Nottingham Trent University, Nottingham, NG11 8NS, United Kingdom (demi.gandy2016@my.ntu.ac.uk)

[†] Department of Physics & Mathematics, Nottingham Trent University, Nottingham, NG11 8NS, United Kingdom (martin.nelson@ntu.ac.uk).

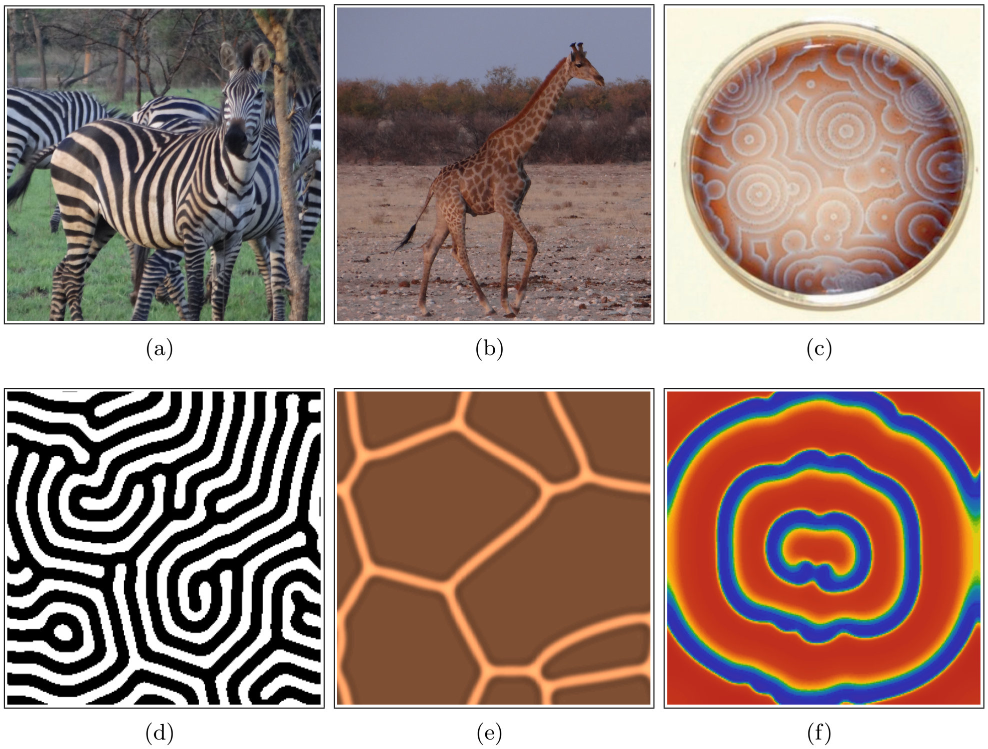


Fig. 1. *(a,b) Striped and hexagonal patterns on the hides of a zebra and a giraffe. Photographs courtesy of Nina Čapek. (c) Spiral patterns produced by the Belousov–Zhabotinsky reaction. Image reproduced from [1] under the terms of the Creative Commons Attribution License http://creativecommons.org/licenses/by/4.0/. Original image courtesy of Michael C. Rogers and Stephen Morris, University of Toronto. (d–f) Corresponding patterns obtained from simulations of the Gray–Scott model.*

by the following pair of equations:

$$\frac{\partial u}{\partial t} = D_u \nabla^2 u - uv^2 + F(1-u), \tag{1.2a}$$

$$\frac{\partial v}{\partial t} = D_v \nabla^2 v + uv^2 - (F+k)v, \tag{1.2b}$$

in which u and v are concentrations of U and V respectively. The feed or flow parameter F is the rate of replenishment of U, and the kill parameter k is the rate at which V decays to produce P. The parameters D_u and D_v are diffusion constants, representing the rate at which each chemical moves spatially, and ∇^2 represents the Laplacian operator, *i.e*

$$\nabla^2 = \frac{\partial^2}{\partial x^2}, \qquad \text{or} \qquad \nabla^2 = \frac{\partial^2}{\partial x^2} + \frac{\partial^2}{\partial y^2}, \tag{1.3}$$

in one- or two-dimensional settings respectively. Since the parameters F, k, D_u and D_v all represent some form of biological rate constant (either a rate of reaction or a rate of diffusion), these parameters always take positive values.

The Gray-Scott model has been shown to exhibit a range of interesting pattern formation behaviour in both one [10, 11, 14, 15] and two spatial dimensions [12, 16], some of which are illustrated in Figure 1(d–f). The starting point for most analyses is to understand the dynamics of the corresponding homogeneous system (obtained by neglecting the diffusion terms in the model). This involves calculation of the model's steady states and their stability, and understanding how these change as parameters are varied, following the standard procedure outlined in a typical undergraduate dynamical systems course. Various authors have performed this type of analysis for the Gray–Scott model [2, 10, 14, 15, 16], to identify a rich structure of bifurcations, some of which an undergraduate would be familiar with, and some of which they typically would not. We explore this in more detail below. With an understanding of the homogeneous system in hand, we then re-introduce the spatial terms and examine how these give rise to spatially inhomogeneous solutions, focusing upon stationary one-dimensional patterns in particular. We then briefly comment on how these ideas may be extended to two-dimensional domains. Our objective throughout this work is to provide an undergraduate reader, who has perhaps only completed an introductory dynamics course so far, with some insight into more complicated dynamical systems problems that give rise to interesting spatial solutions (patterns) and their analysis in XPPAUT. We occasionally also make use of Matlab to visualise our patterns; all XPPAUT and Matlab codes used in this paper are freely available online at https://github.com/martinrnelson/GrayScott. Thus, the pre-requisite knowledge required in order to work through the tutorial given here includes: an introductory foundation in dynamical systems analysis and bifurcation theory (including saddle-node and Hopf bifurcations); some familiarity with common, more simple PDEs such as the heat equation (including time-varying solutions on one- and two-dimensional spatial domains); and knowledge of basic numerical methods techniques, such as solution of ODEs and PDEs via finite difference methods. We expand on some of these topics and delve more deeply into the behaviours of the Gray-Scott model below.

2. Analysis of the homogeneous model. We begin our analysis by considering the homogeneous model obtained by eliminating the spatial terms from Equation (1.2), *i.e.*

$$\frac{\mathrm{d}u}{\mathrm{d}t} = -uv^2 + F(1-u) \equiv f(u,v), \tag{2.1a}$$

$$\frac{\mathrm{d}v}{\mathrm{d}t} = uv^2 - (F+k)v \equiv g(u,v). \tag{2.1b}$$

In this setting, u and v do not exhibit any spatial dependence and our PDE system has hence reduced to a pair of ODEs, to which we can apply the ideas of our undergraduate dynamical systems course. Solutions of Equation (2.1) correspond exactly to the subset of solutions of Equation (1.2) that are spatially homogeneous (*i.e.* unpatterned).

We identify the steady states of this system by setting the right-hand sides of Equation (2.1) to zero, in the usual way, to provide the following pair of simultaneous equations for u and v:

$$F(1-u) - uv^2 = 0, \tag{2.2a}$$

$$uv^2 - (F+k)v = 0. \tag{2.2b}$$

From this we obtain a trivial steady state at $(u^*, v^*) = (1, 0)$ and two further steady states, which we denote by (u^+, v^-) and (u^-, v^+) respectively, where

$$(u^+, v^-) = \left(\frac{1}{2}\left(1+\sqrt{1-\frac{4(F+k)^2}{F}}\right), \frac{1}{2}\frac{F}{F+k}\left(1-\sqrt{1-\frac{4(F+k)^2}{F}}\right)\right), \tag{2.3a}$$

$$(u^-, v^+) = \left(\frac{1}{2}\left(1-\sqrt{1-\frac{4(F+k)^2}{F}}\right), \frac{1}{2}\frac{F}{F+k}\left(1+\sqrt{1-\frac{4(F+k)^2}{F}}\right)\right). \tag{2.3b}$$

Since the steady state at $(1, 0)$ does not depend on the parameters, it exists for all choices of F and k. It is straightforward to show that the quadratic equation that gives rise to the steady states (u^+, v^-) and (u^-, v^+) only has real roots provided that

$$0 < k < \frac{1}{2}\left(\sqrt{F} - 2F\right), \qquad 0 < F < \frac{1}{4}. \tag{2.4}$$

Equation (2.4) provides a restriction in (k, F)-space for the existence of our additional two fixed points, as illustrated in Figure 2. For choices of parameter values that fall outside of the solid curve in Figure 2, *i.e.* in region I, only the steady state at $(1, 0)$ exists. Inside this curve, in regions II and III, all three steady states exist. Since crossing the solid curve by moving from region I to regions II or III results in two additional steady states being created, we know that the solid curve represents a curve of saddle-node bifurcations.

