Computational design and characterisation of synthetic genetic switches

Miriam Leon

A thesis submitted in partial fulfilment of the requirements for the degree of:

Doctor of Philosophy of
University College London

2016

Primary supervisor:
Dr. Chris P Barnes

Secondary supervisor:
Prof. Geraint MH Thomas

I, Miriam Leon, confirm that the work presented in this thesis is my own. Where information has been derived from other sources, I confirm that this has been indicated in the thesis.

Abstract

blabla

Contents

List of Figures

List of Tables

Abbreviations

ABC Approximate Bayesian Computation.

ATc anhydrotetracycline.

BSA Bovine Serum Albumin.

CS-LU Lu classic switch.

CS-MA Mass action classic switch.

DNA Deoxyribonucleic acid.

dNTPs Deoxynucleotide.

DP-LU Lu double positive switch.

DP-MA Mass action double positive switch.

GFP green fluorescent protein.

GPU Graphical Processing Units.

IPTG Isopropyl-beta-D-thiogalactopyranoside.

KS Kolmogorov-Smirnov.

LB Lysogeny broth.

MCMC Markov Chain Monte Carlo.

MJP Markov jump process.

ODE Ordinary differential equation.

PCA Principal component analysis.

PCR Polymerase Chain Reaction.

QSSA quasi-steady state approximation.

SDE Stochastic differential equation.

SMC Sequential Monte Carlo.

SP-LU Lu single positive switch.

Acknowledgements

blabla

1 Introduction

1.1 Quantitative dynamical modelling in synthetic biology

1.2 Contents of this thesis

1.2.1 Outline

1.2.2 Publications

Parts of this thesis has been published in the following articles:

1. Woods, M., Leon, M., Perez-Carrasco, R., Barnes, C. P. (2015). 'A statistical approach reveals designs for the most robust stochastic gene oscillators'. bioRxiv, 025056.

2. Woods, M., Leon, M., Perez-Carrasco, R., Barnes, C. P. (2015). 'A statistical approach reveals designs for the most robust stochastic gene oscillators'. bioRxiv, 025056.

2 Background

2.1 Introduction to synthetic biology

Synthetic biology aims at the rational design and construction of biological parts, devices, and systems in order to engineer organisms to perform new tasks (Lu, Khalil, & Collins 2009; Andrianantoandro et al. 2014). A part is a basic unit, like a promoter or a ribosome binding site that when combined with other parts will make a functional unit, a device (Heinemann & Panke 2006). A device processes inputs, performs functions and produces outputs (Andrianantoandro et al. 2014). A system comprises of a collection of devices.

Emphasis is put on the use of engineering principles such as modularity, standardisation, use of predictive models and the separation of design and construction (Agapakis & Silver 2009; Heinemann & Panke 2006). A hierarchy similar to computer science is used, with cells, pathways and biochemical reactions acting as computers, modules and gates respectively (Andrianantoandro et al. 2014).

Numerous applications of synthetic biology have emerged, from altering existing metabolisms to producing synthetic drugs (Holtz & Keasling 2010) or creating new synthetic life forms (Agapakis & Silver 2009). Despite the successes there is still a lack of predictive power due to the stochasticity and lack of complete knowledge of the cellular environment (Andrianantoandro et al. 2014).

Synthetic biology is now entering an age where simple synthetic circuits have been built, such as toggle switches (**Isaacs:2003ht**; **Deans:2007cya**; Gardner, Cantor, & Collins 2000; Kramer et al. 2004; Ham et al. 2008; Friedland et al. 2009), oscillators (Stricker et al. 2008; Fung et al. 2005; Tigges et al. 2009) and pulse generators (Basu et al. 2004), but larger circuits have proven more difficult (XXX). The leap from building low-level circuits to assembling them into complex networks has yet to be made successfully (Lu, Khalil, & Collins 2009), and predictable circuit behaviour remains challenging (XXX). Efforts to do so are plagued by intra-circuit

crosstalk and incompatibility, as well as cellular noise, which can render synthetic networks non-functional *in vivo* (XXX).

