Imperial College London

Engineering and Physical Sciences Research Council

A COMPUTATIONAL APPROACH TO ENGINEERING SYNTHETIC TURING PATTERNS

Martina Oliver Huidobro

Supervisors: Professor Robert Endres and Professor Mark Isalan

Project Thesis MRes Systems and Synthetic Biology

Department of Life Sciences

Imperial College London

20 August, 2020

Abstract

Turing patterns are self-assembling patterns shaped as dots, stripes or laberynths that arise naturally from an initial homogeneous state. Turing patterns are commonly thought to be the explanation for many developmental processes and might be the solution for the synthetic production and control of patterned tissues or organoids. Genetically engineered tissues capable of producing Turing patterns are key to achieve both a better understanding of the mechanism and to control these patterns for downstream bioengineering applications. Due to the complexity of the networks and the limited parameter space that leads to Turing pattern formation, mathematical models need to be leveraged to guide the design and tuning of these genetic circuits. In this study, a detailed mathematical model describing each species of a novel synthetic circuit, potentially capable of Turing pattern formation, is developed to understand the system's behaviour. The parameter space of this system is explored using latin-hypercube sampling, where each sample is analysed through linear stability analysis to obtain information on its patterning capabilities. Some parameter sets leading to the formation of Turing patterns are found. Other patterning behaviours are observed such as unstable Turing patterns or oscillatory patterns. Finally, characteristics such as the shape of the pattern and the speed of pattern formation are studied as well as their relationship with the dispersion relation peak height. Faster forming patterns and dotted patterns are shown to be linked to a higher dispersion relation peak. This novel discovery could enable tuning of parameters experimentally to control the shape and the speed of pattern formation.

Contents

1 Introduction

1.1 Patterning in nature

Everywhere we look around in nature, the presence of patterns can be observed. This accounts both for non-living and biological systems. In the case of biological systems, the acquisition of patterns can potentially result in an evolutionary advantage, and therefore lead to the selection of those genes responsible for the advantageous pattern mechanism (J. D Murray 2002).

From simpler 2D skin patterns to more complex 3D pattern such as human organs, there is a wide range of examples of patterning in biology, where cells coordinate themselves to form more complex structures that provide a biological advantage. Amongst those examples are included some such as: Fractal shaped bacterial colonies, to maximise nutrient intake (Matsushita and Fujikawa 1990); Animal stripes to break an animal's outline making it harder for the predator to spot it (Stevens et al. 2006); butterfly eyespots in their wings, making them look like they are part of a larger animal (Blest 1957); or even spirals in phylotaxis, where certain divergence angles such as the golden angle, allow for spiral formation and optimisation to light exposure (Strauss et al. 2020).

1.2 Types of patterning mechanisms

There are many types of mechanisms, which are responsible for these types of patterns mentioned above and many more. Some of them are: differential adhesion, mechanical instabilities such as wrinkling and branching or diffusion based mechanisms (Košmrlj 2020). Two of the most influential theories in the field of morphogenesis, which try to explain the processes controlling spatial distribution of cells during embryonic development, are based on diffusion mechanisms: Lewis Wolpert's positional information, and Alan Turing's reaction-diffusion (RD), which gives rise to the very well known Turing patterns (Green and Sharpe 2015). In Lewis Wolpert's positional information, also known as the French flag model, a prior morphogen gradient is present in the tissue. Cells can measure the concentration of morphogen and respond to it, therefore having a positional identity that determines how they behave or differentiate (Wolpert 1969). Turing's reaction diffusion mechanism consists of a network of interacting and diffusing species that can generate a pattern starting from a homogeneous tissue.

Turing's theory was first eclipsed by Wolpert's theory due to the complexity of Turing's model. Turing's theory was non-intuitive and had differential equations involved, which lead to biologists ignoring it to follow a simpler theory that could explain some of the morphogenesis phenomena. However, as our knowledge on molecular biology advanced, limitations and pitfalls of Wolpert's French flag model were realised. Firstly, the system would have to depend on earlier stable heterogeneities that sometimes are not physically possible (J. D Murray 2002). Furthermore, not as many patterns could be attributed to this mechanism as we could to RD systems (Green and

Sharpe 2015). Because RD is one of the best-known theoretical models that explains self-regulated pattern formation in the developing animal embryo (Kondo and Miura 2010), it has become the main focus on scientists studying morphogenesis.

1.3 Turing patterns

Reaction-diffusion theory was first introduced in 1952 in the article "The chemical basis of morphogenesis", where Alan Turing proposes a new theory to explain the phenomena of morphogenesis. This theory argues that pattern formation can arise from two interacting species diffusing at different rates through a tissue (See Figure 1b). He called these interacting and diffusible molecules morphogens and suggests that these morphogens could potentially be hormones, skin pigments or even genes (although less likely). The example used in Turing's paper is composed of two simple equations consisting of an activation term, and inhibition term and a degradation term, modelled through the law of mass action, where the rate of reaction is directly proportional to the concentration of reactants (See Figure 1a) (Turing 1952). In the RD theory, the system is initially homogeneous. Upon a perturbation like biological noise, the reaction rates change and this perturbation brings the system out of steady state. Finally, the system settles into another steady state which is heterogeneous, potentially forming a pattern.

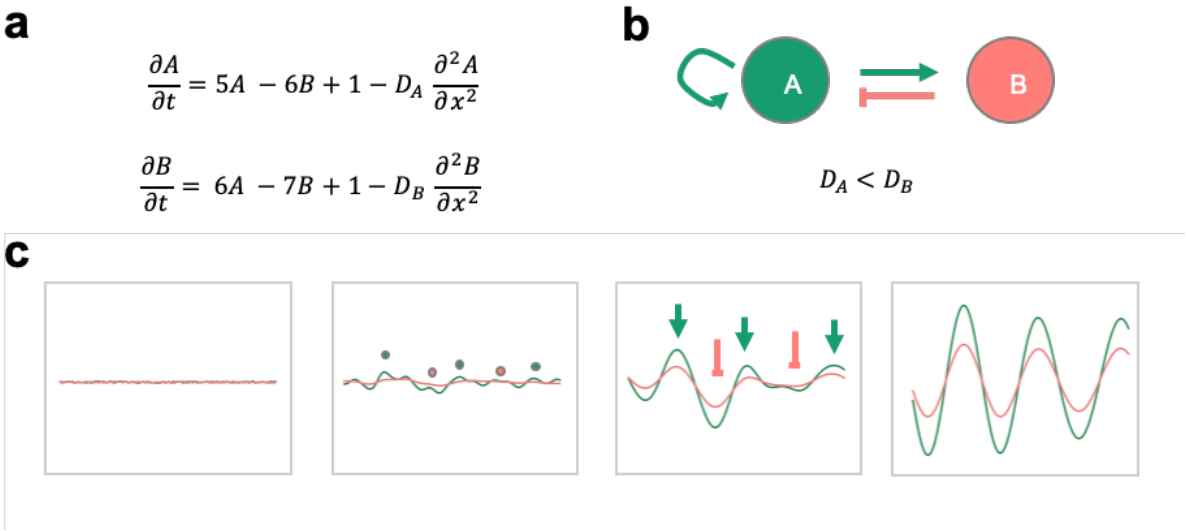


Figure 1: Reaction-diffusion Mechanism. A) Turing's proposed equations for a patterning reaction-diffusion system . **B)** Turing's proposed network for a patterning reaction-diffusion system with a slow activator (green) and a fast inhibitor. The activator shown in green and the inhibitor shown in pink. The diffusion rate of the activator is smaller, than that of the inhibitor. **C)** Gierer-Meindhardt's short-range activation and long-range inhibition (SALI) mechanism for pattern formation. The x axis is space and y axis is concentration of activator (green) and inhibitor (pink). Green and pink arrows indicate prevalent activator and inhibitor respectively.

As mentioned previously, Turing's RD system was initially ignored due to its complexity, which meant biologists opted to study simpler systems as Wolpert's French flag model. However, Gierer and Meinhardt 1972 provided biologists with a more intuitive way to understand Turing patterns. Turing's model was developed in the context of two very simple equations that could not be applied to biological systems (e.g. morphogens could even reach negative concentrations). Alternatively, Gierer and Meindhardt proposed that Turing patterns can be simply achieved by having short-range activation and long-range inhibition (SALI). This way, pattern formation is independent of Turing's equations and therefore independent of number of reagents, how the activation or inhibition occurs and even how the signal is transduced. For example, inhibition can now be replaced by depletion of the morphogen; and diffusion by an activator or inhibitory signal from cell to cell (Gierer and Meinhardt 1972). This new definition of the requirements for Turing pattern formation was key in relating Turing patterns to real morphogenetic phenomena occurring in living systems: It allowed developmental biologists to study Turing patterns using non-linear interactions and to introduce interactions occurring in molecular biology. All of these other variants to the original RD model highlighted the potential of RD models in developmental biology (Hans Meinhardt and Alfred Gierer 2000).

The SALI mechanism proposed by Meindhard for the development of patterns from a homogeneous state is explained in Figure 1c. The system starts in a homogeneous steady state, with two morphogens A and B that react with each other and diffuse at different rates. Upon a perturbation such as biological noise, there are local places where the concentration of one species is prevalent over the other one. This can be seen in the second panel of Figure 1c, where the dots represent which species is prevalent over the other one. This prevalence of activator leads to local production of the activator and the inhibitor. The activator, which diffuses slowly travels a short range, while the inhibitor spreads more through space due to its faster diffusion rate. This leads to short range activation and long range inhibition. Finally, this results in activation peaks and inhibition troughs that settle into a steady state, which correspond to a stationary pattern (Figure 1c, panel 4) (Gierer and Meinhardt 1972).

1.4 Analysing reaction-diffusion systems

Complex systems such as RD systems cannot typically be solved analytically. Therefore, other methods need to be used to understand whether a system is capable of forming Turing patterns

or not. These two methods are numerical solvers or linear stability analysis. Each one has its own advantages and disadvantages and hence both are combined for a more efficient study of patterning systems. Numerical methods provide a solution in time and space of the concentrations of each specie, allowing to visualise a pattern once it has converged into steady state. However, this approach is very computationally expensive. The second method is to study the stability of the system with and without diffusion using linear stability analysis. This method only shows if a system can form Turing patterns, but no information is obtained on pattern development over time or shape. Although less information is acquired through linear stability analysis, it is far less computationally expensive, meaning it can be scaled up to study a greater number of systems than with numerical methods (J. D Murray 2002).

Linear stability analysis is used to study the stability of the system under different specific conditions. Instead of obtaining a solution, we will obtain qualitative information about the stability of the system (Glendinning 1994). For Turing pattern formation, the linear stability analysis profile must be the following: The steady state must be stable without diffusion, which makes any perturbations decay over time. When diffusion is added to the system, that steady state becomes unstable so any perturbations will be amplified over time. This amplification of the perturbations leads to the formation of the pattern (Turing 1952). The amplification or decay of the perturbation can be described with the following equation:

$$\delta X = X_0 e^{\sigma t} \cdot e^{ikx} \tag{1}$$

In this formula, δX is a perturbation around the steady state at time t. $X_0 e^{\sigma t}$ represents the amplitude of the perturbations, being X_0 the amplitude of the perturbation at $t = 0$. On the other hand, e^{ikx} represents the spatial oscillations. Looking at the first part of Equation 1, we can observe that σ is responsible for the exponential growth or decay of the perturbation's amplitude, depending on its sign. k being the wavenumber, will be zero with no diffusion and a positive term with diffusion. The dispersion relation is the relationship between σ and k. For a Turing instability to occur, the dispersion relation must be negative with $k = 0$ (stable steady state with no diffusion) and positive for $k > 0$ (unstable steady state with diffusion) (J. D Murray 2002; Schneider 2012). An example of the dispersion relation of a Turing instability is shown in Figure 2. This simplification of linear stability analysis is explained in more detail in the Methods section.

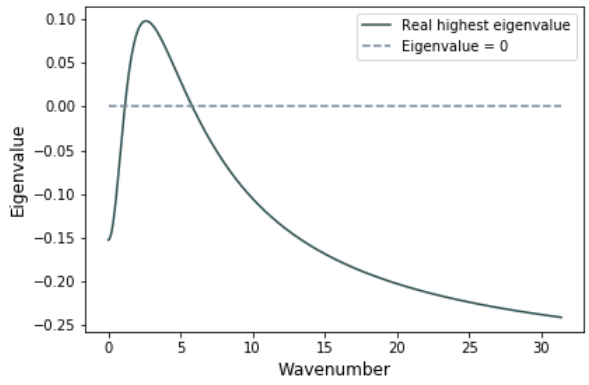


Figure 2: Dispersion relation of a Turing Instability. The x axis corresponds to the wavenumber (k), and the y axis corresponds to the real part of the highest eigenvalue from the matrix of the linearised system.

Using this approach, the system is linearised around the steady state, meaning the equations are easier to handle. However, any conclusions drawn from it only work if the system never deviates far from the original steady state. The assumption must be made that patterns produced in the early stage (close to steady state), have a strong qualitative similarity to those appearing in the later stages (Turing 1952).

Linear Stability Analysis is used for sampling large parameter or topological spaces, while numerical solvers are used to study a specific system and parameter set in depth.

1.5 Biological examples of Turing patterns

Evidence has been found to show that RD systems can generate patterns such as those found in nature. Some patterns have been replicated by simulating RD models, such as seashell pigmentation and fish patterns. In the case of zebrafish stripes, disposition of feather buds in chick and hair follicles in mice, the dynamic properties have been explored and are consistent with computer simulations (e.g. pattern regenerates in specific manner after being physically disrupted, as predicted by the model) (Kondo and Miura 2010). Finally, molecules known to be responsible for patterning in some organisms have been shown to be involved in SALI networks. This is the case for Nodal & Lefty, which are responsible for vertebrate assymetry (Nakamura et al. 2006); Bmp, Sox9 & Wnt control digit patterning in limb development (Raspopovic et al. 2014); and finally, Wnt & Dkk, are involved in lung branching in vertebrates (De Langhe et al. 2005) and hydra head regeneration (Augustin et al. 2006). Many other patterns not mentioned above have also been shown to be in someway related to RD systems. As seen above, some cases contain more evidence than others, but overall a relationship between patterning and RD systems is present.

Involvement of these molecules in patterning does not directly prove it is an RD system, however it strongly suggests it is an underlying mechanism for these self-regulated and self-regenerative patterns. Proving the role of RD mechanisms in these examples is a challenge due to the tangled nature of the system, with an extremely high number of interactions involved. Because it is almost impossible to validate Turing's proposal in developmental processes, a shift from a top-down method to a bottom-up synthetic biology approach is needed. The aim of this bottom-up approach is to build a tunable RD network using genetic orthogonal parts, which generates spatial patterns in a tissue culture. Artificial Turing patterns in cell culture will not only demonstrate the importance of RD systems in morphogenesis, but will also enable the synthesis of tissues tissues with tunable patterns for biotechnological applications (Basu et al. 2005; Chenli Liu et al. 2011; Sekine et al. 2018).

1.6 Engineered Turing systems

Experimental approaches to engineering self-organising Turing systems have been already presented in the literature, although some improvement still needs to be made. This examples include both engineered chemical RD systems and engineered biological RD systems.

The first engineered RD system, in 1990, was a chemical system involving a chlorinedioxide/iodine/malonic acid reaction (CIMA reaction). The system was highly tunable and involved differential diffusivity between morphogens (Castets et al. 1990; Lengyel and Epstein 1992). Later, other chemical Turing patterns have been engineered such as the FIS and the TuIS systems (Horváth et al. 2009).

Biological implementations of RD systems have not been as successful because of the complexity of biological systems, including unknown regulatory functions, delays and biological noise. One of the first attempts of engineering Turing patterns using biological parts was carried out by Karig et al. 2018. Using two orthogonal interacting morphogens (IC4HSL, A3OC12HSL), a pattern generator was successfully engineered in bacteria. However the experimental patterns varied in size, shape, fluorescence intensity and intervals between them. Two models were generated to explain the experimental data: a stochastic model and a deterministic model. The experimental data corresponded to the stochastic model, suggesting that the patterns generator

exhibited stochastic Turing pattern formation. Sekine et al. 2018 carried out a similar study where they built the first eukaryotic and protein based pattern generator using Nodal and Lefty. However the patterns generated experimentally could again not be matched to a deterministic Turing model and were less regular in size and shape compared to those of deterministic Turing patterns. Furthermore, when the initial conditions were altered, the pattern observed varied; whereas in classical deterministic Turing patterns, the mechanism of pattern formation is independent of the initial condition. Considering the behaviour of the patterns, they were classified as solitary localised structures and not Turing patterns. Although some sort of pattern was obtained in both papers, none of them could claim to have developed a Turing pattern generator. Another issue related with this paper was that the nodal-lefty network, which exhibits SALI behaviour, had been previously observed in nature and is thought to be responsible for certain patterning phenomena such as left-right asymmetry in vertebrates (Nakamura et al. 2006). Because the network previously existed in nature and is not built from basic independent parts, the mechanism responsible for their differential diffusivities or the nature of their interaction is not fully understood. This makes the components almost untunable and therefore fails to fulfil the standards expected in synthetic biology.

Although great improvement has been done in the biological implementations of RD patterning systems, there is still work to do to obtain a tunable Turing pattern generator, that is validated by a deterministic RD model (Scholes and Isalan 2017). The work done in this study focuses on the development of a new synthetic circuit capable of generating Turing patterns *in vitro*.