We compute the stability of our three steady states by considering, as usual, the eigenvalues of the Jacobian matrix $\mathbf{J}$, given by

$$\mathbf{J} = \begin{pmatrix} -F - v^2 & -2uv \\ v^2 & 2uv - (F+k) \end{pmatrix}, \tag{2.5}$$

evaluated at each steady state in turn. Given that the eigenvalues of $\mathbf{J}$ are given by $\lambda = \frac{1}{2}\left(\mathrm{tr}(\mathbf{J}) \pm \sqrt{\mathrm{tr}(\mathbf{J})^2 - 4|\mathbf{J}|}\right)$, and recalling that our steady state is stable if both eigenvalues have negative real part, we can easily show that the trivial steady state $(1, 0)$ is always stable, the fixed point at (u^+, v^-) is always a saddle point, and the steady state at (u^-, v^+) is stable provided that

$$F > \frac{1}{2}\left(-\sqrt{k - 4k^{\frac{3}{2}}} - 2k + \sqrt{k}\right), \tag{2.6}$$

and unstable otherwise. Plotting this restriction in (k, F)-space provides the dotted black curve in Figure 2; the steady state at (u^-, v^+) is stable in region II and unstable in region III. Since crossing the dotted line in Figure 2 corresponds to a pair of complex conjugate eigenvalues having a real part that changes sign, this line corresponds to a line of Hopf bifurcations, which we also associate with the existence of oscillatory solutions. Our analysis below will largely focus upon the two steady states that have the potential to be stable for some parameters. The steady state at $(1, 0)$ is typically referred to as the red state, in which no chemical V is present in the system; the steady state at (u^-, v^+) is often called the blue state, in which the concentration of V has grown up to a level where it is effectively controlled by depletion of U [16].

Our analysis so far has essentially followed a standard undergraduate dynamical systems analysis, and revealed two sets of bifurcations (saddle-node and Hopf) that most readers will be aware of from that setting. From Figure 2, we also see an additional interesting point at $(F, k) = (0.0625, 0.0625)$ where the saddle-node (solid) line

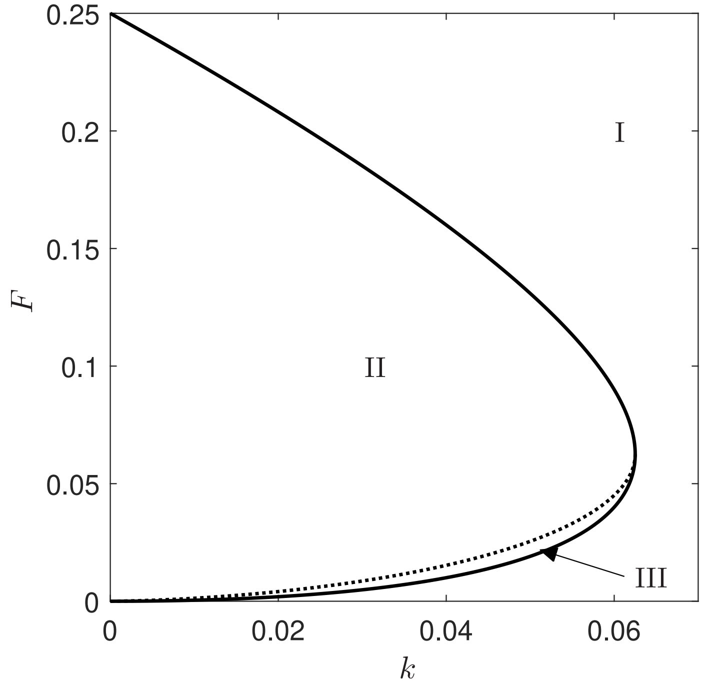


FIG. 2. *Bifurcation diagram for Equation (2.1). Inside the solid curve, Equation (2.4) is satisfied and all three steady states exist, whereas in region I, only the trivial steady state* $(1,0)$ *exists. Above the dotted black line (in region II) Equation (2.6) is satisfied and the steady state* (u^-, v^+) *is stable. As we decrease* F *down through the dotted black line into region III, the steady state* (u^-, v^+) *becomes unstable. The solid and dotted curves represent curves of saddle-node and Hopf bifurcations respectively.*

and the Hopf (dotted) line intersect. This point corresponds to a Bogdanov-Takens bifurcation [8]; a bifurcation of codimension two that many undergraduate readers may not have encountered previously. The equilibrium solution at the Bogdanov-Takens point has a zero eigenvalue of algebraic multiplicity two. At this point, a saddle and a non-saddle collide at the same point that the non-saddle undergoes a Hopf bifurcation that gives rise to a periodic orbit. The interaction between the periodic orbit and the saddle then also eliminates the periodic orbit via a homoclinic bifurcation [18]. It is certainly the case that visualising the phase planes that the model permits for parameters close to the Bogdanov-Takens bifurcation is far from trivial. We thus turn to XPPAUT below in order to compute full bifurcation diagrams for our model (which supplement our existing understanding of the steady states with more detailed analysis of periodic structures, in particular), before then also examining corresponding phase planes for various choices of parameters. The reader may like to note, here, that phase planes corresponding to parameter choices in the three regions of Figure 2 will be given in Figure 7 below. (We also recommend that readers interested in knowing more about Bogdanov–Takens bifurcations refer to [5], which nicely illustrates how phase planes change close to these points.)

The XPPAUT software incorporates two related tools. Firstly, the XPP tool is used to perform numerical integration from given initial conditions to *e.g.* a stable steady state, for a fixed set of parameter values. Secondly, the AUTO tool is used to track how the coordinates of steady states *etc.* change as one or more model parameters are varied, allowing us to construct corresponding bifurcation diagrams. This is achieved via a process of numerical continuation, primarily using a pseudo-arclength continuation algorithm. Briefly, the objective of continuation algorithms in general is to identify approximate solutions of an equation of the form $f(\mathbf{x}, \mu) = 0$ in terms of some parameter μ. Here, $\mathbf{x}$ is a vector of unknowns, and we wish to determine which values of $\mathbf{x}$ satisfy $f(\mathbf{x}, \mu) = 0$ for given choices of the parameter μ. In the context of dynamical systems, μ can be thought of as a model parameter and $\mathbf{x}$ as a vector of steady state coordinates. Figure 3 briefly illustrates how the pseudo-arclength continuation approach tackles this task. We begin with an initial choice of the parameter $\mu = \mu_0$ and numerically integrate our ODEs in XPP until they reach steady state; the steady state coordinates provide a choice of $\mathbf{x}$ that satisfies $f(\mathbf{x}, \mu_0) = 0$, and hence gives us an initial point on our bifurcation curve. We then numerically compute the tangent to the curve at this point, and then attempt to predict a new point on the curve at a distance $\mathrm{d}s$ along this tangent vector. For suitably small choices of $\mathrm{d}s$, we can expect to attain a new point for which the function f takes values close to (but not exactly) zero; an iterative root finding method, such as Newton's method is then applied to more accurately determine the appropriate value of $\mathbf{x}$ at our new parameter value, and hence our new point on the curve sought. This process is repeated iteratively until the entire curve has been identified. Since the algorithm takes steps along tangents to the curve, we take small steps in the parameter μ when gradients are steep, and larger steps when gradients are shallow. Furthermore, we have the freedom to choose negative choices of $\mathrm{d}s$ in order to track along the curve in the opposite direction. Here, the parameter $\mathrm{d}s$ is an approximation of the arclength between two successive points on our curve. Carefully choosing this parameter is key to the success of the algorithm; however, AUTO also includes the capability to automatically adapt its choices of $\mathrm{d}s$ between prescribed maxima and minima, which takes some of the burden of choosing $\mathrm{d}s$ away from the user (although this does still require some user control, as we will see below). For more detail on numerical continuation algorithms in general, we direct the reader to the excellent textbooks of [7] and [17].

Our XPPAUT analysis begins with the construction of an ODE file, specifying the model of Equation (2.1), as shown in Algorithm 2.1. The first line of this file is a simple heading which XPPAUT ignores. This is followed by specification of the parameters F and k (here denoted `Feed` and `kill` on line 2), initial conditions (on line 3) and the ODEs themselves (on lines 4–5). Finally, on line 6, we specify default axis limits that XPPAUT will use, together with a maximum integration time (`total`). These parameters are not crucial, and can be amended at any time. The keyword `done` demarks the end of the file. In this example, we have chosen the parameter values $F = 0.01$ and $k = 0.03$, and initial conditions are chosen as the corresponding fixed point (u^-, v^+) obtained by inputting said parameters into Equation (2.3b).