2.2 System design in synthetic biology

Creating synthetic devices that are robust to changing cellular contexts will be key to the success of synthetic biology. Unknown initial conditions and parameter values as well as the variability of the cellular environment, extracellular noise and crosstalk makes the majority of synthetic genetic devices non-functional (Chen, Chang, & Lee 2009). Designing devices robust to this environment will lead to reliable behaviour of the systems. When faced with a set of competing designs for a given genetic circuit, one is likely to choose the simplest possible model that can achieve the desired behaviour. However, simple systems are often the least robust. Feedback loops are well known key regulatory motifs (Brandman et al. 2005). Negative feedback loops are essential for homeostasis and buffering (Thomas, Thieffry, & Kaufman 1995) thus increasing robustness to extrinsic noise sources and positive feedback loops can generate multistationarity in a system (Thomas, Thieffry, & Kaufman 1995). Incorporating this kind of additional feedback interactions can make a design more robust and reliable. Maximising production is an important goal for a metabolic engineering project if it is to produce an economically viable substance (Holtz & Keasling 2010). Network topologies and parameter values of different toggle switch designs are explored here in order to identify the design that maximises robustness and distance between steady states. This ensures the reliable production of the product with the greatest distance between the on and off states of the switch. In the future, by selecting the system components accordingly, the parameter values can be adjusted *in vivo*. For example, the parameter value corresponding to the translation initiation rate can be chosen by selecting the appropriate RBS sequence which given a nucleotide sequence will produce the desired rate (Holtz & Keasling 2010), a method developed by Salis, Mirsky, & Voigt (2009). Another method to tweak the parameter values *in vivo* is to select the promoter to have the strength corresponding to the levels of gene expression and repression desired. Activity of each promoter can be measured and standardised (Kelly et al. 2009) making this process possible. For a system requiring more than one promoter, these can be efficiently selected from a promoter library using a genetic algorithm created by Wu, Lee, & Chen (2011). These standardised interchangeable components with known sequence and activity are what synthetic biology classes

as BioBricks (Kelly et al. 2009; Canton, Labno, & Endy 2008). These can be selected and used to construct a desired system and replicate the parameter values found in the scan presented here.

The first computational approach for the tuning of robust synthetic networks was that of Batt et al. (2007) where they examined the problem of finding a subset of the parameter set for which a given property was satisfied for all the parameters. Chen, Chang, & Lee (2009) used the fuzzy dynamic game method to solve the minimax regulation design problem of synthetic genetic networks. In that method the worst case effect of all disturbances is minimised for a given network. An evolutionary algorithm has also been used to solve the robust design problem by evolving the parameters of the system in order to make it more robust to cellular disturbances by **Chen:2011hj** The added value of the methodology presented here is that the network structure in addition to the network parameters are adjusted to select a network that can robustly create the desired behaviour.

2.3 Introduction to Biochemical Modelling

2.3.1 Graphical representation of biochemical systems

It is common to represent coupled biochemical reactions graphically. In a graph, as shown in Figure **??**, nodes represent the species and the edges represent an interaction between the species it connects, in which a transcription factor directly affects the transcription of a gene (**alon:2007b**). An arrow at the end of an arc represents activation, i.e. that when the transcription factor binds to the promoter the rate of transcription of the gene increases. A flat line perpendicular to the arc at the end of an arc represents repression, i.e. that when the transcription factor binds to the promoter the rate of transcription of the gene decreases (**alon:2007b**).

2.3.2 Deterministic and Stochastic modelling

Modelling attempts to describe the elements and dynamics of the biochemical system of interest. It is a tool used for integrating knowledge and experimental data as well as for making predictions about the behaviour of the system (**wilkinson:2006**). When modelling a biochemical system it is generally assumed that the rates of a reaction are directly proportional to the concentration of the reactants, raised to the power of their stoichiometry (**wilkinson:2006**). This is known as mass-action kinetics and is used in this work to model the various systems. There are two main ways

of modelling a system, deterministically and stochastically. Deterministic modelling utilises Ordinary differential equation (ODE) and models the concentrations of the species (proteins or other molecules) by time-dependent variables (de Jong 2002). Rate equations are used to model gene regulation where the rate of production of a species is a function of the concentrations of the other species (de Jong 2002). When modelling deterministically the model is viewed as a system which, with sufficient knowledge of the system, its behaviour is entirely predictable. Nevertheless we are still a long way away from having complete knowledge of a system of interesting size (**wilkinson:2006**). Deterministic modelling also assumes a homogenous mixture where species concentrations vary continuously and deterministically, assumptions that often are not met *in vivo*. A cell is spatially and temporally separated, due to small molecule numbers and fluctuations in the timing of processes (de Jong 2002).