1.7 Circuit 3954, a Turing pattern generator

As previously mentioned, engineering biological RD systems is extremely challenging and this can overall be attributed to two main challenges. Firstly, there is a lack of orthogonal components, specifically diffusers, required for building an RD system. Second and most importantly, RD systems are not robust meaning very few conditions (experimentally) or parameters (mathematically) will produce a Turing behaviour (Scholes et al. 2019)

The issue of finding the diffusers required for building an RD system, was partly resolved with the appearance of the Marionette system in Meyer et al. 2019. In this paper, 12 new orthogonal small-molecule sensors were introduced, which present low background, high dynamic

range, high sensitivity and low cross-talk. This Marionette molecule-receptor system is an optimal tool for the independent control of gene expression in synthetic circuits. Two out of the diffusers used in the circuit implemented below, HSL and PCA, were found in this paper. Additional diffusers have recently appeared in the literature that could potentially be tested in this circuit.

The issue of limited robustness in RD systems, has been partly addressed by several research studies which predict the robustness of different network topologies for Turing pattern formation (Marcon et al. 2016; Scholes et al. 2019; Zheng et al. 2016). High throughput studies were carried out analysing different network topologies under different parameter sets. Intracellular robustness was explored for all networks, defined as the fraction of parameter combinations that is capable of Turing pattern formation. In Scholes et al. 2019 and Zheng et al. 2016 2 node and 3 node topologies were explored, showing that 3 node topologies exhibit an overall higher robustness. In Scholes et al. 2019, other two types of robustness were defined including extracellular robustness (robustness of Turing network to changes in diffusion constants) and topological robustness (fraction of networks one edge away that have the capability of producing Turing I instabilities); and finally total robustness as the product of the three. Although all three papers were relevant, Scholes et al. 2019 was used to select a robust network to implement experimentally. Surprisingly, 61% of the 7625 networks tested in Scholes et al. 2019 were able to produce Turing patterns. However, less than 0.1% of the parameter space overall supported Turing patterns. This shows that Turing patterns are common but not robust, and therefore topologies need to be carefully designed to maximise robustness. To summarise this study, the 3 node network 3954 (Fig. 1a) was the non-competitive network found with highest robustness and was therefore chosen for biological implementation. Its intracellular robustness being 0.015, extracellular 0.277 and its topological robustness 0.938. In principle, this network has a higher parameter flexibility than other networks, it allows equal diffusivity (although with reduced robustness) and slight changes can be made to the topology still maintaining its Turing potential.

The theoretical model describing 3954 circuit by Scholes et al. 2019 is shown in Figure 3a and described with the equations below:

$$\frac{\partial [A]}{\partial t} = b_A + V_A \cdot \frac{1}{1 + \left(\frac{[k_{aa}]}{A}\right)^{n_{aa}}} \cdot \frac{1}{1 + \left(\frac{[B]}{K_{ba}}\right)^{n_{ba}}} - \mu_A \cdot [A] + D_A \cdot \frac{\partial^2 [A]}{\partial x^2} \tag{2a}$$

$$\frac{\partial[B]}{\partial t} = b_B + V_B \cdot \frac{1}{1+\left(\frac{[k_{ab}]}{A}\right)^{n_{ab}}} \cdot \frac{1}{1+\left(\frac{[C]}{K_{cb}}\right)^{n_{cb}}} - \mu_B \cdot [B] + D_B \cdot \frac{\partial^2[B]}{\partial x^2} \tag{2b}$$

$$\frac{\partial[C]}{\partial t} = b_C + V_C \cdot \frac{1}{1+\left(\frac{[k_{cc}]}{C}\right)^{n_{cc}}} \cdot \frac{1}{1+\left(\frac{[A]}{K_{ac}}\right)^{n_{ac}}} \cdot \frac{1}{1+\left(\frac{[B]}{K_{bc}}\right)^{n_{bc}}} - \mu_C \cdot [C] \tag{2c}$$

Having a tool for fine-tuning of gene expression, and a network topology which is relatively robust, were two necessary steps for building a tunable synthetic network that successfully yields Turing patterns. This work was done by PhD student Jure Tica in Professor Isalan's group, who converted Scholes theoretical 3 node network (Figure 3a) into a synthetic biology network (Figure 3b).

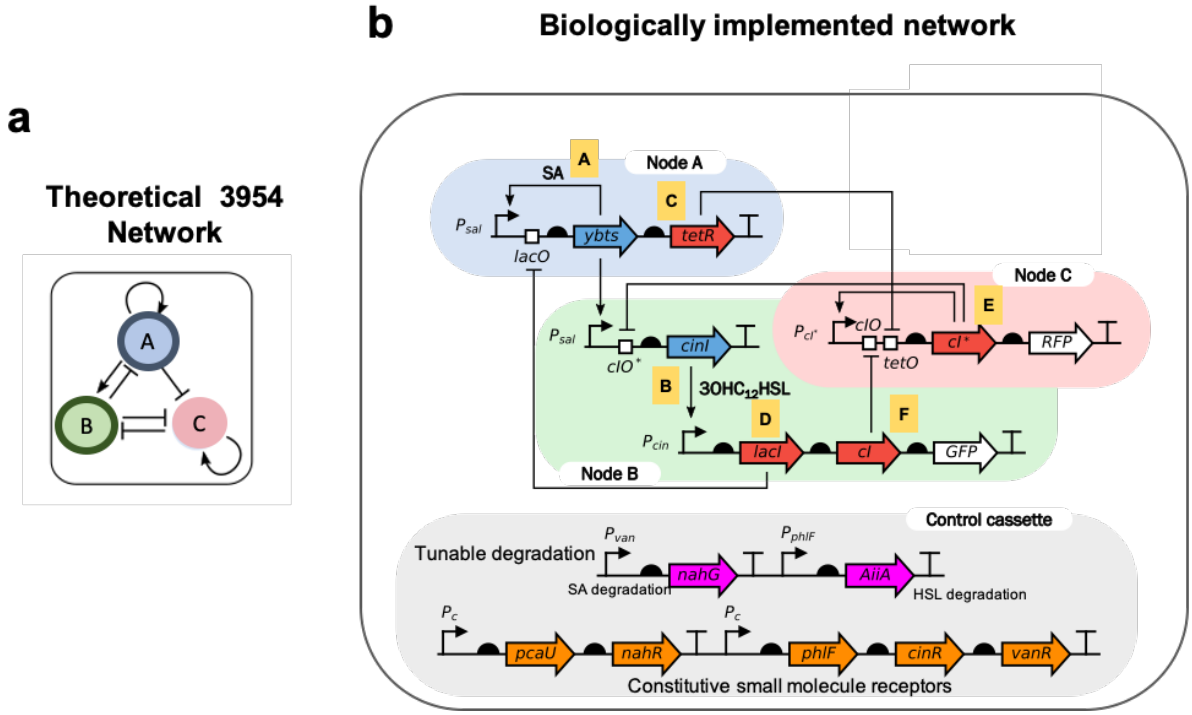


Figure 3: Circuit 3954. **(A)** Theoretical network 3954 developed by Scholes et al. 2019 with three species. Species A and B are diffusing molecules and species C is static. **(B)** Biological implementation of network 3954 with six species. Genes producing the diffusing SA and HSL, shown with blue arrows. Genes producing the non-diffusing species TetR, LacI, cI and cI* are shown with red arrows. The three molecules from the theoretical network (A) are represented as nodes in the biological implementation (B). The control cassette shows the potential tunability of the circuit in terms of degradation of SA and HSL, with regulated expression of nahG and AiiA degradation enzymes. The constitutive small molecule receptors, always expressed, are receptors needed for the binding of small molecules (PCA, SA, DAPG, HSL and Vanillin) to their respective promoters.

Due to the lack of components with the specific regulatory functions needed to build network 3954, several components had to be used for a single node, leading to a more complex circuit

than the proposed theoretical one. The 3 node network abstracted to a synthetic network has 6 genetic components as seen in Figure 3b. Although the synthetic 6 component network is expected to produce patterns as it is based on the 3 node network described by Equations 2, no model has been studied describing this synthetic implementation. Such a complex network, which include delays and non-linear relations between the components could potentially result in a different behaviour than the theoretical 3 node network. In this thesis, a new model describing the biological implementation of 3954 will be developed. The model will show whether the synthetic biology network is a Turing pattern generator.

Although more complex, the advantage of this genetic circuit is the ability to tune certain parameters, therefore making it easier to hit Turing space. The tuning can be done chemically or genetically. Chemical tuning, which is experimentally simpler, involves adding a chemical substance to the solution which will affect the parameters of the model. Chemically, the degradation of SA or HSL can be increased by adding Vanillin or DAPG respectively. The repression by LacI or TetR can also be decreased by adding IPTG or aTc to the medium. Finally, we can also add SA or HSL to the solution which will change starting concentrations of the system, and might affect the final solution. Genetical tuning consists on mutating promoters, changing plasmid copy number and other techniques that can affect regulation or promoter leakiness, which can be linked back to parameters such as Michaelis-Menten constants or basal production rates. An in-depth study of what parameter values allow Turing pattern formation is needed, to understand the parameter space and guide the experimental tuning to achieve synthetic Turing patterns *in vitro*.

1.8 Project aims

This project is composed of several aims:

- The first one is to define a new model with 6 equations that can describe the behaviour of the biological implementation of circuit 3954, in terms of its six components. The 3 equation model does not describe in detail the behaviour of each species over time making it not accurate and hard to link to any experimental data obtained from the circuit.
- Secondly, the model will be exploited to understand whether this synthetic biology implementation of the 3 node system has the potential to produce Turing patterns. This is

fundamental to continue the experimental development and testing of this circuit.

- If the synthetic network is a Turing pattern generator, the distribution of parameter values responsible for Turing pattern formation or other types of patterns will be studied, to detect any parameter bias leading to pattern formation.
- Furthermore, features of each pattern found will be studied such as parameter robustness, pattern wavelength, pattern shape and speed of pattern formation.
- Finally, the relationship between the height of the dispersion relation peak and these features will be explored.

2 Results

First of all, a new model that describes the biological implementation of circuit 3954 will be defined, including 6 equations. This model will be then analysed to understand its Turing pattern forming capabilities and a general overview of the parameter space. Robustness will be analysed for the parameter sets capable of Turing patterning. Finally, it will be studied how a higher peak in the dispersion relation affects patterning in terms of shape and speed of pattern formation.

2.1 Model definition of the 3954 biologically implemented network

As mentioned in the introduction, the biological implementation of circuit 3954 results in a more complex network than the one originally described by Scholes et al. 2019. The theoretical model is composed of only three nodes and is described using only 3 equations. However, the new implemented network contains 6 regulators (molecules responsible for activation or inhibition of gene expression). Therefore, to obtain an accurate model that can be linked to the experimental data and that describes the behaviour of the system, all 6 regulators must be modelled. A 3 equation system would not suffice as it would not represent the complex interactions between all species in the system, and fitting experimental data would be very complex or require too many assumptions. This results in a more complex model than the one defined by Scholes et al. 2019

The model consists of partial differential equations describing the change in concentration of species at a specific time and space point. The following terms are used to describe this system: The basal production term describes the production of the species when no activation

or inhibition is present. The non-competitive activation or inhibition term refers to the species production dependent on activation or inhibition acting at a promoter level. The degradation refers to how many molecules are degraded over time and it is modelled with a linear term. Finally, the diffusion contains the spatial factor of the equation, and it refers to the change in species concentration as they move around in space. All of these terms combined will give the change in species over time in a specific point in space. By applying this modelling framework to the biological implementation of network 3954, the following system of equations is obtained:

$$\frac{\partial[A]}{\partial t} = b_A + V_A \cdot \frac{1}{1+\left(\frac{[k_{aa}]}{A}\right)^{n_{aa}}} \cdot \frac{1}{1+\left(\frac{[D]}{K_{da}}\right)^{n_{da}}} - \mu_A \cdot [A] + D_A \cdot \frac{\partial^2[A]}{\partial x^2} \tag{3a}$$

$$\frac{\partial[B]}{\partial t} = b_B + V_B \cdot \frac{1}{1+\left(\frac{[k_{ab}]}{A}\right)^{n_{ab}}} \cdot \frac{1}{1+\left(\frac{[E]}{K_{eb}}\right)^{n_{eb}}} - \mu_B \cdot [B] + D_B \cdot \frac{\partial^2[B]}{\partial x^2} \tag{3b}$$

$$\frac{\partial[C]}{\partial t} = b_C + V_C \cdot \frac{1}{1+\left(\frac{[k_{ac}]}{A}\right)^{n_{ac}}} \cdot \frac{1}{1+\left(\frac{[D]}{K_{dc}}\right)^{n_{dc}}} - \mu_C \cdot [C] \tag{3c}$$

$$\frac{\partial[D]}{\partial t} = b_D + V_D \cdot \frac{1}{1+\left(\frac{[k_{bd}]}{B}\right)^{n_{bd}}} - \mu_D \cdot [D] \tag{3d}$$

$$\frac{\partial[E]}{\partial t} = b_E + V_E \cdot \frac{1}{1+\left(\frac{[C]}{K_{ce}}\right)^{n_{ce}}} \cdot \frac{1}{1+\left(\frac{[F]}{K_{fe}}\right)^{n_{fe}}} \cdot \frac{1}{1+\left(\frac{[k_{ee}]}{E}\right)^{n_{ee}}} - \mu_E \cdot [E] \tag{3e}$$

$$\frac{\partial[F]}{\partial t} = b_F + V_F \cdot \frac{1}{1+\left(\frac{[k_{bf}]}{B}\right)^{n_{bf}}} - \mu_F \cdot [F] \tag{3f}$$

The species modelled in the equations are shown in Table 1, and the model parameters in Table 2.

Table 1: Model species

Variable	Molecular specie
A	Salicylate (SA)
B	Homoserine Lactone (HSL)
C	TetR
D	LacI
E	cI*
F	cI

SA and HSL are both signalling molecules with small molecular weights. This makes them good morphogen candidates as they can regulate gene expression and diffuse through tissues. TetR, LacI, cI and cI* are repressor proteins that will act as regulators in the circuit. Their diffusion is negligible as they have a bigger molecular weight.

Table 2: Model parameters

Parameter	Description	Units	Parameter range
b_x	Background production rate	nM/h	0.01-50
V_x	Induced maximum production rate	nM/h	10-1000
Kxy	Michaelis Menten constant	nM	0.1-250
nx	Cooperativity constant	no units	2
μ_x	Degradation rate	1/h	0.001-50
D_x	Diffusion rate	mm^2/h	$D_A, D_B = (0.0789, 0.0511)$

Ranges for parameters b_x, V_x, $K_{x,y}$ and μ_x shown in Table 2 were estimated from the literature (Pušnik et al. 2019; Scholes et al. 2019). Because higher cooperativities leads to increased robustness in Turing pattern space (Diambra et al. 2015), cooperativity was set to 2 to assume the worse case scenario were robustness is more limited. Diffusion rates were set to 0.078 and 0.051 for SA and HSL respectively. These rates were estimated by PhD student Jure Tica by fitting a diffusion model to experimental diffusion data of the two molecules diffusing on a bacterial lawn.

To reduce the number of parameters, some assumptions have been made on the Michaelis-Menten (k) constants and the degradation rates (μ). For Michaelis-Menten constants, all genes under the same promoter will have the same k parameter for their activation and inhibition respectively. This means we can assume $k_{aa} = [k_{ab}, k_{ac}]$, $k_{bd} = kbf$ and $k_{da} = k_{dc}$. In terms of degradation rates, all 4 non-diffusing species have the same degradation tag: the LVA tag (Andersen et al. 1998). Therefore, we can assume the same degradation rate, so $\mu_{LVA} = [\mu c, \mu d, \mu e, \mu f]$. After these assumptions have been made, the number of parameters to explore are 22, making the system a 22 dimensional parameter space.

This new derived model is in theory more accurate to describe the biological circuit 3954 implemented experimentally, as it takes into account all 6 components of the system. Firstly, this model is key to understand whether the new circuit is able to generate patterns and the characteristics of the patterns formed. Finally, as experimental data appears, this more complex model will be necessary to fit the data and to obtain a predictive model of the synthetic circuit.

2.2 Global parameter space search

2.2.1 Types of patterns

Not all circuits are pattern generators, therefore understanding whether our circuit has this ability is key before carrying on with the experimental implementation of the circuit. Although it is suggested this new circuit is able to generate patterns as it is based on Scholes circuit 3954, it still has not been proved. Because the parameter values of our model are unknown, the whole 22 dimensional parameter space must be explored. The circuit will only be a pattern generator if a specific parameter set is able to generate patterns. To answer this question, the parameter space was sampled using Latin hypercube sampling (LHS), a technique shown to be efficient when sampling high dimensional spaces (Bergstra et al. 2012; Iman et al. 1980). In total, 1 million parameter sets were analysed using linear stability analysis. Although the main focus of the search is finding Turing patterns, a wide variety of other results were obtained from this sampling. The analysis workflow done on each parameter is shown in Figure 4.