Algorithm 2.1 ODE file used to define the Equation (2.1) in XPPAUT.

```
# Homogeneous Gray-Scott System
par Feed=0.01 kill=0.03
init u=0.2 v=0.2
u'=-u*v^2+Feed*(1-u)
v'= u*v^2-(Feed+kill)*v
@ xlo=0 xhi=1000 ylo=0 yhi=1 total=1000
done
```

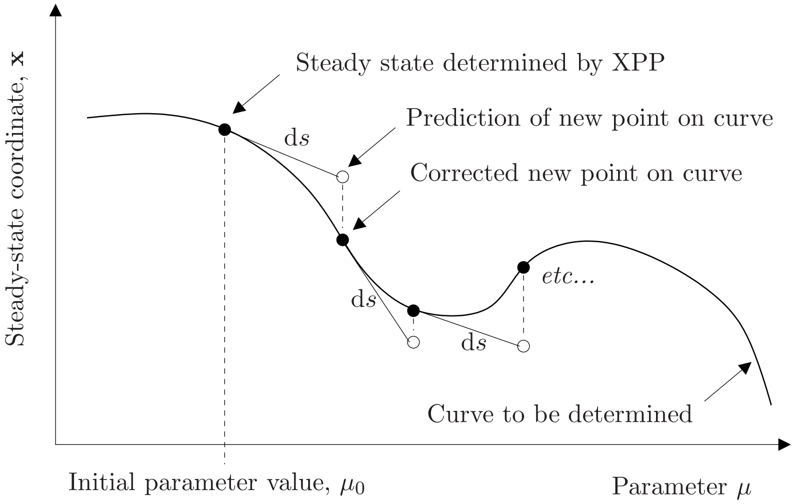

Steady-state coordinate, x
Steady state determined by XPP
Prediction of new point on curve
Corrected new point on curve
ds
ds
ds
etc...
Curve to be determined
Initial parameter value, μ_0
Parameter μ

FIG. 3. *Schematic diagram of a typical pseudo-arclength continuation algorithm, such as that used by AUTO to track steady states as a function of parameter* μ*.*

The easiest way to navigate XPPAUT is via keyboard shortcuts. The shortcut letter for the desired option in the active menu is given by a capital letter (which may or may not be the first letter of the word). Upon launching XPPAUT with the above ODE file, we can click the drop down menu titled `Initialconds` and then navigate to `(G)o`, or instead we can simply type `IG` for the same result. Since we set our initial conditions as the steady state (u^-, v^+) in the ODE file, the resulting graph is a horizontal line, as shown in Figure 4. (As a proof of concept, we can alternatively click `Initialconds` and this time navigate to `(N)ew`, which allows us to pick initial conditions different to those in the ODE file; Figure 4 additionally shows the integration resulting from the initial condition at $(0.8, 0.2)$, as an example.) Before moving to the AUTO window, it is important to make sure that the integration has reached equilibrium to a reasonable degree of accuracy (else the continuation methods used to trace a branch in AUTO can often fail). We can achieve this by either increasing `total` in the ODE file if needed, or by selecting `Initialconds` and `(L)ast` (`IL`) two or three times, to continue the integration from the values obtained on the last computation.

Once we have achieved full convergence to a steady state (or alternatively, convergence to a periodic orbit, although this is less reliable) we can use AUTO to perform bifurcation analysis. To launch AUTO, select `File` then `Auto`, or alternatively use the keyboard shortcut `FA`. Before we can generate a bifurcation diagram, we need to change the default numerical settings. We simply click `Numerics` to bring up the relevant window. For different systems, it can take some trial and error to find a good balance, but in the case of creating a bifurcation diagram for the system given in Equation (2.2), changing the default values of the following options works well:

- `Nmax:250`. This describes the maximum number of steps taken along any branch. This has been increased slightly from the default value.
- `Npr:250`. This instructs AUTO to provide information about a point every `Npr` steps. If we set `Npr=Nmax`, only points of interest will be highlighted, making the resulting bifurcation diagram clearer.
- `Ds:0.001`. This is the initial step size. Since the step size is adaptive, this

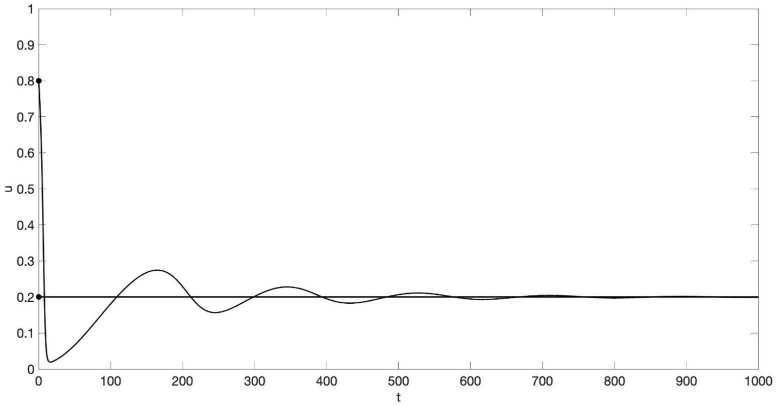


FIG. 4. *Integrations of Equation (2.1) in XPPAUT with initial conditions $(u,v) = (0.2, 0.2)$ and $(u,v) = (0.8, 0.2)$, showing $u(t)$ only.*

is only a suggested starting value. (We note that, in some problems, setting this parameter to be negative can sometimes be helpful in tracing branches that fold back on themselves.)

- `Dsmax:0.01`. This is the maximum step size. If this is too big, AUTO will sometimes miss important points, and the bifurcation diagram can end up looking jagged. Therefore, this has been decreased from the default value.

The remaining numerical parameters that can be controlled in this window are not important here; for more guidance about the remaining parameters, see [3]. Once we have changed the numerical settings, we return to the main AUTO window and simply click `Run` and then `Steady state` (`RS`) to draw the first branch(es) of the bifurcation diagram. This bifurcation diagram will likely be produced at an inefficient scale, so we can select `Axis` then `Fit` (`AF`) to automatically adjust the scale of the axes (or `Axis` then `hI-lo` (`AI`) to do this manually). Upon creating a bifurcation diagram, AUTO will mark points of interest by a numbered cross, and add a label to indicate the nature of this point; most commonly: `EP` for the end of a branch; `LP` for a limit point or turning point of a branch, and `HB` for a Hopf bifurcation. By selecting the `Grab` tool (`G`) and clicking the tab key, we can toggle between these different points and select each in turn to use as the starting point for new branches to be added to the diagram (via the appropriate tool from the `Run` menu). (We note that, throughout this article, we illustrate bifurcation diagrams generated in XPPAUT as described here, but plotted via Matlab. We briefly describe how to do this in the supplementary material.)

Figure 5 shows a bifurcation diagram for the system given in Equation (2.2), for fixed $k = 0.03$; this is equivalent to taking a vertical slice through Figure 2 at this value of k. Stable and unstable steady states are shown by solid and dashed lines respectively. In Figure 5, the upper (unstable) branch represents the steady state at (u^+, v^-), whereas the lower branch represents the steady state at (u^-, v^+). The two points labelled S at $(F, u) \approx (0.0049, 0.5)$ and $(0.1851, 0.5)$ were highlighted as `LP` points in AUTO, and are the two saddle-node bifurcations expected from the solid curve in Figure 2. For values of $F \lesssim 0.0049$, the only steady state is the (stable) red state at $(u,v) = (1,0)$. As we increase F through the first saddle-node bifurcation, the steady states at (u^+, v^-) and (u^-, v^+) are born but are initially both unstable.