In stochastic modelling, species are measured in discrete amounts rather than concentrations and a joint probability distribution is used to express the probability that at time t the cell contains a number of molecules of each species (de Jong 2002). It takes uncertainty into account and does not assume a homogenous mix. It is thus often more appropriate for modelling cellular systems, although more computationally intensive. In stochastic systems the Gillespie algorithm is widely used to simulate the time-evolution of the state of the system (**wilkinson:2006**). The algorithm, developed by Gillespie (1977) can be summarised in four steps:

1. Number of molecules in the system initialised
2. Two random numbers generated, one to determine which reaction will occur next and one to determine the time step
3. Time step increased and molecule counts updated according to Step 2
4. Repeat from Step 2 until total simulation time reached

2.3.3 Steady state and stability

In a steady state, the state of a system remains fixed. In non-linear systems, like the ones systems biology deals with, there is generally not an analytical solution thus the system has to be solved numerically. A stable steady state is defined as a fixed point whose nearby points approach the fixed point (**kaplan:1959**). This means that after a small perturbation the system will quickly return to the steady state. An unstable steady state is one which if the system is perturbed slightly then it moves away from the steady state (**konopka:2007**).

2.4 The genetic toggle switch

One of the most common devices used in synthetic biology is the genetic toggle switch. A toggle switch consists of a set of transcription factors that mutually repress each other (Gardner, Cantor, & Collins 2000). Genetic switches play a major role in binary cell fate decisions like stem cell differentiation, as they are capable of exhibiting bistable behaviour. Bistability of a system is defined by the existence of two distinct phenotypic states but no intermediate state. Bistability is a property that is important in nature and a valuable resource to tap into in synthetic biology. It allows cells to alter their response to environmental cues and increases the overall population fitness by 'hedge-betting' the response of the population (XXX).

2.4.1 Importance in natural systems

In developmental processes, bistability ensures that the differentiating cell will follow one pathway, or the other, with no possible intermediate phenotypes. This is vital for the correct development of a cell in a specific pathway. One example is the trophectoderm differentiation pathway, in which a mutually inhibitory toggle switch exists between Oct3/4 and Cdx2. This determines whether an Embryonic Stem cell will differentiate into a Trophectoderm cell, if Cdx2 dominates the system, or an Inner Cell Mass cell if Oct3/4 dominates (Niwa et al. 2005). Bistability is critical in this system as a cell must differentiate into either a trophectoderm cell or an inner cell mass cell, thus the signal to do so must be straightforward. In the case of the GATA1 and PU.1 toggle switch, the transcription factor pair controls the fate of the common myeloid progenitors, and the two possible differentiation paths are erythroid and myeloid blood cells (Chickarmane, Enver, & Peterson 2009). The double-negative feedback loop created by the mutually repressive pair of transcription factors sustains the system in balance until an external stimulus causes one of the two transcription factors to increase in concentration. The increased concentration of one transcription factor causes the increased repression of the production of the antagonistic transcription factor, tipping the balance towards the dominance of the first transcription factor. The double negative feedback loop reinforces this dynamic and the system remains in the same state, until an external stimulus disturbs it (Ferrell 2002).

2.4.2 Uses in synthetic biology

Despite their simplicity, toggle switches can be powerful building blocks with which to create complex responses in a synthetic network. They can be used in isolation or in tandem to create complex networks and signalling cascades. The toggle switch has been used for the regulation of mammalian gene expression (**Deans:2007cya**; Kramer et al. 2004). Other synthetic applications of the toggle switch include the construction of a synthetic genetic clock (**Atkinson:2003tu**), of a predictable genetic timer (Ellis, Wang, & Collins 2009), and the formation of biofilms in response to engineered stimuli (Kobayashi et al. 2004). These applications are modifications of the classical toggle switch (Gardner, Cantor, & Collins 2000), and to our knowledge no application made of a cascade or collection of the switch has been successful. This would make more complex applications possible and could be used to solve real-life problems. For example, an analog-to-digital converter to translate external stimuli like the concentration of an inducer into an internal digital response, or programmable bacteria to move from point to point up different chemical gradients (Lu, Khalil, & Collins 2009). For a review on current circuits see (Khalil & Collins 2010) and for possible future applications see (Lu, Khalil, & Collins 2009). This leap will be difficult to achieve before first being able to build robust and well characterised individual switches.