Different steady states are found which are classified in 4 types depending on their dis-

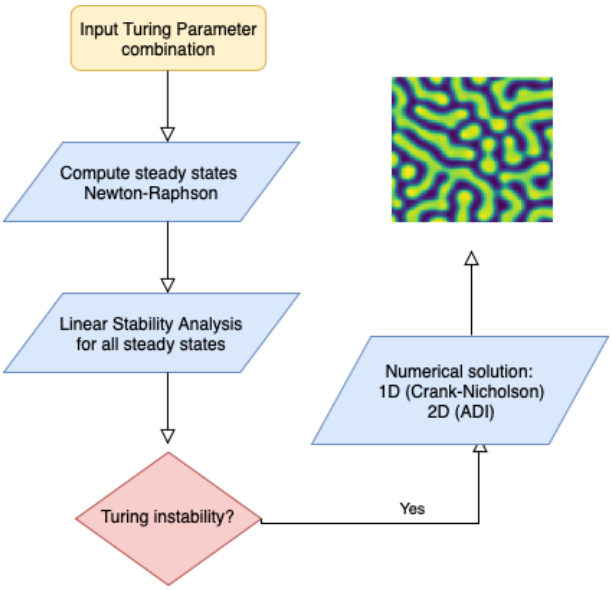
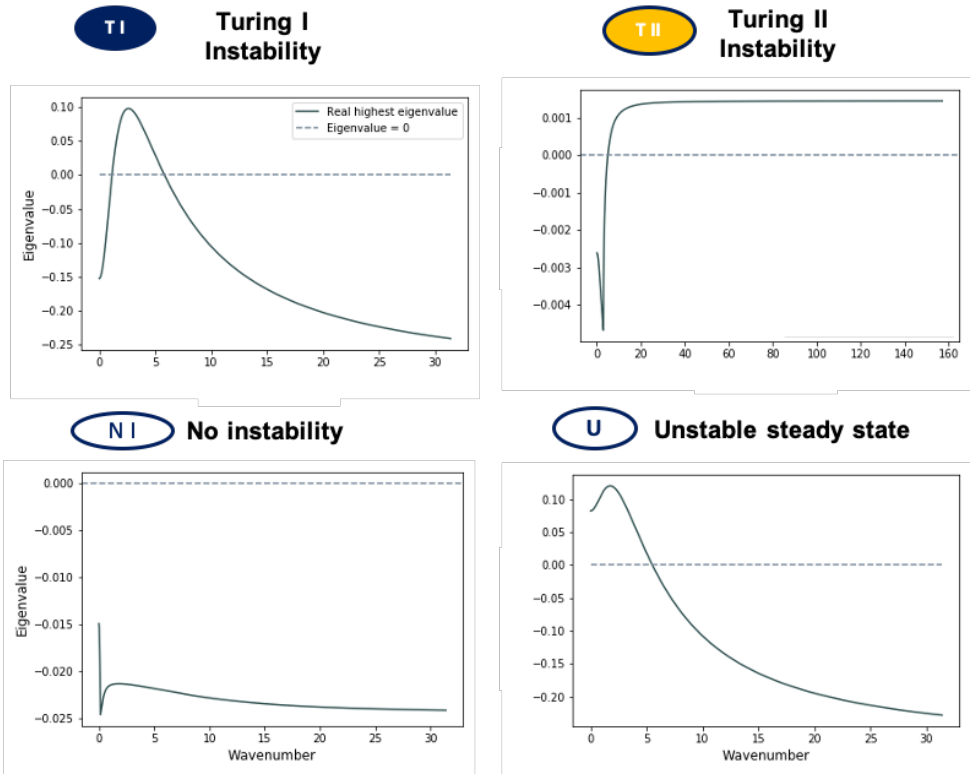


Figure 4: Turing analysis workflow. Analysis to determine whether a parameter set can form Turing patterns. It consists of finding steady states of the system and then analysing the stability of each steady state in the absence and presence of diffusion. Finally, if linear stability analysis suggests the parameter set is a pattern generator, this parameter set is further explored in detailed using a numerical solver.

persion relation. The dispersion relation of the 4 types of steady states can be seen in Figure 5. The steady state capable of Turing pattern formation is Turing I. As previously mentioned, the steady state is stable without diffusion (meaning a perturbation will decay), and unstable upon diffusion (perturbation is amplified exponentially).

Figure 5: Types of steady states found found in parameter search. Dispersion relation of the different types of steady states. The objects next to the name will be used to refer to each steady state in Figures 7,8 and 9.



The different types of steady states can give rise to different patterning regimes. The 3 different patterning regimes found in the sampling of 1 million parameter sets are shown in Figure 6. These pattern behaviours can be a result of either a monostable system or a multistable system.

The first type of pattern is the classical Turing pattern (Figure 7). This pattern is stationary and remains constant over time once it is formed. This pattern forms under different circumstances: when a monostable system has a Turing I steady state or when a multistable system has a combination of unstable and Turing I steady states. For this pattern to occur, the imaginary part of the highest eigenvalue in the dispersion relation must be zero. Figure 7a shows a Turing pattern evolves from a homogeneous starting point.

The second type of pattern is the oscillatory pattern (Figure 8). The pattern does not converge into a stationary form, but it has waves that oscillate and travel in space. This pattern is obtained when the highest eigenvalue of the dispersion has a positive imaginary part. This type of pattern can occur when either Turing I or Turing II steady states are present.

Finally the third type of pattern is the unstable Turing pattern (Figure 9). This type of pattern occurs only in multistable systems. The behaviour seems similar to a classical Turing pattern (Figure 7), except the pattern breaks after some time. This behaviour can be explained by the system starting in a Turing I steady state, and upon amplification of the perturbation, the system jumps into another stable steady state with no patterning abilities (no-instability steady state). This hypothesis is supported by the fact that the concentration values at pre-pattern and post-pattern correspond to steady states Turing I and No-instability steady states respectively, meaning there is a jump from one to the other. This type of patterning regime has not been been widely described in the literature, showing the lack of understanding on alternative types of patterns.

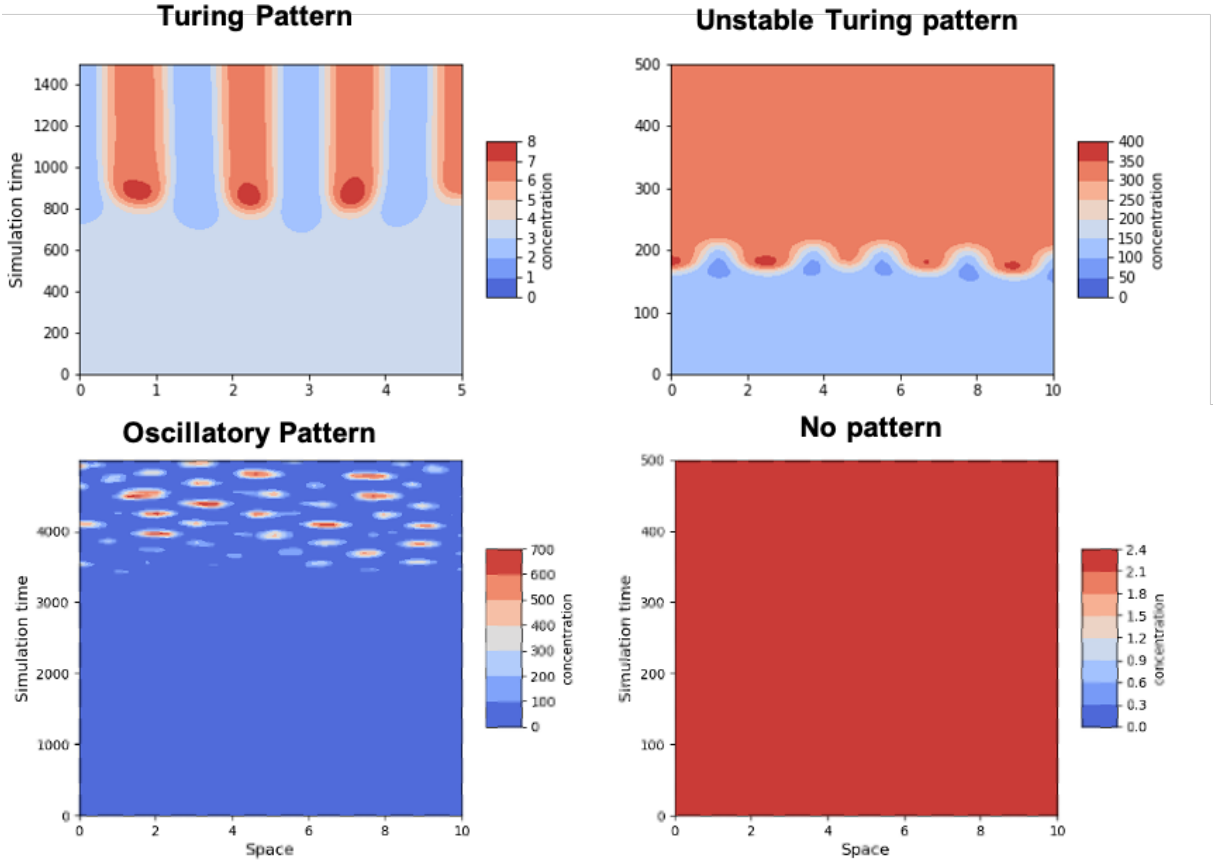


Figure 6: Pattern types found in parameter search. Evolution of a 1D pattern over time (y axis). Concentration of species is shown with a colour scale defined on the right of each graph.

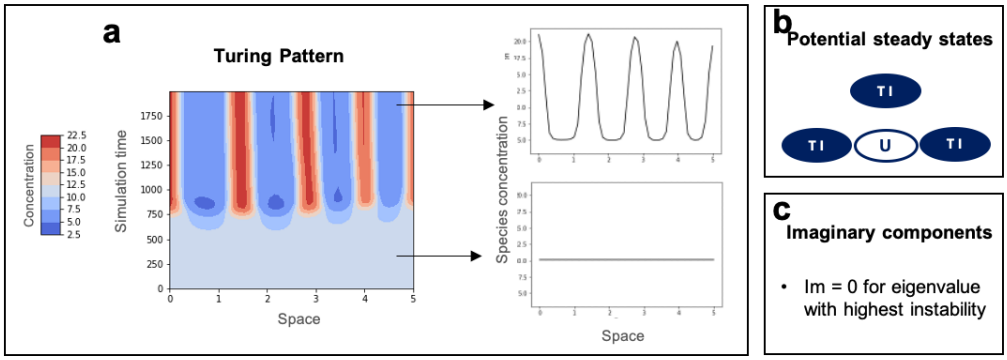


Figure 7: Characteristics of Turing patterns. **(A)** Evolution of pattern over time and concentration of species at single time points. **(B)** Combination of steady states capable of forming Turing patterns. Blue dot indicates Turing I steady state and crossed dot indicates unstable steady state. **(C)** Imaginary part of the dispersion relation for the steady states.

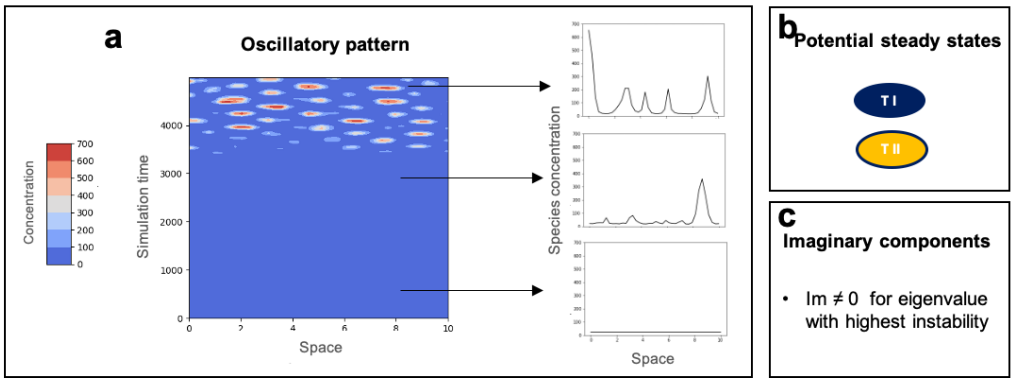


Figure 8: Characteristics of oscillatory patterns. **(A)** Evolution of pattern over time and concentration of species at single time points. **(B)** Combination of steady states capable of forming Turing patterns. Blue dot indicates Turing I steady state and crossed dot indicates unstable steady state. **(C)** Imaginary part of the dispersion relation for the steady states.

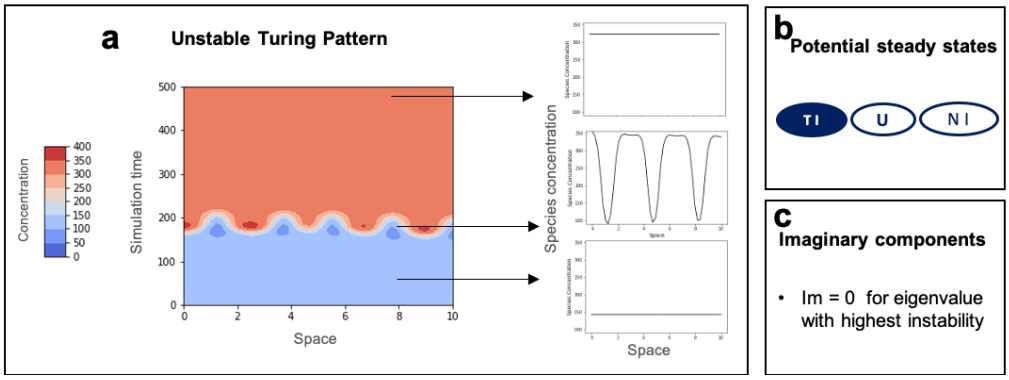


Figure 9: Characteristics of unstable Turing patterns. **(A)** Evolution of pattern over time and concentration of species at a single time point. **(B)** Combination of steady states capable of forming Turing patterns. Blue dot indicates Turing I steady state and crossed dot indicates unstable steady state. **(C)** Imaginary part of the dispersion relation for the steady states.

Out of the 1 million parameter sets analysed, 253 exhibited one of the 3 patterns mentioned above. Out of those, 169 showed an oscillatory pattern; 9 exhibited an unstable Turing pattern behaviour; and finally 3 parameter sets gave rise to Turing patterns.

2.2.2 Turing patterns

The three parameter sets and the respective patterns are shown below in Figure 10. These 3 parameter sets capable of Turing pattern formation, prove that the biologically implemented circuit 3954 with 6 species is a Turing pattern generator under certain conditions.

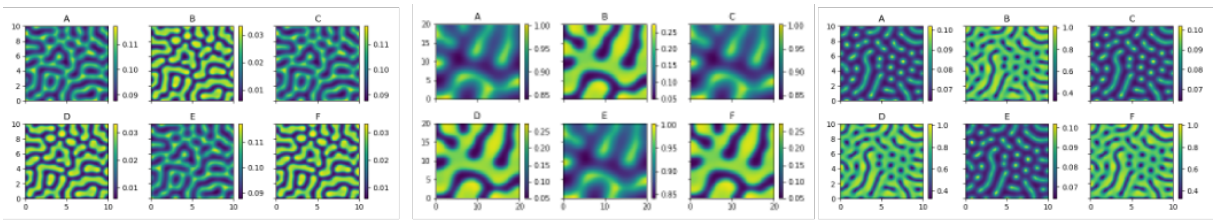


Figure 10: Turing patterns found in parameter space search. 6 patterns are shown for each parameter set, representing the numerical result for each specie modelled. Note parameter set 694300 is shown in a different space scale (20x20) due to its higher wavelength.

The wavelength of these 3 patterns can be obtained using either linear stability analysis or the numerical solution. In linear stability analysis, the wavenumber corresponding to the highest peak of the dispersion relation is theoretically the one attributed to the pattern. The wavelength can be obtained from the dispersion relation using $wavelength = \frac{2\pi}{wavenumber}$. From the numerical simulation, the peaks of the solution can be identified and the average distance between peaks is calculated giving the wavelength. The wavelengths calculated from these two methods are shown in Table 3.

Table 3: Turing pattern wavelengths estimated from the linear stability analysis and the numerical simulation.

Parameter set	LSA wavelength (mm)	Simulation wavelength (mm)
221368	1.38	1.40
694300	6.89	5.46
835723	1.43	1.25

In the three parameter sets, the two predictions are similar. This further suggests the linear

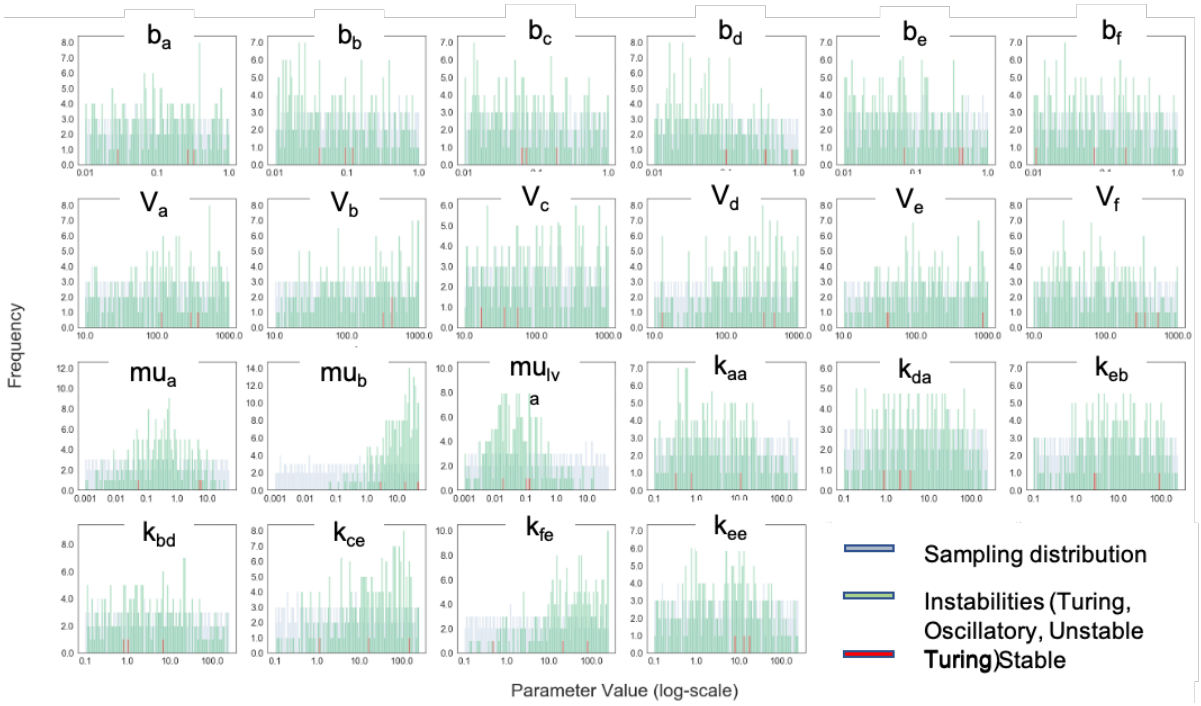


Figure 11: Distribution of values for every parameter of the model. Distribution of parameter values for initial sampling distribution (blue), instabilities (green) and Turing patterns (red). Instabilities represent any parameter sets containing Turing I or Turing II steady states. The graph represents the frequency of each parameter value. The parameter value is represented in log scale.

stability analysis has a relationship with the numerical solution, even though the model is complex and has non-linearities. Overall, all wavelengths are of the same magnitude. This might suggest what type of scale we should be looking at when imaging the pattern experimentally.