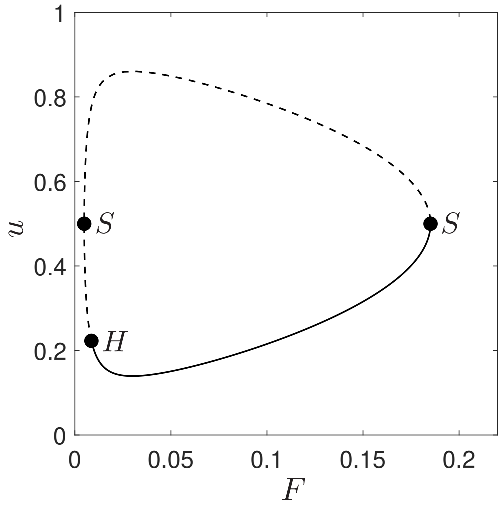
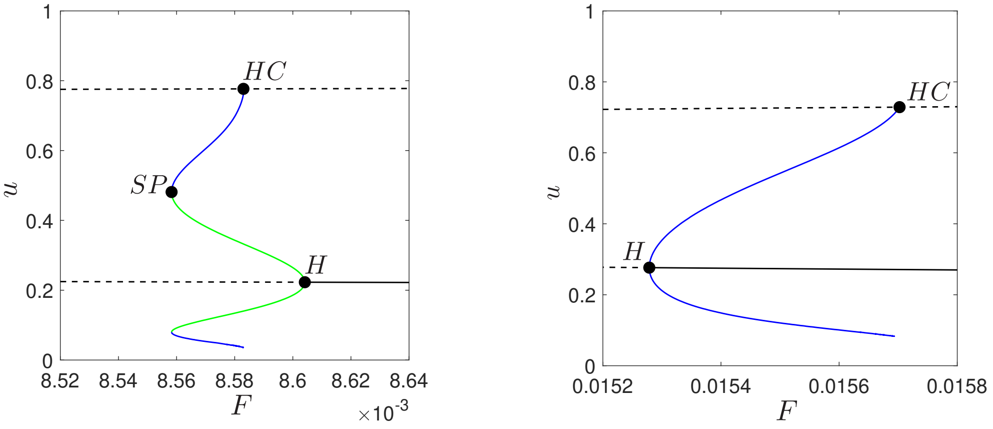


FIG. 5. *Bifurcation diagram showing u as a function of F, for $k = 0.03$. The solid and dashed lines represent stable and unstable fixed points respectively. The points labelled S are saddle-node bifurcations and the point labelled H is a Hopf bifurcation. This figure was created by importing XPPAUT data into Matlab.*

As we further increase F through the Hopf bifurcation (marked H in Figure 5), the blue state (u^-, v^+) becomes stable. The blue state remains stable until it eventually collides with the steady state at (u^+, v^-), and both are eliminated via the second saddle-node bifurcation at $F \simeq 0.1851$.

Next, we seek to understand the periodic solutions associated with the Hopf bifurcation. We use the grab tool to select the Hopf bifurcation point, and then select the `Periodic` option from the `Run` menu to plot the amplitudes of periodic orbits. Stable periodic orbits are plotted in green and unstable periodic orbits are plotted in blue. In this example, the additional branches that appear are quite steep, so it is helpful to rescale the axes to see the detail. Figure 6 illustrates the additional branches that appear for $k = 0.03$ and $k = 0.04$, revealing that the Hopf bifurcation can be either subcritical or supercritical in this model. For $k = 0.03$, the Hopf bifurcation is supercritical, as the unstable fixed point at (u^-, v^+) is surrounded by a stable periodic orbit. Here, the stable periodic orbit is itself surrounded by an additional unstable periodic orbit, for some values of F. As F is decreased through the point labelled SP in Figure 6, the two periodic orbits move toward each other and eventually collide and disappear. We refer to the point SP as a saddle-node of periodic orbits. For $k = 0.04$, the Hopf bifurcation is subcritical, as the stable fixed point at (u^-, v^+) is surrounded by an unstable periodic orbit. In both cases, we can additionally see that the unstable periodic orbit eventually interacts with the saddle point at (u^+, v^-) and is eliminated (at the point labelled HC). This point is a homoclinic bifurcation. By constructing bifurcation diagrams for a range of k values, increasing gradually from $k = 0.03$, we observe that the point SP moves toward the Hopf bifurcation (H), until eventually (at $k = 9/256$) SP and H occur simultaneously (at $F = 3/256$) such that the stable orbit never exists. Then, for subsequent increases in k, the Hopf bifurcation is subcritical. The change in the criticality of the Hopf bifurcation at $(k, F) = (9/256, 3/256)$ is known as a Bautin bifurcation [2, 8]. (Note that, in the phase planes of Figure 7 below, crossing the Bautin bifurcation corresponds to moving



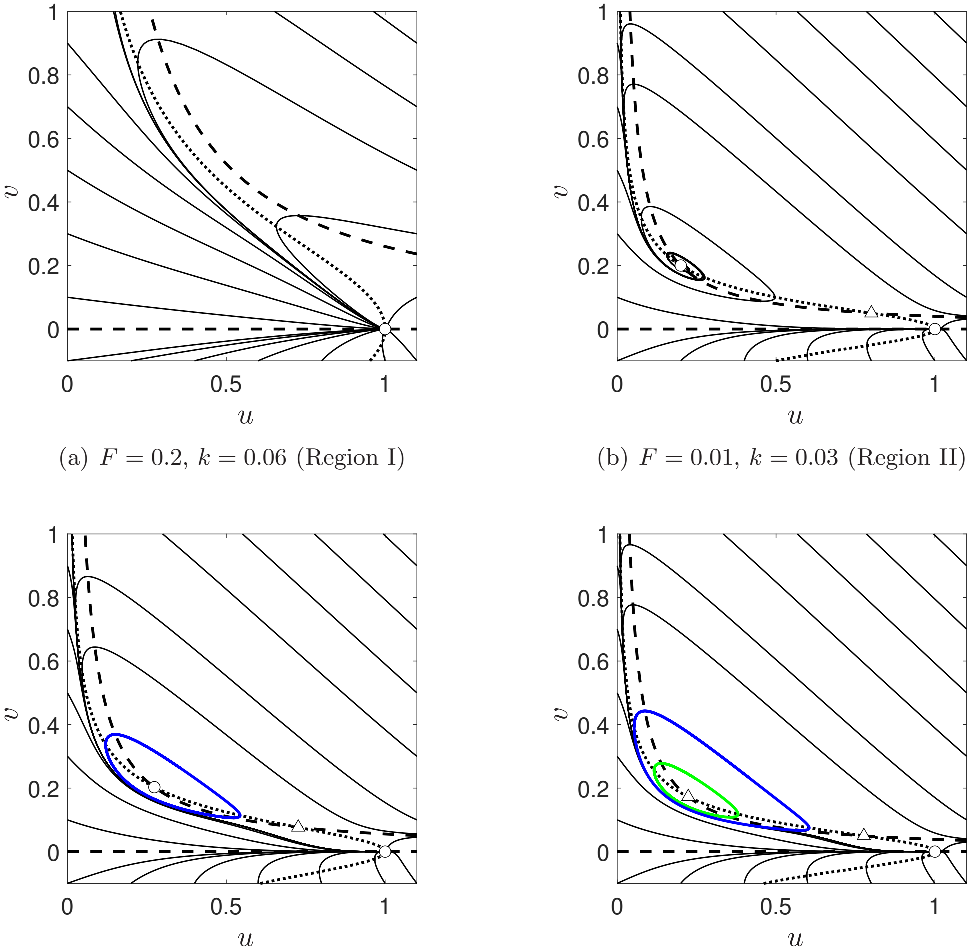
(a) $k = 0.03$ (b) $k = 0.04$

FIG. 6. *Bifurcation diagrams showing u as a function of F for (a) $k = 0.03$ and (b) $k = 0.04$, with stable and unstable periodic orbits included in green and blue respectively. The point labelled H is a Hopf bifurcation, the point labelled SP is a saddle-node of periodic orbits, and the points labelled HC are homoclinic bifurcations. Figure created by importing XPPAUT data into Matlab.*

between the phanes planes of Figure 7(c) and Figure 7(d).)

With a thorough understanding of the model's fixed points and bifurcations in place, we now briefly discuss how to generate phase planes in XPPAUT. This task is completed in XPP, rather than AUTO, so it is easiest to close any previous bifurcation analysis and restart the software to start from a 'clean' state. Upon relaunching XPPAUT, change the axis variables of the window by clicking `Viewaxis` and then `2D` (`V2`). This will bring up a menu that allows you to change the axis variables to u and v respectively, and also to change the axis size. Following this, we can quickly draw the nullclines by clicking `Nullclines` and then `(N)ew` (`NN`). Fixed points lie on the intersections of the nullclines. Now, we can draw trajectories by selecting `Initialconds` then `m(I)ce` (`II`) and then clicking anywhere on the window to generate a new trajectory starting at that point. To get a more complete picture of the direction field, click `(D)ir.field.flow` then `(S)caledDir.Fld` (`DS`) and press Enter to accept the default mesh size. Finally, to further investigate the nature of any particular fixed point, we can select `Sing pts` then `(M)ouse`, and then click close to the fixed point of interest. Various windows then appear in succession; for our purposes, it is sufficient to accept all the default options here, following which stable and unstable fixed points are marked with circles and triangles respectively.