2.4.3 Modelling the genetic toggle switch

The toggle switch motif has been studied extensively and there are numerous studies based on a number of different methods of modelling and analysis of the dynamics, including both deterministic and stochastic approaches. Deterministic modelling utilises ordinary differential equations (ODE) and models the concentrations of the species (proteins or other molecules) by time-dependent variables (de Jong 2002). When modelling deterministically the model is viewed as a system whose behaviour is entirely predictable, given sufficient knowledge. In stochastic modelling, species are measured in discrete amounts rather than concentrations and a joint probability distribution is used to express the probability that at time t the cell contains a number of molecules of each species (**Wilkinson:2006**; de Jong 2002). It takes uncertainty into account and is thus often more appropriate for modelling cellular systems, although more computationally expensive. In stochastic systems the Gillespie algorithm is widely used to simulate the time-evolution of the state of the system (Warren & ten Wolde 2005).

The conclusions drawn about the stability and robustness of the toggle switch also vary between the different modelling approaches. Numerous studies have concluded that cooperativity is a necessary condition for bistability to arise (Gardner, Cantor, & Collins 2000; Walczak, Onuchic, & Wolynes 2005; Warren & ten Wolde 2004; Warren & ten Wolde 2005; Cherry & Adler 2000). However, Lipshtat et al. (2006) found that stochastic effects can give rise to bistability even without cooperativity in three kinds of switch; the exclusive switch, in which there can only be one repressor bound at any one time, a switch in which there is degradation of bound repressors, and the switch in which free repressor proteins can form a complex, which renders them inactive as transcription factors (Lipshtat et al. 2006). In another study, Ma et al. (2012) found that the stochastic fluctuations in a system involving such a small number of molecules, like the toggle switch, uncovers effects that can not be predicted by the fully deterministic case (Ma et al. 2012). In their system, the toggle switch was found to be tristable, as small number effects render the third unstable steady state stable. Biancalani & Assaf (2015) identified multiplicative noise as the source of bistability in the stochastic case (Biancalani & Assaf 2015). Warren & ten Wolde (2005) concluded that the exclusive switch is always more robust than the general switch, since the free energy barrier is higher (Warren & ten Wolde 2005). A summary of the toggle switch models is shown in Table 2.1. As is clear from above, there is yet to exist a consensus on the stability a switch is capable of, and the most appropriate method of modelling it. Different methods arrive at different conclusions, creating confusion on which behaviour to be expected by the experimentalist for even a simple system like the toggle switch, consisting of just two genes. The toggle switch cannot be used as a building block of larger, more complex systems until its behaviour can be predicted accurately. Until then, designing systems with predictable behaviour will be near impossible.

Table 2.1 Summary of stability for the CS and DP switches found via different modelling approaches

	CS		DP	
Stability	Deterministic	Stochastic	Deterministic	
Monostable	(Loinger & Biham 2009)	(Loinger & Biham 2009)	(Guantes & Poyatos 2008)	
	(Gardner, Cantor, & Collins 2000)	(Lu, Onuchic, & Ben-Jacob 2014),		
Bistable	(Loinger & Biham 2009)	(Lipshtat et al. 2006),	(Guantes & Poyatos 2008)	
		(Biancalani & Assaf 2015),		
		(Loinger & Biham 2009)		
Tristable		(Loinger & Biham 2009),	(Guantes & Poyatos 2008),	
		(Ma et al. 2012)	(Lu, Onuchic, & Ben-Jacob 2014)	
Quadristable			(Guantes & Poyatos 2008)	

2.5 Introduction to Bayesian statistics

$$p(\theta|x) = \frac{p(x|\theta)p(\theta)}{\int p(x|\theta)p(\theta)d\theta \frac{p(x|\theta)p(\theta)}{p(x)}}$$

because

$$p(x)p(\theta|x) = p(\theta)p(x|\theta)$$

where $p(x|\theta)$ is the likelihood, $p(\theta)$ is the prior, and $\int p(x|\theta)p(\theta)d\theta$ is the evidence. This is the normalisation.