2.2.3 Parameter distributions

Although the remaining parameter sets that form oscillatory patterns and unstable Turing patterns are not as relevant for experimental applications, they still contain useful information. The distribution of parameter values for those patterning parameter sets can indicate how to increase the chance of obtaining a pattern (oscillatory, unstable Turing or Turing). By assuming oscillatory and unstable Turing patterns are close to Turing patterns in parameter space, we can obtain an indication on how to tune parameters to increase the chance of obtaining Turing patterns experimentally using circuit 3954.

As shown in Figure 11, the distribution of parameter values from the original latin hypercube sampling is different than that of the distribution of values capable of patterning (Turing pattern, unstable Turing pattern or oscillatory pattern). A clear example is *mub*, which corre-

sponds to the degradation rate of HSL. For mub, the distribution of parameter values showing patterning is biased towards higher values compared to the original sampling distribution. This could mean that sampling from a higher range for mub, could increase the chance of finding patterning systems and therefore Turing patterns. Experimentally, this could mean increasing the degradation rate of HSL might increase our chances of finding a Turing pattern using circuit 3954. This bias is also seen in other parameters such as $Vd, mua, mulva, kce$ and kfe.

2.3 Robustness study

As previously mentioned in the Introduction, one of the main challenges of engineering synthetic RD systems is the limited fraction of Turing space (parameter sets able to generate Turing patterns) compared to the whole parameter space (Scholes and Isalan 2017). In other words, the circuit might be able to generate patterns but the conditions required are extremely limited. Therefore, having a model is key to understand what the parameter space looks like, what areas of the parameter space are more robust to noise and therefore maximise the chances of hitting Turing space experimentally. Robustness in this case is defined as the amount of perturbation a Turing parameter set can withstand before it stops being a Turing parameter set. So if you find yourself in a region of parameter space which is quite robust for Turing pattern formation, no biological noise will affect this outcome.

The three Turing parameter sets were studied to understand how robust to noise they are using parameter sensitivity analysis. This method involves studying the robustness of each parameter in the model independently, to understand the stiffness or sloppiness of that parameter (or how constrained it is). In a Turing parameter set, the 22 parameters are tested through sensitivity analysis. This means that each parameter value is increased and decreased gradually until the variation makes it a non-Turing parameter set. Therefore each parameter of each parameter set has a higher and an upper bound. The range between the higher and upper bound are the values where that parameter allows for Turing pattern formation and that range can be considered as a robustness measure for each parameter. Because each parameter is of a different size, the upper and lower bounds are standardised according to the original parameter value. This way robustness can be compared across parameters and across parameter sets. Boundaries were calculated for each of the 3 parameter sets and are shown in Figure 12.

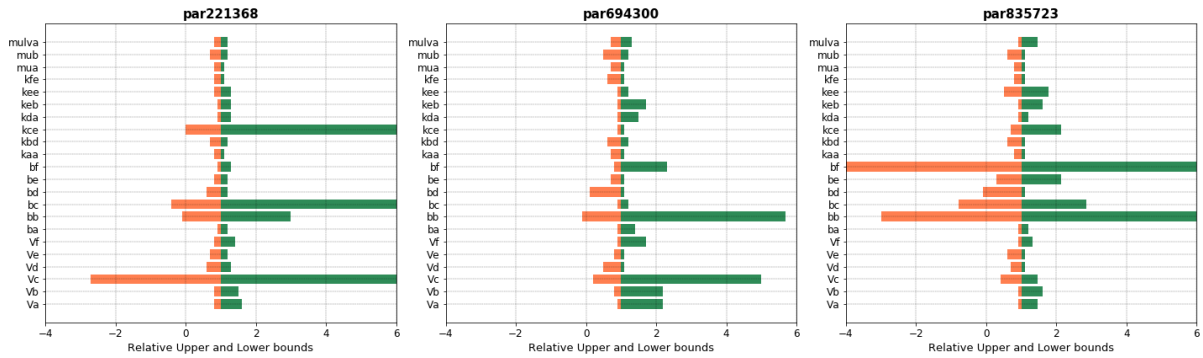


Figure 12: Sensitivity analysis for each parameter of the model independently. Lower bounds of each parameter are shown in orange and upper bounds are shown in green. The data is standardised so the centre of the graph is fixed at 1. A parameter with a bound close to 1 has almost no range of variability for Turing behaviour, and therefore has limited robustness.

Robustness in different parameters differs across the 3 parameter sets. However there are some common trends as seen in Figure 12. Overall basal production rates appear to be more robust than other parameters, specially the basal production rate of HSL (bb) which is robust in the three parameter sets. Other parameters also show robustness such as the induced maximum production rate for TetR production (Vc) or the Michaelis-Menten constant for activation of cI* by TetR (kce). However, this robustness study should be analysed sceptically as only three parameter sets are tested.

One of the aims that would facilitate biological implementation of this circuit to obtain Turing patterns would be to find a region of parameter space with maximum robustness. However robustness in itself is hard to maximise for several reasons. First, robustness in itself cannot be accurately expressed with a single function that can be maximised. And secondly, even if an average value of robustness from the sensitivity analysis was taken as a function, it would extremely computationally expensive to carry a sensitivity analysis at every iteration of the optimisation process. A potential alternative to maximise robustness is to find a link between robustness and the dispersion relation. A hypothesis explored in this thesis is whether a higher peak in the dispersion relation affects robustness in a positive way. Because a negative peak would not lead to any Turing pattern and a positive peak would, it is fair to hypothesise that a higher peak would give rise to a more robust Turing pattern with respect to noise (as a high peak is further away from negative values).

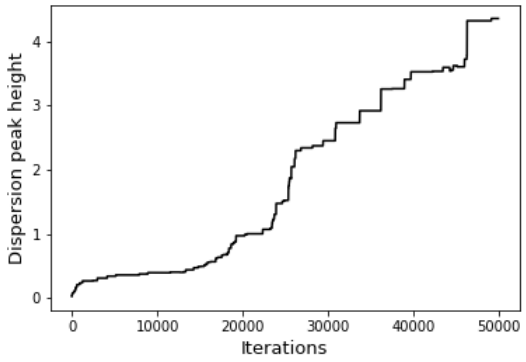


Figure 13: Optimisation of dispersion peak height. Dispersion peak height is optimised for 50000 iterations. The value of dispersion peak height at every iteration is represented, showing the efficiency of the optimisation.

To test this hypothesis, a method to optimise the dispersion peak was introduced. A Turing parameter set is used as a starting point and the parameter space around it is explored to find parameter sets with higher dispersion peak heights. The output of this optimisation is a group of n Turing parameter sets (n being the number of iterations), all with different dispersion peak heights. This group of n parameter sets will be used to compare dispersion peak height and robustness of each. The optimisation is carried out using a Markov-Chain Monte Carlo combined with the Metropolis algorithm. The function to optimise is the height of the peak in the dispersion relation (e.g the highest value of the highest eigenvalue). Although the analysis was carried out on the three parameter sets, only the results for parameter set 221368 are shown to avoid repetition, as similar results were obtained for the other two parameter sets.

The dispersion peak was optimised from 0.22 to 4.36 after 50000 iterations of the algorithm as seen in Figure 13. Although no significant improvement occurs from iterations 100 to 15000, a steady increase in dispersion is made after iteration 15000. This shows how the algorithm can avoid getting stuck in local maxima to go towards a global maxima. There is no indication the algorithm is close to reaching a global maxima as the rate of improvement is still high after many iterations.

From that optimisation path, 64 parameter sets were selected, each with a different dispersion peak height. The height of the dispersion peak and the standard robustness for each parameter are compared in Figure 14 to understand the correlation. The correlation value is a Pearson's correlation calculated using *np.corrcoef*.

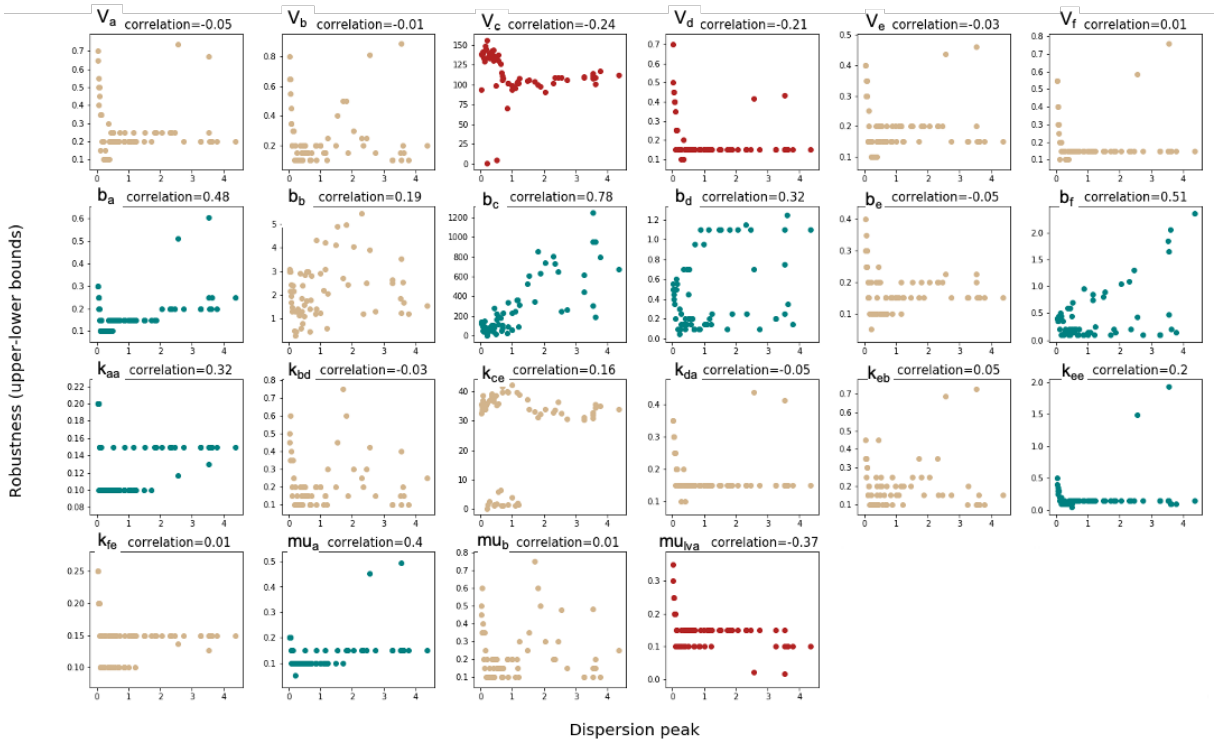


Figure 14: Effect of dispersion peak height on robustness for each parameter independently. Each graph shows the relationship between the robustness of that parameter in sensitivity analysis and the dispersion peak height. Each dot of the graph represents a single parameter set chosen from the optimisation path where both robustness and dispersion peak height have been analysed. Blue scatter plots represent a positive linear correlation, yellow represent insignificant correlation and red represent negative linear correlation.

Every graph in Figure 14 corresponds to one parameter, and every dot in the graph to one of the 64 parameter sets obtained from the optimisation path. Although some parameters show a positive correlation like bc, bd and bf, there is no clear trend of positive correlation between dispersion peak height and robustness across all parameters in the model. Therefore no clear conclusion can be drawn from this, meaning robustness cannot be maximised by optimising the height of the dispersion peak.

2.4 Relationship between pattern characteristics and dispersion peak height

Although a higher dispersion peak is not positively correlated to an increase in robustness, other interesting and useful conclusions can be obtained from this optimisation study. The pattern characteristics (e.g. time for pattern convergence and shape of pattern) will be studied for parameter sets with different dispersion peak heights. 64 parameter sets from the optimisation path will be solved numerically to observe the time for pattern convergence.

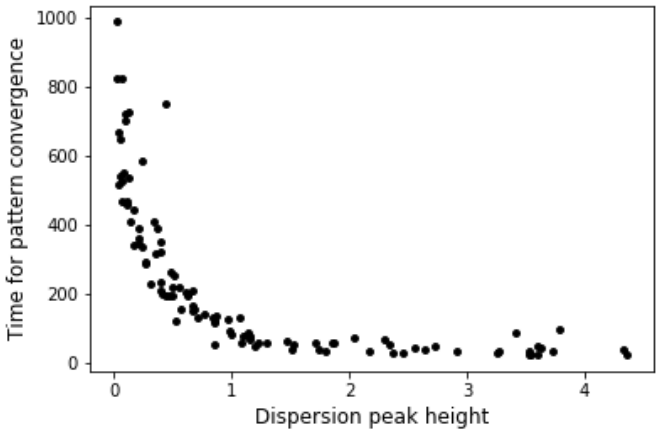


Figure 15: Effect of dispersion peak height on convergence time. Y axis represents the time it takes for a parameter set to go from homogeneous state to a converged pattern using a numerical simulation.

In Figure 15, the correlation between time for pattern convergence and dispersion peak height is shown. Only the behaviour of parameter set 221368 is shown here. The relationship can be fitted to $y = \frac{1}{x}$, meaning there is a negative correlation between time it takes for pattern convergence and dispersion peak height. This means parameter sets with a higher dispersion peak will result in a higher speed of pattern formation.

The numerical solutions of this 64 parameter sets were also observed. The first and last parameter sets of every optimisation path can be seen in Figure 16. The first has a smaller dispersion peak height while the last has an optimised and higher dispersion peak height. As dispersion peak height is increased, the labyrinths transition into spots. Parameter sets with smaller dispersion peak heights produce labyrinth shaped patterns, while parameter sets with an optimised dispersion peak height give rise to dotted patterns.

This information is useful if the speed of pattern formation or shape of pattern is to be controlled experimentally. Ideally, some parameters could be increased or decreased to control this behaviour. In Figure 17, the relationship between height of dispersion peak and parameter values is shown for every parameter of the model. This relationship is plotted for the 64 parameter sets from the optimisation path. For some parameters we can see a clear positive (blue) or negative (red) correlation between dispersion peak height and parameter value. For example, *mulva* has a clear positive correlation ($y = 0.99x$) while *mub* has a negative correlation ($y = -0.94x$) This correlations could be used to manipulate the dispersion peak height and therefore the speed of pattern formation or shape of pattern.

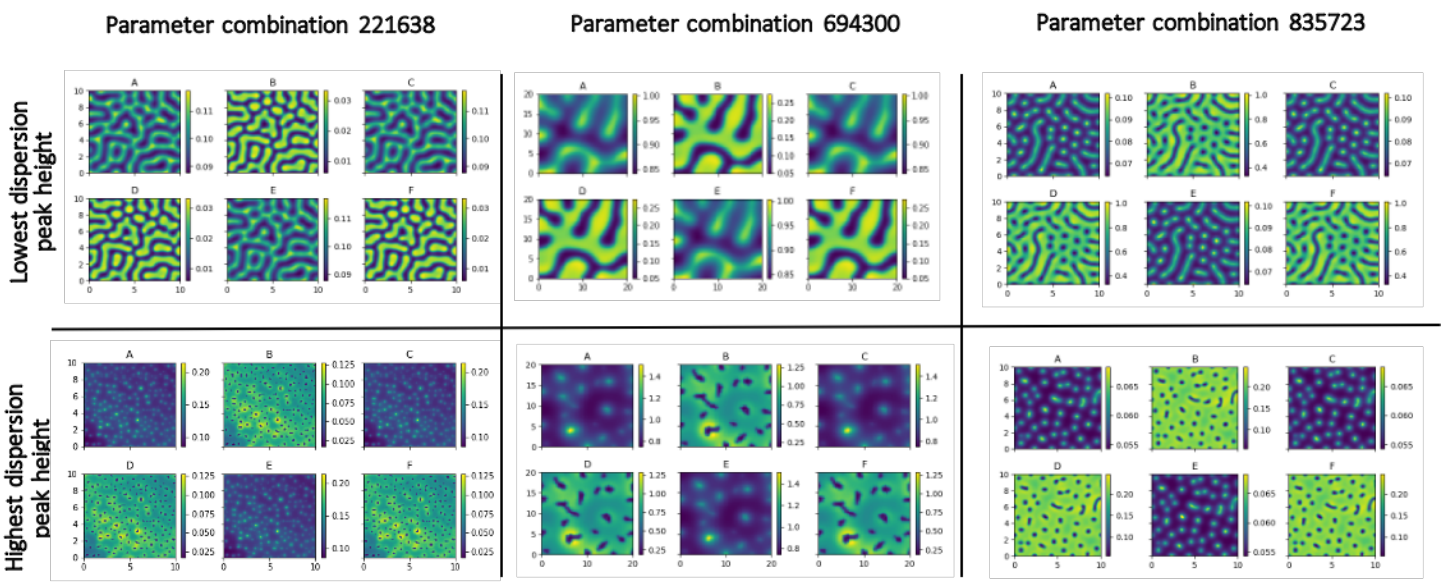


Figure 16: Effect of dispersion peak height on pattern shape. Top figures are non-optimised parameter sets (lowest dispersion peak height). Bottom figures represent parameter sets originating from the top parameter sets, that have been optimised for dispersion peak height (highest dispersion peak height). The 6 panels of every parameter set represent the 6 molecular species of the model.