Figure 7 shows phase planes for the homogeneous system given by Equation (2.2), for various choices of the parameters F and k chosen from the various regions of parameterspace indicated in Figure 2. In region I, illustrated in Figure 7(a) for $F = 0.2$ and $k = 0.06$, we observe that the only steady state is the red state at $(u, v) = (1, 0)$, which is stable. As we move in to region II, two additional fixed points are created: a saddle point at (u^+, v^-) and a stable spiral at (u^-, v^+), as shown in Figure 7(b,c). For some parameter choices close to the Hopf bifurcation in region II, the phase plane also exhibits an unstable periodic orbit as described above (Figure 7(c)). In region III, the fixed point at (u^-, v^+) becomes unstable, and there is scope for two periodic orbits (one stable, one unstable) to concentrically surround this fixed point, as shown



(a) $F = 0.2$, $k = 0.06$ (Region I)

(b) $F = 0.01$, $k = 0.03$ (Region II)

(c) $F = 0.0155$, $k = 0.04$ (Region II)

(d) $F = 0.00857$, $k = 0.03$ (Region III)

Fig. 7. *Phase planes for the system given by Equation (2.1), for various parameter choices. Dotted and dashed lines represent $u-$ and $v-$nullclines respectively; fixed points lie on the intersection of the nullclines. Stable and unstable fixed points are marked by circles and triangles respectively. Stable periodic orbits are shown in green; unstable periodic orbits are shown in blue. Regions indicated correspond to the regions of parameterspace illustrated in Figure 2.*

in Figure 7(d).

3. Stationary one-dimensional patterns. We now examine some of the spatial patterns that this model can emit, starting with the simplest case of a one-dimensional spatial domain. Our analysis below is predominantly a bifurcation analysis in XPPAUT (although we also supplement this with numerical simulations in Matlab in places). In order to do this, we must reconfigure our PDE model of Equation (1.2) as a system of ODEs. We do this by applying a simple discretisation, as described in a typical undergraduate numerical methods course. We make use of the following second-order finite difference approximation for the second derivative:

$$\frac{\partial^2 u}{\partial x^2}\Big|_{x=x_j} = \frac{u_{j+1} - 2u_j + u_{j-1}}{h^2} + \mathcal{O}(h^2), \tag{3.1}$$

where h is an appropriate spatial step size and subscripts denote the index of the spatial meshpoint. Substituting this into Equation (1.2), we obtain the following system of $2n$ ODEs (to leading order, having neglected the higher-order terms in Equation (3.1)):

$$\frac{\partial u_j}{\partial t} = \frac{D_u}{h^2}(u_{j-1} - 2u_j + u_{j+1}) + f(u_j, v_j), \tag{3.2a}$$

$$\frac{\partial v_j}{\partial t} = \frac{D_v}{h^2}(v_{j-1} - 2v_j + v_{j+1}) + g(u_j, v_j), \tag{3.2b}$$

for $j \in [1, n]$, where the functions f and g represent the spatially-independent reaction terms that appear in the right-hand sides of Equation (2.1). We supplement Equation (3.2) with periodic boundary conditions, which enforce that $u_0 = u_n$ and $u_{n+1} = u_1$ (and similarly for v_0 and v_{n+1}).

Before we proceed with our analysis, it is helpful to briefly clarify how concepts such as steady states and their stability should be interpreted within the spatial (PDE) context. From our undergraduate background in dynamical systems analysis, and also from the analysis performed so far in this article, we should be comfortable with the concept of steady states of an ODE system; *i.e.* a steady state of an ODE system is a set of values of the dependent variables which, when attained, results in the system remaining in this state forever. A steady state is referred to as stable if, when initialised with an initial condition close to this state, the system evolves towards it. This interpretation still applies in the PDE context; we simply have to understand that we have turned our PDE into a very large system of ODEs, with new variables representing the values that our original variables take on each of the points in the discretisation of our spatial domain. If we represent patterns by some colouring scheme, steady states with all u–variables equal and all v–variables equal would appear as a uniform colour across the whole domain. We refer to these as homogeneous steady states. Spatial patterns would appear as steady states that include some regular variation between these variables; we refer to these as inhomogeneous steady states. In order for a spatial steady state to be stable, we require that initialising our system with a 'colouring' close to the steady state pattern results in this pattern being attained in the long term. Equivalently, we start with some spatial distribution of our two chemicals, which then move around in space until they reach some (potentially non-uniform) equilibrium, giving rise to the pattern. Identification of spatial patterns in our PDE system requires that we identify inhomogeneous steady states that are stable; any steady states that are unstable are unlikely to ever be observed, either in nature, or in our analyses.

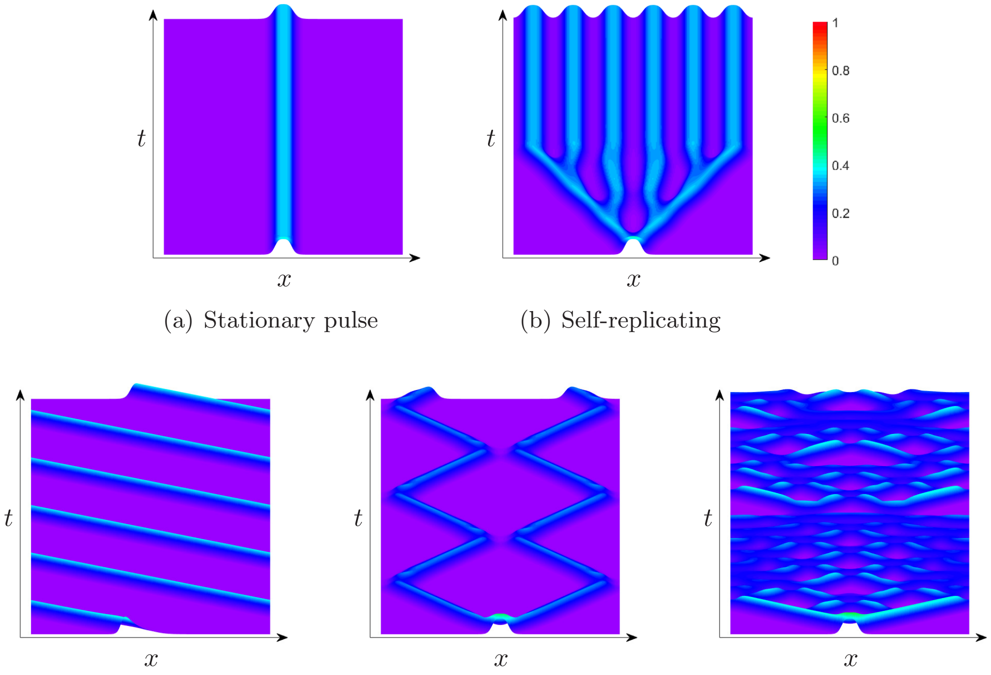
Implementation of Equation (3.2) in Matlab (code online) results in spatial patterns similar to those shown in Figure 8, including stationary and travelling pulses, self-replicating patterns and chaotic outcomes. We note, in particular, that the stationary pulses and self-replicating patterns shown in Figure 8 correspond to steady states of Equation (3.2), since there is no long-term temporal dependence. Our aim below is to understand how these patterns arise (and for which choices of parameters) via a bifurcation analysis in XPPAUT. Below, we focus, in particular, upon self-replicating patterns similar to that of Figure 8(b).

We implement Equation (3.2) in XPPAUT via the ODE file given in Algorithm 3.1. Here, we set $n = 101$ and therefore we solve a system of 202 unknowns. We also set $D_u = 0.00002$, $D_v = 0.00001$, $F = 0.04$ and choose a domain of length $L = 0.8$, which in turn implies $h = 0.008$ since $h = L/(n-1)$. Much of the ODE file is similar to that of Algorithm 2.1; however, on lines 3 and 4 we now introduce functions $f(u, v)$ and $g(u, v)$ and use these to simplify the definition of our ODEs on subsequent lines. We note that the lines starting with percentage symbols indicate the definition of a loop used to define ODEs governing the variables u_j and v_j for $j = 2, \ldots, 100$. The equations for u_1, u_{101}, v_1 and v_{101} (written here as `u1` *etc.*) are defined separately to account for the periodic boundary conditions. In this code, it is of great importance to consider the order in which our equations are presented. Here, the equation for any variable u_j is always immediately followed by an expression for the corresponding v_j; as such the code sets up a matrix problem in which odd-numbered rows correspond to u and even-numbered rows correspond to v. This results in a matrix that is of banded structure, and is hence relatively easy for the software to invert. (Defining, for example, $u_1, \ldots, u_{101}$ first, followed by $v_1, \ldots, v_{101}$ would result in a loss of this structure and code that is, at best, much slower.) On the penultimate line of Algorithm 3.1, `meth` enables us to set the integration method; `CVODE` is the recommended method for stiff systems, and by informing this solver that our problem has a banded structure, we can dramatically improve integration times. Here we set `bandlo` and `bandup` to two, since the ODE for each variable (*e.g.* u_j) refers to the variables two prior (u_{j-1}) and two ahead (u_{j+1}), together with those in between (u_j and v_j). Again, we conclude the ODE file with the command `done`.