Bayes factor:

$$B_{12} = \frac{\int p(x|\theta, M_1)p(\theta, M_1)d\theta}{\int p(x|\theta, M_2)p(\theta, M_2)d\theta}$$

In our case, O is the objective, and D is the design. Therefore:

$$p(O|D_1) = \int p(O|\theta, D_1)p(\theta|D_1)d\theta,$$

This is the robustness, or evidence or marginal likelihood

$$p(O|D_1) = \int p(O|\theta, D_1)p(\theta|D_1)d\theta,$$
$$p(O|D_1) = \iiint_{\underline{\Theta}} p(O|\underline{\Theta})p(\underline{\Theta}|D_1)d\underline{\Theta}$$

where $\underline{\Theta} = \{\theta_1, \theta_2, \theta_3\}$

Assuming the prior is uniform, and $a = 0$:

$$p(O|D_1) = \iiint_{\underline{\Theta}} p(O|\underline{\Theta})\frac{1}{b_1}\frac{1}{b_2}\frac{1}{b_3}d\underline{\Theta}$$
$$p(O|D_1) = \frac{1}{b_1}\frac{1}{b_2}\frac{1}{b_3}\iiint_{\underline{\Theta}} p(O|\underline{\Theta})d\underline{\Theta}$$

Assuming uniform likelihood:

$$p(O|D_1) = \frac{1}{b_1}\frac{1}{b_2}\frac{1}{b_3}\iiint_{\underline{\Theta}_F} 1d\theta_1\theta_2\theta_3 + \frac{1}{b_1}\frac{1}{b_2}\frac{1}{b_3}\iiint_{\underline{\Theta}_F} Od\underline{\Theta}$$

2.5.1 Bayes' theorem

2.5.2 Bayesian inference

2.5.3 Model checking

2.5.4 Prior selection

2.5.5 Model parametric Robustness

During this thesis I define robustness as the ability of a system to retain its function despite parameter perturbations (Stelling et al. 2004). The robustness of biological systems has been studied extensively (Barkai & Leibler 1997; Stelling et al. 2004; Prill, Iglesias, & Levchenko 2005; Kim et al. 2006; Kitano 2007; Hafner et al. 2009; Shinar & Feinberg 2010; Zamora-Sillero et al. 2011; Woods et al. 2016). and it is well known that feedback loops can increase the robustness of a system (Becskei & Serrano 2000; Doyle & Csete 2005).

The robustness of a model can be calculated by dividing the volume of its functional region by the volume of its priors. This is a measure of the volume of the posterior distribution is compared to the priors. It comes from Bayes' rule that:

$$f(\theta|x) = \frac{f(\theta)f(x|\theta)}{\int p(x|\theta)p(\theta)d\theta} \tag{2.1}$$

where $p(x|\theta)$ is the likelihood, $p(\theta)$ is the prior, and $\int p(x|\theta)p(\theta)d\theta$ is the evidence. The evidence is the normalisation added so that the distribution integrates to 1. For a given model design D and objective O we define the functional region F as the region within the prior where O is satisfied. So within the prior we can assign 1 to any region that falls within F and 0 to any region outside that.