Although robustness is not linked to dispersion peak height, speed of pattern formation and pattern shape are. To increase the speed of pattern formation and to obtain a dotted like pattern, dispersion peak height must be increased. This can be done by tuning certain parameters experimentally. As far as we know, this relationship between dispersion peak height and speed of pattern formation or shape has not been previously described in the literature. This novel discovery could potentially help in the understanding and control of Turing patterns.

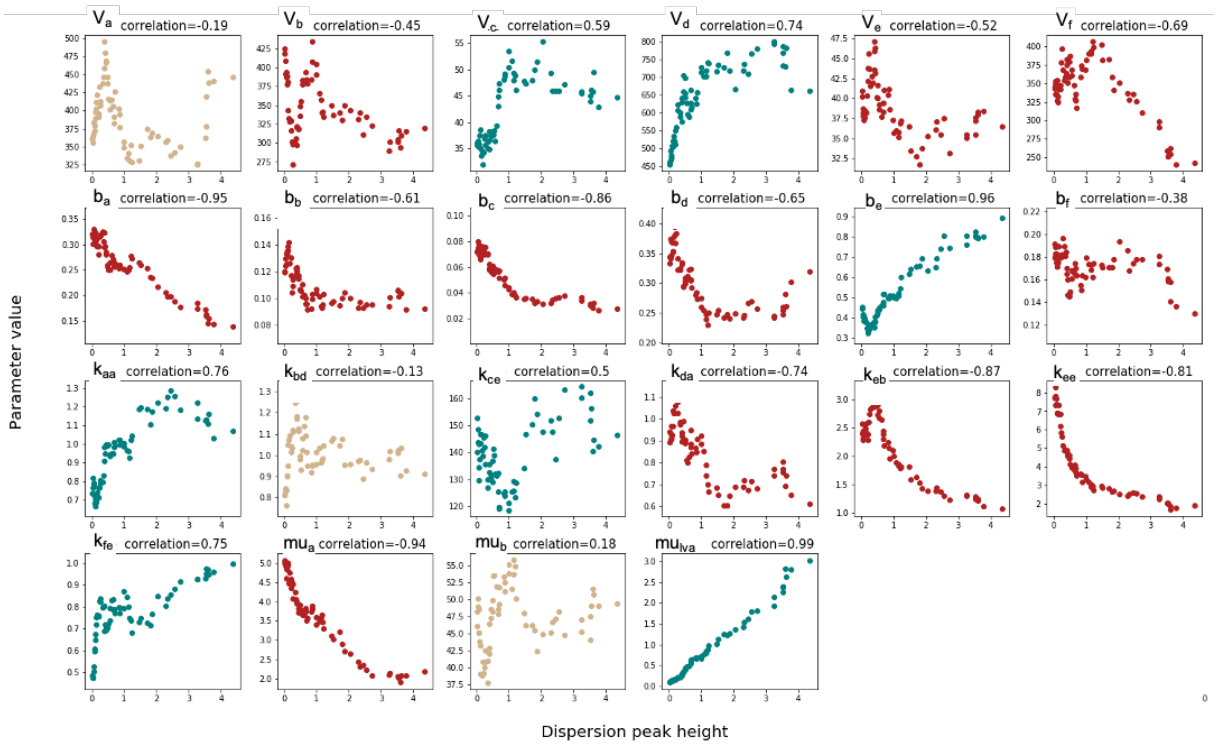


Figure 17: Parameter values as a function of dispersion peak height. Each graph shows the relationship between the parameter value and the dispersion peak height. Each dot of the graph represents a parameter set from the optimisation path containing that parameter value. Blue scatter plots represent a positive linear correlation, yellow represent insignificant correlation and red represent negative linear correlation.

3 Methods

The parameter space of the model defined is explored to find Turing patterns. 1 million parameter sets are analysed to understand if any can generate a Turing pattern. The workflow for Turing analysis of a parameter set is shown in Figure 4 in the Results section. The first step is to find the steady states or equilibrium states of a system. Secondly, linear stability analysis is carried out on each steady state. Depending on the number of steady states and their nature (different types shown in figure 5), the system will potentially give rise to a Turing pattern. Figure 7b shows which steady states are required for a Turing pattern formation. If linear stability analysis shows a Turing pattern can be formed with that parameter set, a numerical simulation is runned. This numerical simulation is computed both in 1D with the Crank-Nicolson (CN) method and in 2D with the Alternating Direction Implicit method (ADI). The numerical simulation provides the solution (i.e. an image of the pattern) and shows different characteristics of the pattern such as speed of pattern formation, wavelength, shape and species concentrations. Every step is explained in detail in the following sections.

3.1 Modelling framework

Each species is modelled using a partial differential equation with 4 different terms. The basal production term is a constant (b_A). The production term subject to regulation is a result of multiplying a maximum production rate (V_A) and Hill-Langmuir sigmoidal terms. Hill-Langmuir equation suggests the fraction of receptors occupied by a ligand (Gesztelyi et al. 2012). Being the ligand an activator, if fraction is 1, the production term subject to regulation is the maximum production rate. For the inhibition, the inverse of the activation term is taken. The activation and inhibition terms are shown in Equations 4a and 4b respectively where A is the ligand.

$$Act = \frac{1}{1 + \left(\frac{K_A}{A}\right)^n} \tag{4a}$$

$$Inh = \left(\frac{1}{1 + \left(\frac{K_A}{A}\right)^n}\right)^{-1} = \frac{1}{1 + \left(\frac{A}{K_A}\right)^n} \tag{4b}$$

Because the regulation acts at an operator level, it's non-competitive, meaning each ligand has its own operator and does not compete with other regulators to bind it. Therefore, to describe the production upon activation and inhibition, the Hill-Langmuir terms can be multiplied as independent. The third term of the equation is a linear function representing the degradation rate $\mu_A \cdot A$. Finally, the diffusion term is represented as a multiplication of a diffusion constant and the heat equation $D_A \cdot \frac{\partial^2 A}{\partial x^2}$ (Turing 1952).

Altogether, this leads to the partial differential equation describing the change in concentration of species over time. For example, for species A, the following equation is obtained:

$$\frac{\partial [A]}{\partial t} = b_A + V_A \cdot \frac{1}{1 + \left(\frac{[k_{aa}]}{A}\right)^{n_{aa}}} \cdot \frac{1}{1 + \left(\frac{[D]}{K_{da}}\right)^{n_{da}}} - \mu_A \cdot [A] + D_A \cdot \frac{\partial^2 [A]}{\partial x^2} \tag{5}$$

3.2 Finding the steady states: Newton-Raphson method

The following section describes the Newton-Raphson method: a root-finding method to solve systems of non-linear differential equations. This method will be used to obtain the equilibrium points of our system without the diffusion terms. A workflow diagram in Figure 18 shows the Newton-Raphson method by steps. The detailed method can be found in Conejo and Baringo 2018

3.2.1 System of equations

The roots of a system of n nonlinear equations with n variables is found through the process described below. The system of equations can be expressed as $\mathbf{f(x) = 0}$, using the following vectors to define the system:

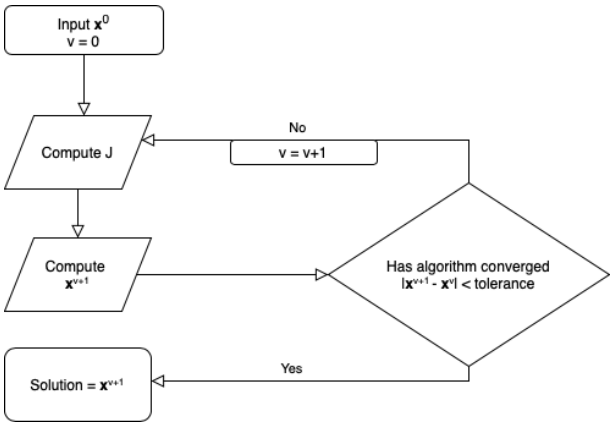


Figure 18: Newton-Raphson workflow to find steady states with an initial condition input ($\mathbf{x}^0$).

$$\mathbf{f(x)} = [f_1(x), f_2(x), ..., f_n(x)] \tag{6a}$$

$$\mathbf{x} = [x_1, x_2, ..., x_n] \tag{6b}$$

$$\mathbf{0} = [0, 0, ..., 0] \tag{6c}$$

The system can also be written in more detail as:

$$\begin{cases} f_1(x_1, x_2, ..., f_n) = 0 \\ f_2(x_1, x_2, ..., f_n) = 0 \\ \qquad \vdots \\ f_n(x_1, x_2, ..., f_n) = 0 \end{cases} \tag{7}$$

A value $\mathbf{x}^0$ will be defined which is the starting condition. A value $\Delta\mathbf{x}^0$ needs to be found for which $\mathbf{f(x)}^1 = \mathbf{f}(\mathbf{x}^0 + \Delta\mathbf{x}^0) = \mathbf{0}$. The system is expressed as a Taylor series. Higher order terms will be ignored and the differential term expressed as the Jacobian:

$$\mathbf{f}(\mathbf{x}^0 + \Delta\mathbf{x}^0) \approx \mathbf{f}(\mathbf{x}^0) + \frac{\mathbf{df}(\mathbf{x}^0)}{\mathbf{dx}^0}\Delta\mathbf{x}^0 \approx \mathbf{f}(\mathbf{x}^0) + \mathbf{J}^0\Delta\mathbf{x}^0 \tag{8}$$

$\Delta\mathbf{x}^0$ can be found using:

$$\Delta\mathbf{x}^0 \approx -[\mathbf{J}^0]^{-1}\mathbf{f}(\mathbf{x}^0) \tag{9}$$

At every iteration, being v the iteration counter, $\mathbf{x}$ can be updated as:

$$\mathbf{x}^{v+1} = \mathbf{x}^v - [\mathbf{J}^0]^{-1}\mathbf{f}(\mathbf{x}^0) \tag{10}$$

The convergence of the algorithm will be checked at every iteration. If $|\mathbf{x}^{v+1} - \mathbf{x}^{v}| < tolerance$, the algorithm has convergence and $\mathbf{x}^{v+1}$ can be considered as a root of the system. Else, the algorithm hasn't converged to the root of the equation and the process is repeated.

3.2.2 Newton-Raphson implementation

To obtain the roots of the system, the Newton-Raphson algorithm was implemented in python using a tolerance value of 0.0001 and a maximum of 15 iterations after which the algorithm stopped searching for a root. Because the system had the potential for multi-stability, several initial conditions need to be searched to obtain all steady states. In total 500 initial conditions were analysed by the Newton-Raphson to obtain all the roots of the system. The 500 conditions were chosen performing latin-hypercube sampling of a 6 dimensional space (6 species) from a uniform distribution with a range from 10^{-3} and 10^{3} . Approximately, the root finding for a single parameter set takes around 2 seconds.

3.3 Linear stability analysis

The following section describes how linear stability analysis is carried out for the equilibrium points in a reaction-diffusion system. The aim of this analysis is to find out if the steady state exhibits a Turing instability or also called a diffusion-driven instability. When it does, the system is capable of forming spatial patterns. As the name describes, diffusion-driven instabilities arise in these systems when a homogeneous steady state is stable to small perturbations in the absence of diffusion, and becomes unstable in the presence of diffusion (Glendinning 1994; J. D Murray 2002). To check for Turing instabilities, the stability of the equilibrium state, will be studied with and without diffusion. The method of linear stability analysis will be explained for the two morphogen reaction-diffusion system, shown below:

$$\frac{\partial A}{\partial t} = f_A(A, B) + D_A \frac{\partial^2 A}{\partial x^2} \tag{11a}$$

$$\frac{\partial B}{\partial t} = f_B(A, B) + D_B \frac{\partial^2 B}{\partial x^2} \tag{11b}$$

where $f_{A,B}$ are the non-linear production terms and $D_{A,B}$ are the diffusion constants of the two morphogens. For future reference, X is a generalisation term to refer to any morphogen A or B.

3.3.1 Stability of steady state without diffusion

To study the stability around the steady state without diffusion, Equations 15a and 15b are used, except diffusion terms are removed ($D_{A,B} = 0$):

$$\frac{\partial A}{\partial t} = f_A(A, B) \tag{12a}$$

$$\frac{\partial B}{\partial t} = f_B(A, B) \tag{12b}$$

The steady states are defined as A^* and B^*, which satisfy the condition:

$$f_A(A^*, B^*) = 0, \qquad f_B(A^*, B^*) = 0 \tag{13}$$

Linear stability analysis is carried by adding an infinitesimally small perturbation δX to the steady state X^*, and studying if the perturbation decays (stable steady state) or grows (unstable steady state) over time. The perturbation needs to be almost insignificant as Taylor expansion is carried out to linearise the system around the steady state. Therefore, the morphogen concentration can be expressed as:

$$A(t) = A^* + \delta A(t), \qquad |\delta A| << A^* \tag{14a}$$

$$B(t) = B^* + \delta B(t), \qquad |\delta B| << B^* \tag{14b}$$

The differential equations 16a and 16b are re-evaluated, using equations 17 and 18a,b:

$$\frac{\partial A}{\partial t} = \frac{\partial [A^* + \delta A(t)]}{\partial t} = f_A(A^* + \delta A(t), B^* + \delta B(t)) = \frac{\partial \delta A}{\partial t} \tag{15a}$$

$$\frac{\partial B}{\partial t} = \frac{\partial [B^* + \delta B(t)]}{\partial t} = f_B(A^* + \delta A(t), B^* + \delta B(t)) = \frac{\partial \delta B}{\partial t} \tag{15b}$$

As previously mentioned, the non-linear system will be linearised around the steady state using Taylor expansion. This is done to have a simpler set of equations, that represent the system around the steady state, as seen below:

$$f(A+\delta A, B+\delta B) = f(A, B) + \frac{\partial f(A, B)}{\partial A}\delta A + \frac{\partial f(A, B)}{\partial B}\delta B + ... + \frac{1}{n!}\frac{\partial^n f(A, B)}{\partial A^n}\delta A^n + \frac{1}{n!}\frac{\partial^n f(A, B)}{\partial B^n}\delta B^n \tag{16}$$

If δA and δB are small enough, higher order terms can be ignored as $\delta(A, B)^n$ becomes infinitesimally small. Furthermore, it can be assumed that $f(A^*, B^*) = 0$, therefore the following

expression is obtained, where f corresponds to either f_A or f_B :

$$f(A^* + \delta A, B^* + \delta B) = \frac{\partial f(A,B)}{\partial A}\delta A + \frac{\partial f(A,B)}{\partial B}\delta B \tag{17}$$

Finally, because $\frac{\partial X}{\partial t} = \frac{\partial \delta X}{\partial t}$ (as seen in Equations 19a and 19b), the change in perturbation (decay or growth) can be expressed as :

$$\frac{\partial \delta A}{\partial t} = \frac{\partial f_A(A,B)}{\partial A}\delta A + \frac{\partial f_A(A,B)}{\partial B}\delta B \tag{18a}$$

$$\frac{\partial \delta B}{\partial t} = \frac{\partial f_B(A,B)}{\partial A}\delta A + \frac{\partial f_B(A,B)}{\partial B}\delta B \tag{18b}$$

The general solution can be expressed as an exponential, where σ determines the growth rate of the perturbation, and whether it grows (unstable steady state) or decays (stable steady state):

$$\delta A = A_0 e^{\sigma t} \tag{19a}$$

$$\delta B = B_0 e^{\sigma t} \tag{19b}$$

If Equations 23a and 23b are introduced into 22a and 22b, and the solution divided by $e^{\sigma t}$ on both sides, the following is obtained:

$$\sigma \delta A_0 = \frac{\partial f_A(A,B)}{\partial A}\delta A_0 + \frac{\partial f_A(A,B)}{\partial B}\delta B_0 \tag{20a}$$

$$\sigma \delta B_0 = \frac{\partial f_B(A,B)}{\partial A}\delta A_0 + \frac{\partial f_B(A,B)}{\partial B}\delta B_0 \tag{20b}$$

This can be repressented as an eigenvalue-eigenvector problem using the jacobian $\mathbf{J}$, the eigenvalue σ and the eigenvector $\delta u_X = [u_A, u_B]$:

$$\sigma \delta X_0 = \begin{bmatrix} \frac{\partial f_A}{\partial A} & \frac{\partial f_A}{\partial B} \\ \frac{\partial f_B}{\partial A} & \frac{\partial f_B}{\partial B} \end{bmatrix} \delta X_0 \tag{21}$$

Finally, σ can be obtained by solving:

$$p(\sigma) = det[J - \sigma I] = 0 \tag{22}$$

The real part of σ corresponds to the growth rate of the perturbation (Equations 23a and 23b). $Im(\sigma)$ corresponds to the oscillations, which can be ignored for now. If $Re(\sigma) < 0$, the steady state will be stable to perturbations, which means the system will go back to steady state after a small perturbation is applied. If $Re(\sigma) > 0$, the steady state will be unstable to perturbations, and the system will go away from the equilibrium point after being slightly perturbed.

3.3.2 Stability of steady state with diffusion

Once the stability of the steady state has been analysed in the absence of diffusion, the effect of diffusion on the same system will be studied. The diffusion will be introduced through the partial derivative term $D_X \frac{\partial^2[X]}{\partial x^2}$, where X is the morphogen, giving rise to Equations 15a and 15b.