Algorithm 3.1 ODE file used to define Equation (3.2) in XPPAUT.

```
# Gray-Scott System: 1D; Spatially Discretised; Periodic BCs
par kill=0.06 Feed=0.04 Du=0.00002 Dv=0.00001 h=0.008
f(u,v)=-u*v^2+Feed*(1-u)
g(u,v)= u*v^2-(Feed+kill)*v

u1'=f(u1,v1)+Du*(u101-2*u1+u2)/h^2
v1'=g(u1,v1)+Dv*(v101-2*v1+v2)/h^2

%[2..100]
u[j]'=f(u[j],v[j])+Du*(u[j-1]-2*u[j]+u[j+1])/h^2
v[j]'=g(u[j],v[j])+Dv*(v[j-1]-2*v[j]+v[j+1])/h^2
%

u101'=f(u101,v101)+Du*(u100-2*u101+u1)/h^2
v101'=g(u101,v101)+Dv*(v100-2*v101+v1)/h^2

@ xlo=0 xhi=10000 ylo=0 yhi=1 total=10000, dt=0.01
@ meth=cvode, bandup=2, bandlo=2,
done
```

Note that, unlike in Algorithm 2.1, initial conditions have not been included in the above ODE file. Rather than laboriously specifying the initial conditions for our 202 variables in our ODE file here, we instead import a set of initial data corresponding to a fully converged inhomogeneous steady state obtained from a typical numerical simulation in Matlab. This has the additional benefit that it increases the likelihood that



(a) Stationary pulse (b) Self-replicating

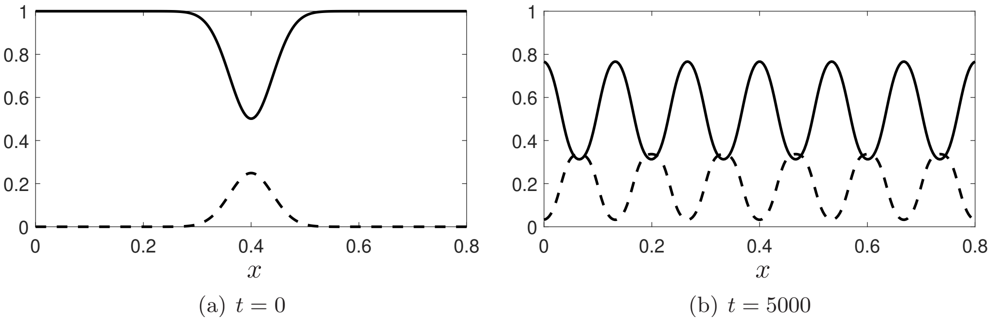
(c) Travelling pulse 1 (d) Travelling pulse 2 (e) Chaos

Fig. 8. *Five different one-dimensional patterns that arise from numerical simulations of Equation (3.2) in Matlab, showing $v(x,t)$ only. (Code online.)*

AUTO will be able to successfully track the corresponding branches of the bifurcation diagram. Figure 9 shows an initial condition used in a typical Matlab simulation, together with the resulting stable stationary pattern that is obtained via self-replication. We can use this inhomogeneous steady state as our initial conditions in XPPAUT. To import the inhomogeneous solution from Figure 9 as our initial data, we must export this from Matlab as a column vector of the form $[u_1, v_1, u_2, v_2, ..., u_n, v_n]^T$. Then, upon launching XPPAUT, we select `Initialconds` then `File` (IF) and navigate to the desired file. Upon selecting the initial data file, the XPPAUT plot of any given variable should be a horizontal line, since the initial conditions should already be at a steady state. If this is not the case, it is advisable to repeat the numerical simulations in Matlab to ensure complete convergence. Following this, we launch AUTO.

Since we are now working with a system of 202 variables, it is preferable to plot the norm of our solution vector on our bifurcation diagrams, rather than simply plotting one variable alone; we select `Axis` then `Norm` (`AN`) in AUTO to do this. We must also, once again, change the default numerical settings as described above, here choosing `Nmax` $= 400$, `Npr` $= 400$, `Ds` $= -0.02$ and `Dsmax` $= 0.02$. Choosing `Ds` to be negative here allows us to trace a branch that initially runs right to left, although AUTO will automatically adapt `Ds` should the branch double back on itself, so further adjustments to this parameter shouldn't be required. We then generate the bifurcation diagram by selecting `Run` and `Steady State` as described above.

Figure 10 illustrates a bifurcation diagram for Equation (3.2) for $F = 0.04$. In this figure, the black curve describes behaviour of the homogeneous system, as analysed



(a) $t = 0$

(b) $t = 5000$

FIG. 9. *A spatially inhomogeneous initial condition used in a typical numerical simulation in Matlab, representing a perturbation to the red state in the centre of the domain, alongside the resulting stable stationary pattern that is obtained via self-replication. Solid and dashed lines represent u and v respectively. (Code online.)*

above, and includes a Hopf bifurcation point (H) and a saddle-node bifurcation point (S). The information represented by this curve corresponds to taking a cross-section through Figure 2 at the point $F = 0.04$. Figure 10 also includes an additional branch, shown in red, which represents the inhomogeneous solution shown in Figure 9. This branch has two saddle-node bifurcation points, labelled S_1^* and S_2^* and a Hopf bifurcation point labelled H^*. (The k-coordinate of S_1^* is referred to as k_{min} in [15].) For $F = 0.04$, as we increase k through the point $S_1^* \approx 0.05774$, the stable inhomogeneous solution appears. We note that, for $F = 0.04$ the value of k at S_1^* is smaller than the value of k at H and, therefore, for $S_1^* < k < H$ the system has at least three steady states available: the red state $(1, 0)$, the blue state (u^-, v^+) and the inhomogeneous solution shown in Figure 9. However, this is not always the case. For example, repeating this analysis for $F = 0.035$ reveals that the value of k at S_1^* is larger than the value of k at H and, therefore, for $H < k < S_1^*$ the only available steady state is the red state in this case. (We omit the details here for brevity.)

As we did for the homogeneous model above, it is of interest to track the relevant bifurcation points in two-parameter-space, in order to understand where we expect to observe inhomogeneous solutions. While it was possible to do this analytically in the homogeneous case (to create Figure 2), the discretisation used in analysing inhomogeneous solutions requires us to now turn to XPPAUT to generate the corresponding two-parameter plot. First, we use the `Grab` tool to select the saddle-node bifurcation point (labelled S_1^* in Figure 10) as described above. We then navigate to `Axis` then `Two par` (AT). This will cause a window to be displayed in which we can select our parameters of interest and plotting options; in this window, we set `Y-axis:Feed`, `Main Parm:kill`, and `Second Parm:Feed`. In this case it is sensible to scale the axis such that the x–axis spans 0.03 to 0.07, and the y–axis spans 0 to 0.08. Upon changing into two parameter space, we click `Run` as before, which results in a line appearing in (k, F)–space that describes the location of S_1^*. If necessary, the lines in this plot can be extended in a similar way to as we described above, when analysing the homogeneous system, by grabbing the most recent `EP` point and selecting `Run` then, this time, `Two param`, noting also that changing the sign of `Ds` in the numerics menu can be useful in changing the direction in which the curves are extended. This process can take quite a long time, as it requires a reasonable amount of manual tuning. Once the saddle-node bifurcation curve in two parameter space is complete, we change back to