3.3.2.1 Reduction of PDE term to ODE through Fourier transformations The second order partial derivative term will be reduced to an ODE simpler term through Fourier transforms. For this purpose, the following Fourier transform definitions are used:

$$F(k) = \mathcal{F}(f(x)) = \int_{-\infty}^{\infty} f(x) e^{-ikx} dx \tag{23a}$$

$$f(x) = \mathcal{F}^{-1}(F(k)) = \frac{1}{2\pi} \int_{-\infty}^{\infty} F(k) e^{ikx} dk \tag{23b}$$

First, using the inverse Fourier transform (Equation 27b), the first order partial derivative will be reduced to an ODE form:

$$\frac{\partial[X(x,t)]}{\partial x}) = \mathcal{F}^{-1}\left(\frac{\partial[X(k,t)]}{\partial x}\right) = \frac{\partial}{\partial x}\left[\frac{1}{2\pi}\int_{-\infty}^{\infty} X(k,t) e^{ikx} dk\right] = \frac{1}{2\pi}\int_{-\infty}^{\infty} X(k,t)\frac{d}{dx} e^{ikx} dk$$

$$= \frac{1}{2\pi}\int_{-\infty}^{\infty} ikX(k,t) e^{ikx} dk = ik\frac{1}{2\pi}\int_{-\infty}^{\infty} X(k,t) e^{ikx} dk = ikX(x,t) \tag{24}$$

giving $\frac{\partial[X(x,t)]}{\partial x}) = ikX(x,t)$. Using this expression, the inverse Fourier transform (Equation 22b) and the fact that $\frac{\partial^2 X(x,t)}{\partial x^2} = \left(\frac{\partial X(x,t)}{\partial x}\right)^2$, the following expression is obtained:

$$\frac{\partial^2[X(x,t)]}{\partial x^2}) = \left(\frac{\partial X(x,t)}{\partial x}\right)^2 = \mathcal{F}^{-1}\left[\frac{\partial[ikX(k,t)]}{\partial x}\right] = \frac{\partial}{\partial x}\left[\frac{1}{2\pi}\int_{-\infty}^{\infty} ikX(k,t)e^{ikx}dk\right]$$

$$= \frac{1}{2\pi}\int_{-\infty}^{\infty} ikX(k,t)\frac{d}{dx}e^{ikx}dk = \frac{1}{2\pi}\int_{-\infty}^{\infty}(ik)^2X(k,t)e^{ikx}dk = -k^2\frac{1}{2\pi}\int_{-\infty}^{\infty}X(k,t)e^{ikx}dk = -k^2X(x,t) \tag{25}$$

giving

$$\frac{\partial^2 A(x,t)}{\partial x^2} = -k^2 \cdot A \tag{26a}$$

$$\frac{\partial^2 B(x,t)}{\partial x^2} = -k^2 \cdot B \tag{26b}$$

This way, the second order partial derivative terms, are reduced to simpler ODE systems to be used in linear stability analysis

3.3.2.2 Definition of boundary conditions The system is defined within some boundaries, meaning x goes from 0 to L. In this case, no molecules can leave the system, so the boundary condition applied is a zero-flux boundary condition. At the boundaries, the partial derivative is zero, defined below as Neumann Boundary conditions:

$$\frac{\partial^2 X(x=0,t)}{\partial x^2} = 0 \qquad \& \qquad \frac{\partial^2 X(x=L,t)}{\partial x^2} = 0 \tag{27}$$

This can be visualised as a fish tank, of length L, filled with water: water diffuses freely within the tank, but cannot diffuse outside the boundaries. In addition, the levels of water at the boundaries are not fixed. For the spatial derivatives to be zero at the boundaries, some restrictions will need to be applied. The expression from Equation 29 will be used to apply our boundary conditions.

$$\frac{\partial^2[X(x,t)]}{\partial x^2}) = -k^2\frac{1}{2\pi}\int_0^L X(k,t)e^{ikx}dk = -k^2\frac{1}{2\pi}\int_0^L X(k,t)(cos(kx)+isin(kx))dk \tag{28}$$

$$= \frac{-k^2}{2\pi}\left(\int_0^L X(k,t)cos(kx)dk + i\int_0^L X(k,t)sin(kx)dk\right)$$

The first integral is integrated by parts to express the equation in terms of $sin(kx)$:

$$\frac{\partial^2[X(x,t)]}{\partial x^2}) = \frac{-k^2}{2\pi}\left(\left[\frac{sin(kx)}{k}X(k,t)\right]_0^L - \int_0^L \frac{sin(kx)}{k}\frac{\partial A(k,t)}{\partial k}dk + i\int_0^L X(k,t)sin(kx)dk\right) \tag{29}$$

As defined with the boundary conditions (Equation 31), the partial derivative must be 0 at $x = 0$ and $x = L$. Therefore, at $x = 0$ and $x = L$, $sin(kx) = 0$. For the two boundary conditions:

- When $x = 0$: $sin(k \cdot 0) = sin(0) = 0$. Therefore for $x = 0$, $\frac{\partial^2 X(0,t)}{\partial x^2} = 0$.
- When $x = L$: $sin(k \cdot L)$ must be zero. If $k = \frac{n\pi}{L}$, $sin(kL) = sin(n\pi) = 0$. Therefore, for $x = L$, $\frac{\partial^2 X(L,t)}{\partial x^2} = 0 \quad \forall n \in \{0, \mathbb{N}\}$

Following these calculations, the zero-flux boundary condition is applied when $k = \frac{n\pi}{L}$ $\forall n \in \{0, \mathbb{N}\}$

3.3.2.3 Linear Stability Analysis of steady state with diffusion effects

Now that the PDE diffusion term has been reduced to an ODE term, and the zero-flux boundary conditions introduced, the stability of the steady state when introducing diffusion can be studied. As shown in Equations 30a and 30b, the diffusion term $D_X\frac{\partial^2 X}{\partial x^2}$ can be expressed as $-k^2X$, therefore the reaction-diffusion PDE equations are reduced to:

$$\frac{\partial A}{\partial t} = f_A(A,B) - k^2A \tag{30a}$$

$$\frac{\partial B}{\partial t} = f_B(A,B) - k^2B \tag{30b}$$

where $k = \frac{n\pi}{L}$ $\forall n \in \{0, \mathbb{N}\}$. In the previous section, the linear stability analysis was carried around the steady state with a perturbation δX (diffusion not included). Using Equation 21, which describes the linearised reaction terms around the steady state, Equations 34 can be expressed as:

$$\frac{\partial \delta A}{\partial t} = \frac{\partial f_A(A,B)}{\partial A}\delta A + \frac{\partial f_A(A,B)}{\partial B}\delta B - D_A k^2 \delta A \tag{31a}$$

$$\frac{\partial \delta B}{\partial t} = \frac{\partial f_B(A,B)}{\partial A}\delta A + \frac{\partial f_B(A,B)}{\partial B}\delta B - D_B k^2 \delta B \tag{31b}$$

The general solution of this system can be written as:

$$\delta A = A_0 e^{\sigma t} \cdot e^{ikx} \tag{32a}$$

$$\delta B = B_0 e^{\sigma t} \cdot e^{ikx} \tag{32b}$$

where $X_0 e^{\sigma t}$ represents the amplitude of the perturbations and e^{ikx} represents the spatial oscillations (with k as the wavenumber). In this case, we are interested on the growth or decay of the perturbations over time. Therefore, to observe the stability of the steady state, we will study the amplitude term ($X_0 e^{\sigma t}$):

- If $\sigma > 0$: perturbation (δX) grows making $\frac{\partial \delta A}{\partial t} > 0$. Therefore, the steady state is unstable with diffusion.
- If $\sigma < 0$: perturbation (δX) decays making $\frac{\partial \delta A}{\partial t} < 0$. Therefore, the steady state is stable with diffusion.

If Equations 36a and 36b are substituted into Equations 35a and 35b, the following equations are obtained:

$$\sigma \delta A_0 = \frac{\partial f_A(A,B)}{\partial A}\delta A_0 + \frac{\partial f_A(A,B)}{\partial B}\delta B_0 - D_A k^2 \delta A_0 \tag{33a}$$

$$\sigma \delta B_0 = \frac{\partial f_B(A,B)}{\partial A}\delta A_0 + \frac{\partial f_B(A,B)}{\partial B}\delta B_0 - D_B k^2 \delta A \tag{33b}$$

These pair of equations can again be treated as an eigenvalue-eigenvector problem written in the following way:

$$\sigma \delta X_0 = \begin{bmatrix} \frac{\partial f_A}{\partial A} - D_A k & \frac{\partial f_A}{\partial B} \\ \frac{\partial f_B}{\partial A} & \frac{\partial f_B}{\partial B} - D_B k \end{bmatrix} \delta X_0 \tag{34}$$

where σ can be solving the characteristic polynomial:

$$p(\sigma) = det[J - \sigma I] = 0 \tag{35}$$

In summary, to understand if perturbations around the steady state decay or grow when diffusion is introduced, Equation 39 must be solved to obtain σ. This characteristic polynomial is solved for a range of k's, being $k = \frac{n\pi}{L}\ \forall n \in \{0, \mathbb{N}\}$. The sign of σ is studied for every k, resulting in the dispersion relation (Figure 2 in the Introduction). If $\sigma > 0$, perturbations grow around the steady state and the system is unstable. In the contrary, if $\sigma < 0$, the system is stable and perturbations decay around the steady state. More detail on linear stability analysis can be found in J. D Murray 2002 or Glendinning 1994.

3.3.3 Linear stability analysis implementation

Linear stability analysis is carried out with and without diffusion for all steady states of the system found using Newton-Raphson. The analysis is done for all $k = \frac{n\pi}{L}\ \forall\{n \in \mathbb{N} : n \leq 5000\}$, meaning 5000 k's are sampled using linear stability analysis. L is defined as 100mm. Linear stability analysis for a single steady state takes approximately approximately 1 second.

3.4 Numerical solution by finite-difference methods

Obtaining a solution for a system of equations can become a complex problem if working with a system of 6 non-linear PDEs. Because an analytical expression for the solution is almost impossible to obtain, finite-difference methods are used for cases like this one. Finite-difference methods consists in discretising space and time to approximate the PDE system to a system of algebraic equations that can be easily solved by matrix algebra techniques (Morton and Mayers 2005). By discretising time and space, the two independent variables can be expressed as:

$$t_n = n\Delta t, n = 0, ..., N-1 \tag{36a}$$

$$x_j = j\Delta x, j = 0, ..., J-1 \tag{36b}$$

While Δt and Δx are the time steps and the space steps respectively, N and J are the number of discrete time and space points in our grid. Δt and Δx can be defined as $\Delta t = \frac{T}{N}$ and $\Delta x = \frac{L}{J}$ respectively where T and L are the final time and space values in the grid. The aim is to derive a numerical solution that is approximates to the unknown analytical solution so $U(j\Delta x, n\Delta t) \approx u(j\Delta x, n\Delta t)$, where U is the analytical solution and u is the numerical solution.

When working with a numerical solver, the solver can perturb the systems behaviour due to the effects of the time-step, the integration method or the computer arithmetic. When choosing a scheme to numerically solve a PDE, three different characteristics of the scheme need to be considered: Consistency, stability and convergence. Firstly, for a scheme to be consistent, the truncation error must be reduced as $\Delta t \to 0$ or/and if $\Delta x \to 0$. The truncation error is the resultant from using a simple approximation to represent an exact mathematical formula. Secondly, the numerical method is said to be stable if the error (truncation or round-off) is not magnified as the number of time steps tends to infinity. Finally, as the Lax equivalence theorem states, the scheme is said to be convergent if both consistency and stability. This means that at

any fixed point, if time and space discretisations tend to zero, the numerical solution will tend towards the exact solution. (Ferziger n.d.; G. D. Smith et al. 1985).

The methods chosen to solve this system of equations are Crank-Nicolson for 1 dimension in space and Alternating Direction Implicit Method for 2 dimensions. These methods are chosen because they are both unconditionally stable as shown by von Neumann stability analysis (Strikwerda 2004). The unconditional stability is important to allow for larger Δt and Δx, without getting an amplification of the error. Larger Δt and Δx will result in reduced computationally power. Although CN is less computationally expensive than ADI, it becomes extremely complex when scaled up to multiple dimensions. On the other hand, ADI has a simpler structure in 2 dimensions that can be solved easily using the tridiagonal matrix algorithm (Flaherty 2019). Hence, CN is used to obtain 1D space solutions while ADI is used for 2D.

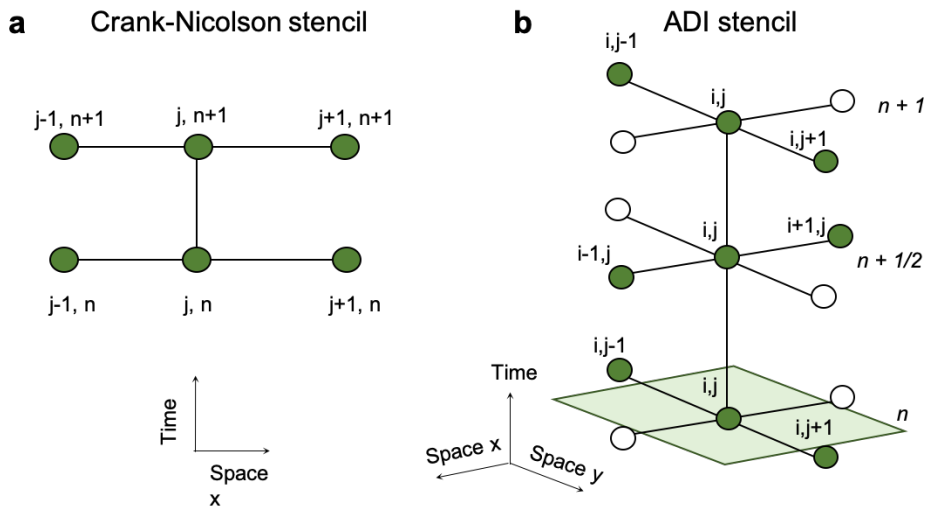


Figure 19: Stencils for numerical solution. A stencil is a geometric representation with nodes and edges, that represents the points of interest for the numerical approximation. The points of interest, which are the ones present in the equations, are shown in green. j and n are the current space and time points. **(A)** Crank-Nicholson stencil used in 1D numerical simulations. The axis are time and space (x). **(B)** Alternating Direction Implicit method stencil used in 2D numerical simulations. The axis are time and 2 dimensional space (x,y).

3.4.1 Crank-Nicolson method

Consider a reaction diffusion system with one space dimension and boundary conditions

$$\frac{\delta u}{\delta t} = f(u) + D\frac{\partial^2 u}{\partial x^2}, \qquad \left.\frac{\partial u}{\partial x}\right|_{x=0,L} = 0 \tag{37}$$

The spatial part of the equation can be approximated to

$$\left.\frac{\partial^2 u}{\partial x^2}\right|_{x=j\Delta x, t=n\Delta t} \approx \frac{1}{2\Delta x^2}\left(U_{j+1}^n - 2U_j^n + U_{j-1}^n + U_{j+1}^{n+1} - 2U_j^{n+1} + U_{j-1}^{n+1}\right), \tag{38}$$

while the production function can be approximated to $f(U_j^n)$. The approximations can be better visualised using the Crank-Nicolson stencil (See Figure 19a). Applying CN's stencil to the grid point (i,j), the reaction-diffusion system can be expressed as

$$\frac{U_j^{n+1} - U_j^n}{\Delta t} = \frac{D}{2\Delta x^2}\left(U_{j+1}^n - 2U_j^n + U_{j-1}^n + U_{j+1}^{n+1} - 2U_j^{n+1} + U_{j-1}^{n+1}\right) + f(U_j^n) \tag{39}$$

By reordering this approximation into a linear equation, the resulting problem is defined by a simple linear equation containing matrices A and B. Where $\mathbf{U}^{n+1} = [U_0^n, ..., U_{J-1}^n]$, the simplified system can be expressed as:

$$\mathbf{U}^{n+1} = A^{-1}(B\mathbf{U}^n + f^n) \tag{40}$$

This method simplifies the complex system into a linear system that can be solved numerically. The solution given will be a 1 dimensional in space solution of the reaction-diffusion system. Although the method is unconditionally stable, the solution can contain oscillations if $\frac{\Delta t}{\Delta x^2} > \frac{1}{2}$ (Trefethen 1996). Therefore, the ratio will be kept below $\frac{1}{2}$ to avoid errors.

3.4.2 Alternating Direction Implicit method

As done in the CN scheme, a reaction diffusion system and its boundary conditions will be consider. However, in this case two spatial dimensions will be introduced.

$$\frac{\delta u}{\delta t} = f(u) + D\left(\frac{\partial^2 u}{\partial x^2} + \frac{\partial^2 u}{\partial y^2}\right), \qquad \left.\frac{\partial u}{\partial x}\right|_{x=0,L} = 0 \qquad \left.\frac{\partial u}{\partial y}\right|_{y=0,L} = 0 \tag{41}$$

If the CN stencil is applied to this 2 dimensional spatial problem, the system would contain banded matrices in the right and left hand sides, that would be very expensive to invert. ADI offers an alternative in which tridiagonal matrices are inverted instead of banded matrices (less computational power required). The characteristic of ADI is the time step Δt is split into two, and each half time step is computed. This means, to compute the change at each time step, first

we compute $U_{i,j}^{n+1/2}$ and from there, $U_{i,j}^{n+1}$ is calculated. This results in two different equations:

$$\begin{aligned}\frac{U_{i,j}^{n+1/2}-U_{i,j}^{n}}{\Delta t/2} &= \frac{D}{2\Delta x^2}\left(U_{i+1,j}^{n+1/2}-2U_{i,j}^{n+1/2}+U_{i-1,j}^{n+1/2}\right)\\ &+\frac{D}{2\Delta y^2}\left(U_{i,j+1}^{n}-2U_{i,j}^{n}+U_{i,j-1}^{n}\right)+\Delta t f(U_{i,j}^{n})\end{aligned} \tag{42a}$$

$$\begin{aligned}\frac{U_{i,j}^{n+1}-U_{i,j}^{n}}{\Delta t/2} &= \frac{D}{2\Delta x^2}\left(U_{i+1,j}^{n+1/2}-2U_{i,j}^{n+1/2}+U_{i-1,j}^{n+1/2}\right)\\ &+\frac{D}{2\Delta y^2}\left(U_{i,j+1}^{n+1}-2U_{i,j}^{n+1}+U_{i,j-1}^{n+1}\right)+\Delta t f(U_{i,j}^{n+1/2})\end{aligned} \tag{42b}$$

In the first half time step (Equation 46a), the x derivative is taken implicitly, and in the second half time step (Equation 46b), the y derivative is taken implicitly. As done in CN, the approximation is reordered into a linear system. Two families of linear systems appear:

$$A\mathbf{U}_{x,i}^{n+1/2} = \mathbf{b}_i + \mathbf{f}(\Delta t\mathbf{U}_{x,i}^{n}), \quad i = 0, ..., I-1 \tag{43a}$$

$$C\mathbf{U}_{y,j}^{n+1} = \mathbf{d}_j + \mathbf{f}(\Delta t\mathbf{U}_{y,j}^{n+1/2}), \quad j = 0, ..., J-1 \tag{43b}$$

Again, this method also simplifies a complex system into a linear system that can be solved numerically, as in CN. However, this method allows for the introduction of a new spatial dimension and therefore produces a 2D spatial solution. The workings of this method can be better understood with the ADI stencil (See Figure 19b). ADI will be used to visualise patterns in 2D.

3.4.3 Analysis of numerical solution

Speed of pattern formation and pattern wavelength are identified by performing additional analysis on the 1D numerical data.

Time for pattern formation The development of the pattern follows a certain behaviour: The molecule concentrations are initially homogeneous; then a pattern gets formed progressively; and finally, the pattern is in its final state and the solution remains constant. The time for pattern formation is measured by comparing the solution (one point in space) at every time point to the solution at the final time point. If the difference if smaller than a tolerance value of 10^{-4} that time point is taken as the convergence time point were the pattern has finished to develop.

Wavelength prediction from numerical Solution The findpeaks package is used from the *scipy.signal* python library. All peaks in the final time point of the 1D simulation are found through the findpeaks package. The average distance between peaks is taken and that distance is averaged throughout the 6 species.

3.5 Sampling method

If no parameters from a system are known, sampling of the whole parameter space must be done to understand the systems behaviour. Different methods exist for sampling parameter spaces. Several studies have shown that latin-hypercube sampling (LHS) has a higher efficiency than grid sampling or random sampling when searching through high dimensional spaces. The efficiency over grid sampling might be explained because not all dimensions of the model are important, meaning some parameters might be sloppy. Therefore not all parameters have to be explored thoroughly as done in grid sampling (Bergstra et al. 2012; Iman et al. 1980). The three sampling regimes are shown in Figure 20a.

In LHS, a distribution and a number of desired samples is given as an input. The algorithm then divides the space sections according to the distribution given (e.g in a normal distribution, more sections will appear near the mean value). Then, one sample is positioned randomly somewhere in each section. For a 2 dimensional parameter space, no samples can be in the same column or row. This is scaled up to high multidimensional spaces such as the 22 dimensional space of the model proposed. This scheme is shown in Figure 20b.

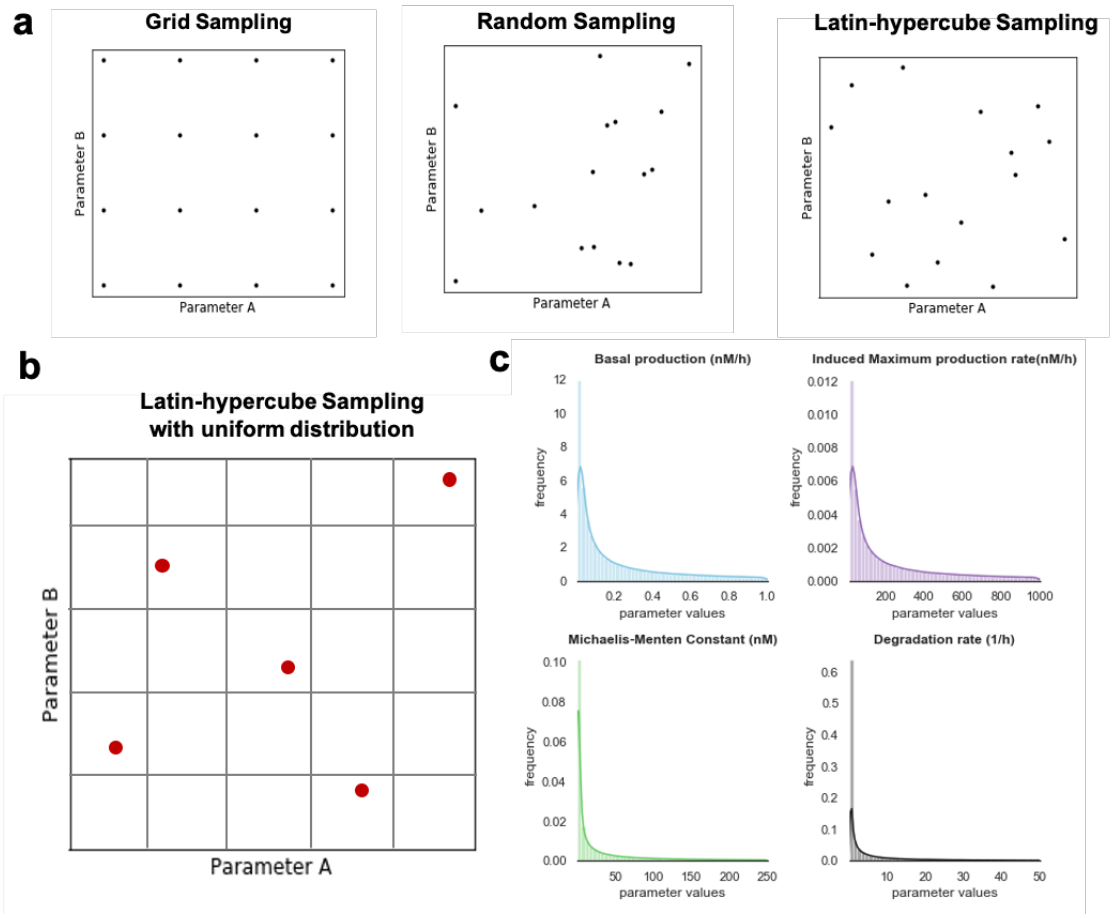


Figure 20: Sampling method for high dimensional spaces. **(A)** Types of potential sampling methods. **(B)** Latin-Hypercube sampling with uniform distribution for a 2 dimensional parameter space. Space is separated in 5 sections for each parameter, leading to 5 samples (red dot). No sample is present in the same row or column. **(C)** Parameter distributions used for Latin-hypersube sampling. The 4 different types of parameters have different distributions depending on the ranges defined. All of them are uniform distributions in log-scale.

The distribution given as an input to the LHS algorithm, will be the resulting distribution of your samples. For the purpose of this search, the distributions chosen are uniform distributions in a logarithmic scale (log-uniform distribution). The uniform distribution, although it does not describe many phenomena in biology, can be useful when no prior knowledge is known about the parameters (Frank 2009). The logarithmic component is used to make sure parameters from all scales are represented equally, instead of having a higher frequency of values from larger scales. Log-normal distributions are commonly used for modelling in biology, however due to the nonexistent prior knowledge on our parameter values, the log-uniform is used instead. The log-uniform distribution is defined within a certain range, which varies depending on the parameter type. The distributions for each parameter type are shown in Figure 20c. Overall, 1 million samples where produced using the LHS with a uniform distribution in log-scale. All

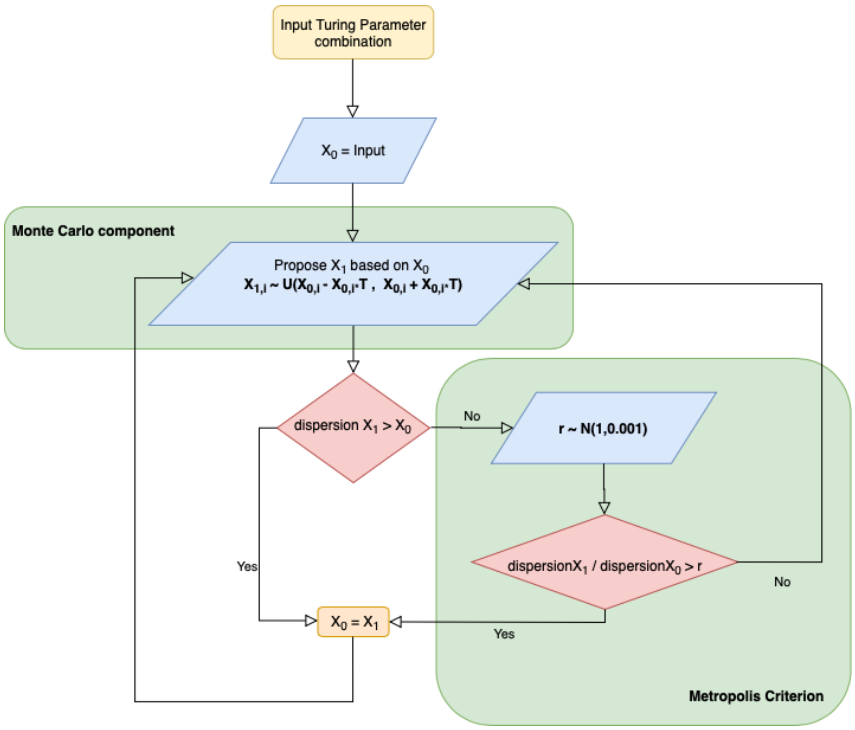


Figure 21: Adapted random walk-Metropolis algorithm workflow.

parameters except the diffusion rates (d) and the cooperativity (n) where sampled using LHS and the distributions shown in Figure 20c.

3.6 Dispersion peak height optimisation: Adapted random walk Metropolis

In this section, the optimisation of the dispersion peak height using an adapted random walk-Metropolis (RWM) algorithm will be explained. The RWM algorithm is a common type of Markov Chain Monte Carlo (MCMC) method that uses a Metropolis Algorithm. The RWM algorithm is used for sampling a variable to understand its probability distribution, without getting stuck in local maxima. This algorithm is used in a Bayesian context when trying to fit a model with parameters θ to a data D. A probability distribution is obtained, which suggest what parameters θ are better to represent the data D. However, in this case, the aim is not to fit a model to any data D, but to maximise the dispersion peak height. Therefore, a variant of the RWM algorithm will be developed as shown in Figure 21.

The starting point of the algorithm is to propose a Turing parameter set which will be the starting parameter set to be optimised in the process. From that parameter set, a new parameter set is proposed where all parameters are varied slightly. The variation is chosen randomly from a uniform distribution around the parameter value. The uniform distribution is

defined as $U(X_0 - X_0T, X_0 + X_0T)$, where X_0 is the initial parameter to be varied and T is a temperature constant that will define the amount of variation to be applied. In this case, $T = 0.1$. So if $X_0 = 100$, the uniform distribution is U(90,110). This step is done for all parameters of the parameter set at every iteration, producing a new parameter set X_1. The Markov Chain component is present because the step taken is only dependent on the current state, and not on information prior to that. The Monte Carlo is due to the randomness involved in choosing the new parameter set. In the normal RWM algorithm, the posterior of X_0 and X_1 are compared to see which parameter set is a better fit to the data D. However, for the purpose of optimising dispersion, the posterior is neither available nor relevant. Instead the dispersion peak height value is used. Once the new step X_1 is taken, the dispersion peak height (d_{X_1}) is calculated and compared to the dispersion peak height of X_0 (d_{X_0}). If the dispersion peak height has improved, $d_{X_1} > d_{X_0}$, the move is accepted and X_1 becomes X_0. If no improvement has been made, $d_{X_1} < d_{X_0}$, the Metropolis algorithm comes in to place: The ratio of the dispersions is calculated, $r = \frac{d_{X_1}}{d_{X_0}}$ and compared to a normal distribution $N(1, 0.001)$. If the ratio is higher than a random number from the distribution $N(1, 0.001)$, the move is accepted and X_1 becomes X_0. Otherwise, the move is rejected. This ensures that big decreases in the dispersion peak height ($r \ll 1$) are not accepted while small decreases ($r \approx 1$) are accepted. Usually, the RWM uses a distribution $U(0, 1)$ for this step. An optimisation with this distribution was attempted, resulting in no significant improvement of the dispersion peak height. Therefore, the variant of the $N(1, 0.001)$ was introduced to ensure a more strict regime is in place, hence reducing the number of accepted negative steps.

3.7 Sensitivity analysis

Using sensitivity analysis, a robustness value is computed for every one of the 22 parameter in the parameter set. This technique consists in finding the upper and lower boundaries of each parameter, where the parameter set still produces Turing patterns. The workflow to test for the robustness of one parameter is show in Figure 22a. This workflow is carried out 22 times for each parameter set, to cover all parameters in the model, giving an output shown in Figure 22b.

Starting on one parameter of a parameter set, the upper and lower boundaries are calculated. To test the upper boundary, a increment of 10% of the parameter value is added, and the new parameter set analysed for Turing patterns. If Turing patterns are still produced, the

process continues until the algorithm reaches a parameter set where no Turing pattern is formed. The same process is carried out on the other direction, by decreasing the parameter value by 10% at every iteration. The result is a tuple containing upper and lower bounds for every parameter of the parameter set.

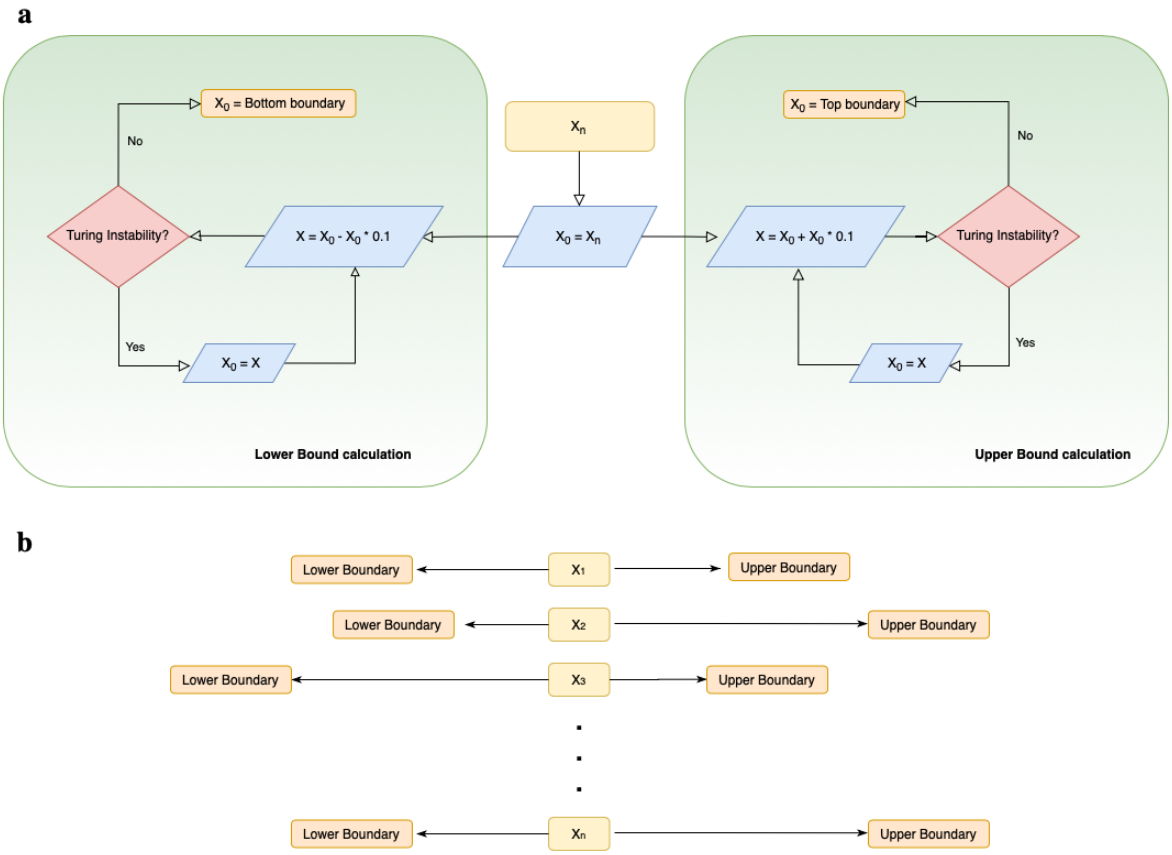


Figure 22: Sensitivity Analysis. (A) Workflow of sensitivity analysis for one parameter. This process is done for the 22 parameters of the model. **(B)** Result of sensitivity analysis. Lower boundaries and upper boundaries are obtained, from a initial parameter value.

4 Discussion

Prior to this investigation, it was not clear whether the biological implementation of circuit 3954 with 6 components could generate patterns. This synthetic implementation is an abstraction of the original 3 node network, where delays and non-linearities can deviate the behaviour and make it a non-Turing pattern circuit. Therefore an accurate 6 equation model that proves its Turing pattern capabilities is needed. A 6 equation model that describes this biological circuit was developed, and its parameter space searched for Turing patterns. The biological implementation of circuit 3954, although not often, has the potential to generate Turing patterns as seen in Figure 10. This a key validation step needed in order to build and test the circuit experimentally

to find Turing patterns *in vitro*.