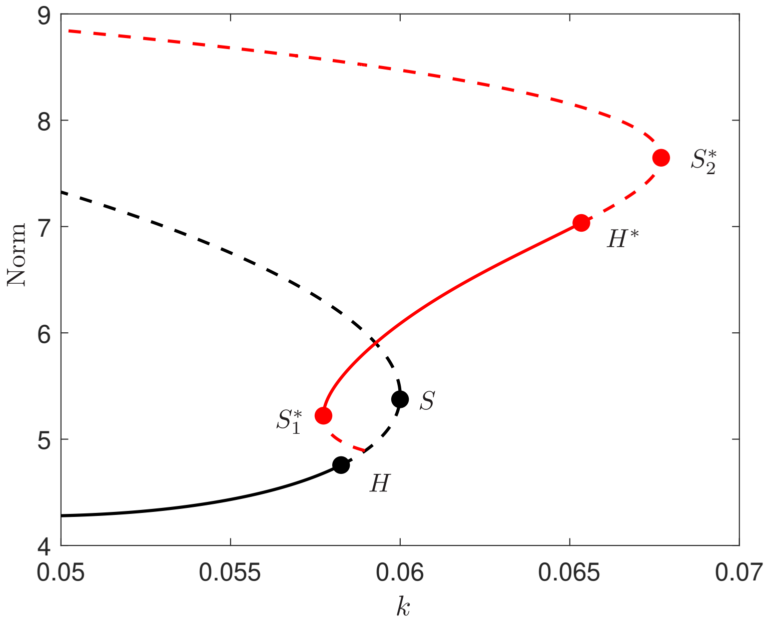


Fig. 10. *A bifurcation diagram for Equation (3.2) with $F = 0.04$, $D_u = 2 \times 10^{-5}$ and $D_v = 1 \times 10^{-5}$. Solid and dashed lines represent stable and unstable solutions respectively. The black curve represents the homogeneous solutions of Section 2, while the red curve represents a spatially-inhomogeneous steady state akin to that of Figure 9. The homogeneous branch exhibits a Hopf bifurcation at H and a saddle-node bifurcation at S, as shown in Figure 2. The inhomogeneous branch exhibits a Hopf bifurcation at H^* and saddle-node bifurcations at S_1^* and S_2^*. The spatially-inhomogeneous solution of Figure 9 is viable for choices of $k \in [S_1^*, H^*]$, where the corresponding branch is stable.*

the original bifurcation diagram by selecting `Axis` then `hI-lo`, and then repeat this process with the Hopf bifurcation point H^* and the other saddle-note bifurcation S_2^*. Adding these curves to those of Figure 2 provides Figure 11, in which black lines refer to the homogeneous system of Figure 2 and red lines refer to the equivalent curves of the inhomogeneous system; solid and dotted lines represent saddle-node and Hopf bifurcations respectively. We observe that reducing the value of F results in the Hopf bifurcation at H^* moving toward the saddle-node at S_1^* until these points eventually collide at $(k, F) \simeq (0.049, 0.020)$, at which point the Hopf bifurcation is eliminated via a Bogdanov–Takens bifurcation. For $F \lesssim 0.020$, the entire branch of inhomogeneous solutions is unstable. Conversely, increasing the value of F has the effect of moving the Hopf bifurcation towards the saddle-node at S_2^*, extending the extent of the stable portion of the inhomogeneous branch, on which we observe patterns. Eventually H^* and S_2^* collide at $(k, F) \simeq (0.068, 0.060)$ – another Bogdanov–Takens point. The stationary, inhomogeneous solutions of Figure 9 exist for all choices of F and k that lie between the two solid red lines (S_1^*, S_2^*), but these are only stable in the region above the dotted red line (H^*) in the figure.

We note that the branch of inhomogeneous solutions investigated in Figures 10 and 11 corresponds to just one of many possible self-replicating patterns permissible in this model. Firstly, note that the solution described here is translation invariant, in that moving the initial perturbation used in Figure 9(a) slightly left or right would result in the pattern of Figure 9(b) also shifting slightly left or right. While we

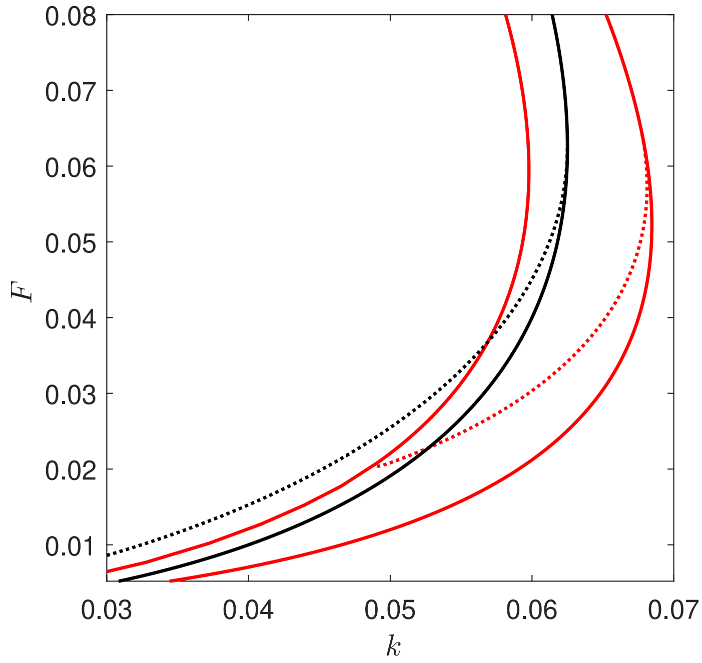


FIG. 11. *Two-parameter bifurcation diagram for* Equation (3.2) *with* $D_u = 2 \times 10^{-5}$ *and* $D_v = 1 \times 10^{-5}$*. Solid and dotted lines represent saddle-node and Hopf bifurcations respectively. Black lines refer to homogeneous solutions, while red lines refer to the stationary, inhomogeneous solution of* Figure 9*. The stationary, inhomogeneous solutions of* Figure 9 *exist for choices of* F *and* k *that lie between the two solid red lines (which represent* S_1^* *and* S_2^* *of* Figure 10*); these solutions are stable above the red dotted line (representing* H^**) and unstable below it.*

could track this new solution in XPPAUT, the curves of Figures 10 and 11 would be unchanged, since the norm of this new solution would be unaffected. Furthermore, for other parameter choices, patterns of other wavelengths also exist. Effectively, the wavelength of the solution attained is controlled by the diffusion parameters (D_u and D_v), and the existence and stability of this solution is determined according to whether this wavelength is compatible with the chosen domain size, and as a function of F and k. Branches corresponding to other self-replicating patterns on varying sizes of domain, not shown here, are investigated in more detail in [14]. We note that, in principle, we can easily trace any of these additional branches in XPPAUT by simply following the steps above, starting with an appropriately modified initial conditions file (generated *e.g.* numerically in Matlab).

4. Two-dimensional patterns. The natural next step in our analysis of the Gray–Scott model is to move to examining inhomogeneous solutions on a two-dimensional domain. Numerous previous studies have identified a diverse range of stationary patterns in 2D variants of this model. (See *e.g.* [12, 16].) In principle, analysis of this model in a 2D setting can broadly follow the steps outlined above; we begin with an understanding of the dynamics of the corresponding homogeneous problem, and then examine how the addition of two-dimensional diffusion supplements or disrupts these dynamics. In particular, a natural approach would be to consider a discretisation of our 2D domain and replicate the analysis of Section 3 above. However, it transpires that this approach is somewhat impractical, within the confines of XPPAUT at least.

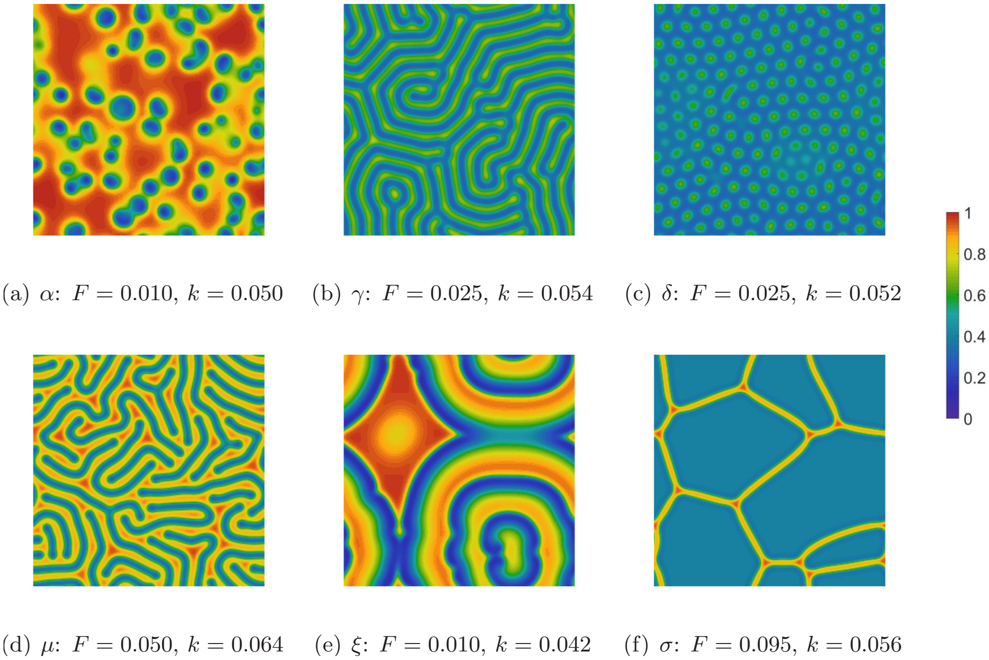
The undergraduate reader will be aware, from their introductory numerics course, that the accuracy of a numerical simulation is inherently linked to the coarseness of the discretisation used (with the spatial meshsize putting significant restrictions on the size of the corresponding timestep). In our 1D problem of Section 3, 101 spatial meshpoints were sufficient to allow numerical progress (in both XPP and Matlab) that retains the required accuracy. Moving from 1D to 2D requires a significant increase in the number of meshpoints required; roughly, in the 2D setting, we require the square of the number of meshpoints used in 1D. Unfortunately, this takes us significantly beyond the memory limitations of XPPAUT. We must therefore turn to alternative methods comprising, largely, of either by-hand analysis of 2D solutions or direct numerical implementation in *e.g.* Matlab. We give further detail on some of these approaches in the supplementary material, but briefly include an overview of results below.