Although the parameter space supporting Turing pattern formation is small, as explained in the introduction, the circuit is highly tunable and therefore can be manipulated to hit the Turing parameter space. By looking at the parameter distributions in Figure 11, we can observe for each parameter, what values lead to some sort of pattern behaviour (not only Turing patterns but also oscillatory or unstable Turing patterns). By increasing certain parameters such as the degradation rate of HSL (mu_b), we increase the chance of finding a pattern state, and therefore finding Turing space. This parameter bias information is not completely accurate as it is based on parameter sets that form patterns that are not only Turing (oscillatory and unstable Turing as well). Although is useful to shed some light on parameter bias, a bigger parameter search should be carried out to obtain more Turing parameter sets and therefore better understand parameter ranges in which the system can form Turing patterns. If a bigger parameter search indicated some bias for a certain parameter range in order to hit Turing space, this could potentially be implemented experimentally by tuning the circuit to fall into that parameter range (e.g. increase the degradation rate of HSL by adding DAPG to the solution).

Another important aim of this project was to understand robustness of the circuit and to find regions of the parameter space with increased robustness. The technique chosen to study robustness is sensitivity analysis, which explores robustness of each parameter independently. No major conclusions can be drawn from the robustness study of the three Turing parameter sets as shown in Figure 12. Also, no relationship can be derived from robustness and dispersion peak height as seen in Figure 14. The lack of conclusive results from sensitivity analysis might suggest that it is not the most suitable technique to study robustness of such systems. In future research, robustness could be assessed for a parameter set as a whole, by exploring the magnitude of Turing space around that parameter set. This can be done by analysing parameter sets around the initial one and assessing what percentage of those parameter sets produces Turing patterns. This alternative way of analysing robustness can be applied to understand robustness of the three Turing parameter sets or to continue studying the relationship between robustness and dispersion peak height.

Although no relationship between robustness and dispersion peak height was found, the optimi-

sation of dispersion peak height provided some interesting and unexpected insights. Dispersion peak height turned out to have a direct relationship with speed of pattern formation and shape of the pattern. This is a novel discovery not previously described in the literature. The dispersion peak height is linked to the time for pattern formation with a $y = \frac{1}{x}$ relationship. A higher dispersion peak is also related to dotted-shaped patterns. If shape and speed are both correlated to dispersion peak height, it can be concluded that a dotted-shape pattern will always take less time to form. Interestingly, tuning of certain parameters can drive the increase of dispersion peak height theoretically (See Figure 17). Therefore, tuning these parameters accordingly can result in regulation of shape and speed of pattern formation. Understanding and controlling speed of pattern formation and shape is key for several purposes: Firstly, because slow pattern development times are a limiting factor to screen a big number of biofilms to search for Turing patterns experimentally, being able to control the pattern development time would enable to speed up this process. Secondly, controlling the shape could be extremely useful for biotechnological applications, where specific pattern shapes are needed. Lastly, controlling the pattern in this manner would be key to prove the patterns obtained experimentally are Turing patterns. Previous studies have managed to synthesise patterning biofilms (Karig et al. 2018; Sekine et al. 2018), however these studies suggested the patterns were stochastic and solitary patterns. If patterns are obtained using circuit 3954, one of the main challenges would be to prove their Turing pattern nature. If speed of pattern formation and shape can be tuned as predicted by the deterministic model developed in this study, the Turing nature of the patterns generated can be demonstrated. For future research, the relationship between pattern wavelength and parameter values could be studied as well, to enable the tuning of the pattern wavelength experimentally.

Carrying out such a big sampling of the parameter space, not only allowed us to find Turing patterns, but also to understand what combinations of steady states generate which types of patterns. Literature on Turing patterns is very centered on monostable systems (meaning there is only one steady state). Multistable systems were analysed in this sampling exercise, showing the potential of these multistable systems to generate not only Turing patterns but also a newly discovered type of patterning regime: the unstable Turing pattern. This finding suggests many more types of systems than previously thought might have the potential to generate interesting patterning regimes. Studies like S. Smith and Dalchau 2018 and Kuznetsov and Polezhaev 2020 have also claimed that criteria for Turing pattern systems has to be widened. In these studies

they find systems that don't follow the classical diffusion-driven instability, which are able to form Turing patterns. Further work should be done to understand the role of multistable systems and non-classical Turing dispersion relations in patterning. The existence of these alternative patterning systems might explain why such a a small Turing parameter space is found when using the current criteria for Turing pattern emergence.

This newly developed 6 equation model is theoretically more accurate, as it describes the behaviour of all the species over time. However, even when making assumptions such as the equivalence of certain Michaelis-Menten constants or assuming same degradation rate for all non-diffusing species, the model is still highly complex. Firstly, the 22 dimensional parameter space is infinitely large and complex to explore. Secondly, the computational power of carrying numerical simulations, Newton-Raphson and linear stability analysis for a 6 equation non-linear system is extremely high, making the whole process of parameter space exploration much slower. To ease this task, experimental data could be used to fit the model and fix the stiff and sloppy parameters to a specific value or range. Once the model is fitted, certain parameters can be fine-tuned to obtain the Turing pattern behaviour that is desired. The same fine-tuning will need to be done experimentally to obtain *in vitro* Turing patterns. To fit the model, experimental data on the species concentrations over time is needed. For the 6 equation model two experimental options remain open to obtain data on the 6 species over time: time-series mass spectrometry or time-series qPCR. In both cases, samples are taken periodically for 24 hours and then each sample run through mass spectrometry or qPCR to obtain a relative measure of each species over time. No research has been carried out before using this type of time-series technique to fit synthetic biology models. Although a challenging experiment, if successful, it could enable to determine parameter values of the model and to better understand the behaviour of the synthetic circuit. If this data is unobtainable, a 3 equation model would remain as an option. This model would describe promoter's activity instead of species concentration. The promoters modelled would be P_{sal} from node A, P_{sal} from node B and P_{cI*} from node C. A fluorescence reporter would have to be introduced after each one of this promoters and fluorescence measured over time for 24 hours. Two issues are present with this approach, which are the reason why the 6 model equation was developed. Firstly, there is no resolution at the species level, so ignored interactions within one node might cause the model to wrongly predict the systems behaviour. Finally, no third fluorescent reporter that could be introduced in node A, and that produced

enough fluorescence to be detected has been found. Both models with their advantages and disadvantages remain as an option to guide the design and tuning of the synthetic Turing circuit 3954.

Engineering synthetic biological Turing patterns *in vitro* has been one of the objectives of many researchers in the fields of synthetic biology and morphogenesis for several reasons: Firstly, it would enable us to better understand key elements of developmental biology and explain how patterns self-assemble from homogeneous tissues. Secondly, controlling the synthesis of these patterns would enable scientists to explore downstream bioengineering applications in the fields of nanotechnology or regenerative medicine. Due to the complexity of the circuit and of the parameter space capable of generating Turing patterns, mathematical modelling is required to help in the design and tuning of these genetic circuits. An accurate model to support the experimental work has been developed in this project as well as ways to control speed of pattern formation and shape of the pattern simultaneously. These advances will hopefully be used to ensure the successful synthesis of engineered biological Turing systems.

References

[1] Jens Bo Andersen, Claus Sternberg, Lars Kongsbak Poulsen, Sara Petersen Bjørn, Michael Givskov, and Søren Molin. "New unstable variants of green fluorescent protein for studies of transient gene expression in bacteria". In: *Applied and environmental microbiology* 64.6 (1998), pp. 2240–2246.

[2] René Augustin, André Franke, Konstantin Khalturin, Rainer Kiko, Stefan Siebert, Georg Hemmrich, and Thomas CG Bosch. "Dickkopf related genes are components of the positional value gradient in Hydra". In: *Developmental biology* 296.1 (2006), pp. 62–70.

[3] Subhayu Basu, Yoram Gerchman, Cynthia H Collins, Frances H Arnold, and Ron Weiss. "A synthetic multicellular system for programmed pattern formation". In: *Nature* 434.7037 (2005), pp. 1130–1134.

[4] James Bergstra, James Bergstra@umontreal Ca, and Yoshua Bengio@umontreal Ca. *Random Search for Hyper-Parameter Optimization Yoshua Bengio*. Tech. rep. 2012, pp. 281–305. URL: `http://scikit-learn.sourceforge.net.`.

[5] A David Blest. "The function of eyespot patterns in the Lepidoptera". In: *Behaviour* 11.2-3 (1957), pp. 209–258.

[6] V Castets, E Dulos, and P De Kepper. "Experimental Evidence of a Sustained Standing Turing-Type Nonequilibrium Chemical Pattern". In: *Physical Review Letters* 64.24 (1990).

[7] Chenli Liu et al. "Sequential Establishment of Stripe Patterns in an Expanding Cell Population". In: *Science* 334.6053 (Oct. 2011), pp. 238–241. DOI: `10.1126/science.1209042`.

[8] Antonio J. Conejo and Luis Baringo. *Power System Operations*. Power Electronics and Power Systems. Cham: Springer International Publishing, 2018. ISBN: 978-3-319-69406-1. DOI: `10.1007/978-3-319-69407-8`. URL: `http://link.springer.com/10.1007/978-3-319-69407-8`.

[9] Stijn P De Langhe, Frédéric G Sala, Pierre-Marie Del Moral, Timothy J Fairbanks, Kenneth M Yamada, David Warburton, Robert C Burns, and Saverio Bellusci. "Dickkopf-1 (DKK1) reveals that fibronectin is a major target of Wnt signaling in branching morphogenesis of the mouse embryonic lung". In: *Developmental biology* 277.2 (2005), pp. 316–331.

[10] Luis Diambra, Vivek Raj Senthivel, Diego Barcena Menendez, and Mark Isalan. "Cooperativity to increase turing pattern space for synthetic biology". In: *ACS Synthetic Biology* 4.2 (Feb. 2015), pp. 177–186. ISSN: 21615063. DOI: 10.1021/sb500233u.

[11] JH Ferziger. "Peric (2002) Computational methods for fluid dynamics". In: *Berlin, Springer.* ().

[12] Joseph E. Flaherty. *Multi-Dimensional parabolic problems.* May 2019.

[13] S. A. Frank. *The common patterns of nature.* Aug. 2009. DOI: 10.1111/j.1420-9101.2009.01775.x.

[14] Rudolf Gesztelyi, Judit Zsuga, Adam Kemeny-Beke, Balazs Varga, Bela Juhasz, and Arpad Tosaki. *The Hill equation and the origin of quantitative pharmacology.* July 2012. DOI: 10.1007/s00407-012-0098-5.

[15] A Gierer and H Meinhardt. *A Theory of Biological Pattern Formation.* Tech. rep. 1972.

[16] Paul. Glendinning. *Stability, instability, and chaos : an introduction to the theory of nonlinear differential equations.* Cambridge University Press, 1994, p. 388. ISBN: 0521415535.

[17] Jeremy B.A. Green and James Sharpe. "Positional information and reaction-diffusion: Two big ideas in developmental biology combine". In: *Development (Cambridge)* 142.7 (Apr. 2015), pp. 1203–1211. ISSN: 14779129. DOI: 10.1242/dev.114991.

[18] Judit Horváth, István Szalai, and Patrick De Kepper. "An experimental design method leading to chemical Turing patterns". In: *Science* 324.5928 (2009), pp. 772–775.

[19] Ronald L Iman, James M Davenport, and Diane K Zeigler. *Latin hypercube sampling (program user's guide).[LHC, in FORTRAN].* Tech. rep. Sandia Labs., Albuquerque, NM (USA), 1980.

[20] J. D Murray. *Mathematical Biology II: Spatial Models and Biomedical Applications.* Third Edit. Springer, 2002.

[21] David Karig, K. Michael Martini, Ting Lu, Nicholas A. DeLateur, Nigel Goldenfeld, and Ron Weiss. "Stochastic Turing patterns in a synthetic bacterial population". In: *Proceedings of the National Academy of Sciences of the United States of America* 115.26 (June 2018), pp. 6572–6577. ISSN: 10916490. DOI: 10.1073/pnas.1720770115.

[22] Shigeru Kondo and Takashi Miura. *Reaction-diffusion model as a framework for understanding biological pattern formation.* Sept. 2010. DOI: 10.1126/science.1179047.

[23] Andrej (Princeton) Košmrlj. "Seminar 3: Multicellular Structure Formation in 3D, lecture notes". In: *Physics of Life Online Summer School* (2020).

[24] Maxim Kuznetsov and Andrey Polezhaev. "Widening the criteria for emergence of Turing patterns". In: *Chaos* 30.3 (Mar. 2020). ISSN: 10897682. DOI: 10.1063/1.5140520.

[25] I. Lengyel and I. R. Epstein. "A chemical approach to designing Turing patterns in reaction-diffusion systems". In: *Proceedings of the National Academy of Sciences of the United States of America* 89.9 (1992), pp. 3977–3979. ISSN: 00278424. DOI: 10.1073/pnas.89.9.3977.

[26] Luciano Marcon, Xavier Diego, James Sharpe, and Patrick Müller. "High-throughput mathematical analysis identifies Turing networks for patterning with equally diffusing signals". In: *Elife* 5 (2016), e14022.

[27] Mitsugu Matsushita and Hiroshi Fujikawa. *DIFFUSION-LIMITED GROWTH IN BACTERIAL COLONY FORMATION.* Tech. rep. 1990, pp. 498–506.

[28] Hans Meinhardt and Alfred Gierer. *Pattern formation by local self-activation and lateral inhibition.* Tech. rep. 2000, pp. 753–760.

[29] Adam J. Meyer, Thomas H. Segall-Shapiro, Emerson Glassey, Jing Zhang, and Christopher A. Voigt. "Escherichia coli "Marionette" strains with 12 highly optimized small-molecule sensors". In: *Nature Chemical Biology* 15.2 (Feb. 2019), pp. 196–204. ISSN: 15524469. DOI: 10.1038/s41589-018-0168-3.

[30] Keith W Morton and David Francis Mayers. *Numerical solution of partial differential equations: an introduction.* Cambridge university press, 2005.

[31] Tetsuya Nakamura, Naoki Mine, Etsushi Nakaguchi, Atsushi Mochizuki, Masamichi Yamamoto, Kenta Yashiro, Chikara Meno, and Hiroshi Hamada. "Generation of Robust Left-Right Asymmetry in the Mouse Embryo Requires a Self-Enhancement and Lateral-Inhibition System". In: *Developmental Cell* 11.4 (Oct. 2006), pp. 495–504. ISSN: 15345807. DOI: 10.1016/j.devcel.2006.08.002.

[32] Žiga Pušnik, Miha Mraz, Nikolaj Zimic, and Miha Moškon. "Computational analysis of viable parameter regions in models of synthetic biological systems". In: *Journal of Biological Engineering* 13.1 (Sept. 2019). ISSN: 17541611. DOI: 10.1186/s13036-019-0205-0.

[33] Jelena Raspopovic, Luciano Marcon, Laura Russo, and James Sharpe. "Digit patterning is controlled by a Bmp-Sox9-Wnt Turing network modulated by morphogen gradients". In: *Science* 345.6196 (2014), pp. 566–570.

[34] Jean Tyson Schneider. *PERFECT STRIPES FROM A GENERAL TURING MODEL IN DIFFERENT GEOMETRIES*. Tech. rep. 2012.

[35] Natalie Scholes and Mark Isalan. *A three-step framework for programming pattern formation*. Oct. 2017. DOI: 10.1016/j.cbpa.2017.04.008.

[36] Natalie Scholes, David Schnoerr, Mark Isalan, and Michael Stumpf. "A Comprehensive Network Atlas Reveals That Turing Patterns Are Common but Not Robust". In: *Cell Systems* 9.3 (Sept. 2019), pp. 243–257. DOI: 10.1016/j.cels.2019.07.007.

[37] Ryoji Sekine, Tatsuo Shibata, and Miki Ebisuya. "Synthetic mammalian pattern formation driven by differential diffusivity of Nodal and Lefty". In: *Nature Communications* 9.1 (Dec. 2018). ISSN: 20411723. DOI: 10.1038/s41467-018-07847-x.

[38] Gordon D Smith, Gordon D Smith, and Gordon Dennis Smith Smith. *Numerical solution of partial differential equations: finite difference methods*. Oxford university press, 1985.

[39] Stephen Smith and Neil Dalchau. "Beyond activator-inhibitor networks: the generalised Turing mechanism". In: (Mar. 2018). URL: http://arxiv.org/abs/1803.07886.

[40] Martin Stevens, Innes C. Cuthill, Amy M.M. Windsor, and Hannah J. Walker. "Disruptive contrast in animal camouflage". In: *Proceedings of the Royal Society B: Biological Sciences* 273.1600 (Oct. 2006), pp. 2433–2438. ISSN: 14712970. DOI: 10.1098/rspb.2006.3614.

[41] Sören Strauss, Janne Lempe, Przemyslaw Prusinkiewicz, Miltos Tsiantis, and Richard S. Smith. "Phyllotaxis: is the golden angle optimal for light capture?" In: *New Phytologist* 225.1 (Jan. 2020), pp. 499–510. ISSN: 14698137. DOI: 10.1111/nph.16040.

[42] John C Strikwerda. *Finite difference schemes and partial differential equations*. SIAM, 2004.

[43] Lloyd Nicholas Trefethen. "Finite difference and spectral methods for ordinary and partial differential equations". In: (1996).

[44] A Turing. "The chemical basis of differentiation". In: *Philos Trans R Soc London Ser B* 237 (1952), pp. 37–72.

[45] L Wolpert. *Positional Information and the Spatial Pattern of Cellular Differentiationt*. Tech. rep. 1969, pp. 1–47.

[46] M. Mocarlo Zheng, Bin Shao, and Qi Ouyang. "Identifying network topologies that can generate turing pattern". In: *Journal of Theoretical Biology* 408 (Nov. 2016), pp. 88–96. ISSN: 10958541. DOI: 10.1016/j.jtbi.2016.08.005.