A common starting point for analysis of spatial patterns is to consider those which arise via diffusion-driven (or 'Turing') instability [13, 19]. In such an analysis, we are interested in whether there are any fixed points of the homogeneous system that are stable in the absence of diffusion, but destabilised by the presence of diffusion. By following the standard undergraduate analysis for computing the stability of steady states (as outlined above), and assuming *e.g.* a travelling wave ansatz in order to appropriately linearise the diffusion term, we can compute curves in parameter space that demark changes in stability of fixed points in both diffusion-free and diffusion-driven cases. For the Gray–Scott model, we have already performed the analysis required for the diffusion-free case above. The homogeneous system has three fixed points: the red state at $(1, 0)$, which is always stable; the steady state at (u^+, v^-), which is always unstable (a saddle); and the blue state at (u^-, v^+), which is stable only in region II of Figure 2. We describe the diffusion-driven portion of this analysis in more detail in the supplementary material. This analysis reveals that introducing diffusion back in to the problem has no effect upon the stability of the red state, but there is a region of parameter space within region II in which the blue state can be destabilised by diffusion. The scale of this region is determined by the parameter $D = D_u/D_v$, the ratio of our two diffusion parameters. This region is plotted in Figure SM1; within this region we determine stationary two-dimensional patterns, including arrangements of spots (as shown in Figure SM2).

More generally, authors such as Munafo [12] and Pearson [16] have determined a wider array of both stationary and moving patterns in the Gray–Scott model. These patterns include examples of spatio-temporal chaos, worm-like branching, solitons, large-scale spirals, patterns resembling soap-bubbles, and more. Examples of some of these patterns are shown in Figure 12; these illustrations were generated numerically in Matlab, the code for which is available online. Generally, these 2D patterns are classified using the Greek letter classification system of [16]; we note that patterns of type δ include the spotted arrangements that arise via Turing instability described above. Proof of the existence and stability of these patterns, from an analytical perspective, is a topic too broad to thoroughly review here, with different approaches required for different types of patterns; for an accessible introduction to these topics, we recommend the excellent textbook of Hoyle [6].

5. Discussion.

The aim of this paper was to introduce the array of interesting behaviours and patterns exhibited by the Gray–Scott model, from a point of view of basic dynamical systems and bifurcation analysis, focusing on how problems such as this may be analysed within the software XPPAUT. We have provided a thorough



(a) α: $F = 0.010$, $k = 0.050$ (b) γ: $F = 0.025$, $k = 0.054$ (c) δ: $F = 0.025$, $k = 0.052$

(d) μ: $F = 0.050$, $k = 0.064$ (e) ξ: $F = 0.010$, $k = 0.042$ (f) σ: $F = 0.095$, $k = 0.056$

FIG. 12. *Snapshots of a selection of two-dimensional patterns obtained for $D_u = 2 \times 10^{-5}$ and $D_v = 1 \times 10^{-5}$, obtained via numerical simulation in Matlab. (Code online.) Patterns are classified according to the Greek letter naming convention used by [12] and [16].*

analysis of the homogeneous system, and have provided tasters of the analysis of spatial patterns on one- and two-dimensional domains. Our ambition here was to provide enough detail that a reader who is new to reaction-diffusion problems, or XPPAUT analysis in general, may take the ideas presented here to form the basis of their own analyses, whether these be related to PDEs or complicated systems of ODEs alike. We encourage the reader to make full use of the XPPAUT and Matlab codes that we have made available online in order to do this.

The Gray–Scott patterns examined in this paper are by no means an exhaustive list. We have understood homogeneous solutions in full, and have examined one of many similar branches of stationary, self-replicating patterns that can be attained in a one-dimensional setting. We note that, in principal, it is easy to adapt the analysis of Section 3 above to track the travelling pulses of Figure 8 in XPPAUT, since these arise as periodic orbits of the corresponding spatially-discretised system. Tracking these in XPPAUT essentially requires providing an initial condition that has accurately converged to this periodic orbit, in place of the initial conditions file used in Section 3. This will, of course, come with a need for some numerical tweaking in XPPAUT in order to track the corresponding branches accurately. In addition, analysis of two-dimensional patterns could be permissible, having performed the necessary analytical steps required to reduce the problem to a tractable ODE problem (not covered here). We leave these topics as possible targets for the reader to take forward, potentially (and hopefully) using this tutorial and our corresponding codes as a helpful stepping stone to do so.

REFERENCES

[1] P. Ball, *Forging patterns and making waves from biology to geology: a commentary on turing (1952)'the chemical basis of morphogenesis'*, Philosophical Transactions of the Royal Society B: Biological Sciences, 370 (2015), p. 20140218.

[2] J. Delgado, L. I. Hernández-Martínez, and J. Pérez-López, *Global Bifurcation Map of the Homogeneous States in the Gray–Scott Model*, International Journal of Bifurcation and Chaos, 27 (2017), p. 1730024.

[3] G. B. Ermentrout, *Bifurcation Calculations with AUTO*. http://www.math.pitt.edu/~bard/xpp/help/xppauto.html. Accessed: 2020-09-20.

[4] P. Gray and S. Scott, *Autocatalytic reactions in the isothermal, continuous stirred tank reactor: Oscillations and instabilities in the system $A + 2B \to 3B$; $B \to C$*, Chemical Engineering Science, 39 (1984), pp. 1087–1097.

[5] F. C. Hoppensteadt and E. M. Izhikevich, *Weakly connected neural networks*, vol. 126, Springer Science & Business Media, 2012.

[6] R. B. Hoyle, *Pattern formation: an introduction to methods*, Cambridge University Press, 2006.

[7] B. Krauskopf, H. M. Osinga, and J. Galán-Vioque, *Numerical continuation methods for dynamical systems*, vol. 2, Springer, 2007.

[8] Y. A. Kuznetsov, *Elements of applied bifurcation theory*, vol. 112, Springer Science & Business Media, 2013.

[9] P. K. Maini, *Using mathematical models to help understand biological pattern formation*, Comptes Rendus Biologies, 327 (2004), pp. 225–234.

[10] W. Mazin, K. Rasmussen, E. Mosekilde, P. Borckmans, and G. Dewel, *Pattern formation in the bistable Gray-Scott model*, Mathematics and Computers in Simulation, 40 (1996), pp. 371–396.

[11] D. S. Morgan, A. Doelman, and T. J. Kaper, *Stationary periodic patterns in the 1D Gray–Scott model*, Methods and applications of analysis, 7 (2000), pp. 105–150.

[12] R. P. Munafo, *Stable localized moving patterns in the 2-D Gray-Scott model*, arXiv preprint arXiv:1501.01990, (2014).

[13] J. Murray, *Mathematical Biology II. Spatial models and biomedical applications*, Springer, 2003.

[14] Y. Nishiura and D. Ueyama, *A skeleton structure of self-replicating dynamics*, Physica D: Nonlinear Phenomena, 130 (1999), pp. 73–104.

[15] Y. Nishiura and D. Ueyama, *Spatio-temporal chaos for the Gray–Scott model*, Physica D: Nonlinear Phenomena, 150 (2001), pp. 137–162.

[16] J. E. Pearson, *Complex patterns in a simple system*, Science, 261 (1993), pp. 189–192.

[17] R. Seydel, *Practical bifurcation and stability analysis*, vol. 5, Springer Science & Business Media, 2009.

[18] S. H. Strogatz, *Nonlinear dynamics and chaos with student solutions manual: With applications to physics, biology, chemistry, and engineering*, CRC press, 2018.

[19] A. Turing, *The chemical basis of morphogenesis*, Philosophical Transactions of the Royal Society of London. Series B, Biological Sciences, 52 (1952), pp. 37–72.