$$p(O|D_1) = \int p(O|\theta, D_1)p(\theta|D_1)d\theta, \tag{2.2}$$

For a design with three parameters this becomes:

$$p(O|D_1) = \iiint_{\underline{\Theta}} p(O|\underline{\Theta})p(\underline{\Theta}|D_1)d\underline{\Theta}, \tag{2.3}$$

where $\underline{\Theta}$ is a vector containing the three parameters = $\theta_1, \theta_2, \theta_3$. To calculate the robustness, or model evidence, we integrate this with respect to $\underline{\Theta}$. We assume all

parameters $\theta_1, \theta_2, \theta_3$ are uniform, $p(\underline{\Theta}|D_1) \sim U(a, b)$. If we assume a = 0 this integral becomes:

$$p(O|D_1) = \iiint_{\underline{\Theta}} p(O|\underline{\Theta}) \frac{1}{b_1}\frac{1}{b_2}\frac{1}{b_3} d\underline{\Theta} \text{ , and} \tag{2.4}$$

$$p(O|D_1) = \frac{1}{b_1}\frac{1}{b_2}\frac{1}{b_3} \iiint_{\underline{\Theta}} p(O|\underline{\Theta}) d\underline{\Theta} \tag{2.5}$$

since $\frac{1}{b_1}\frac{1}{b_2}\frac{1}{b_3}$ is a constant. Then assuming that the likelihood is uniform Equation2.5 becomes:

$$p(O|D_1) = \frac{1}{b_1}\frac{1}{b_2}\frac{1}{b_3}\left[\iiint_{\underline{\Theta}_F} 1 d\underline{\Theta} + \iiint_{\underline{\Theta}_{\bar{F}}} \overset{0}{0 d\underline{\Theta}} \right] \tag{2.6}$$

$$\tag{2.7}$$

since we assign 1 to any region within F and 0 to any region outside it. This becomes:

$$p(O|D_1) = \overbrace{\frac{1}{b_1}\frac{1}{b_2}\frac{1}{b_3}}^{|P|} \underbrace{\iiint_{\underline{\Theta}_F} 1 d\underline{\Theta}}_{|F|}, \tag{2.8}$$

$$\therefore p(O|D_1) = \frac{|F|}{|P|}, \tag{2.9}$$

where |P| is the volume of the prior P and |F| the volume of the functional region F. Therefore, in the case where both the prior and the likelihood are uniform, the robustness R of the design is the ratio of the volumes of the two.

If on the other hand we assume the likelihood is multivariate normal, with priors remaining uniform, Equation2.5 becomes:

$$p(O|D_1) = \frac{1}{|P|} \iiint_{\underline{\Theta}} f(\underline{\Theta}; \mu, \Sigma) d\underline{\Theta} \tag{2.10}$$

$$\therefore p(O|D_1) = \frac{1}{|P|} \underbrace{\times (2\pi)^{\frac{k}{2}} \times |\Sigma|^{\frac{1}{2}}}_{|F|} \tag{2.11}$$

$$\therefore p(O|D_1) = \frac{|F|}{|P|}, \tag{2.12}$$

We can use the Bayes' factor in order to compare the robustness between two model designs. The Bayes' factor is defined as follows:

$$B_{ab} = \frac{\int p(x|\theta, D_a)p(\theta, D_a)d\theta}{\int p(x|\theta, D_b)p(\theta, D_b)d\theta} \tag{2.13}$$

$$\therefore B_{ab} = \frac{|Fa|}{|Pa|} / \frac{|Fb|}{|Pb|} \tag{2.14}$$

Therefore, we can use the ratio of the two robustness measures to calculate the Bayes' factor. If two models have a different number of parameters, the robustness of the system will only increase if |F| increases by more than the proportion by which |P| increased (Woods et al. 2016). A model will be penalised for an additional if it does not increase the volume of the functional region by more than the volume that the added parameter added to the prior. This is true for nested models, where one model is wholly contained in the other.

2.6 Approximate Bayesian Computation (ABC)

2.6.1 ABC algorithms

Stability Finder is based on a statistical inference method which combines ABC with Sequential Monte Carlo (SMC) (Toni et al. 2009). This simulation-based method uses an iterative process to arrive at a distribution of parameter values that can give rise to observed data or a desired system behaviour (Barnes et al. 2011).

ABC methods are used for inferring the posterior distribution in cases where it is too computationally expensive to evaluate the likelihood function. Instead of calculating the likelihood, ABC methods simulate the data and then compare the simulated and observed data through a distance function (Toni et al. 2009). Given the prior distribution $\pi(\theta)$ we can approximate the posterior distribution, $\pi(\theta \mid x) \propto f(x \mid \theta)\pi(\theta)$, where $f(x \mid \theta)$ is the likelihood of a parameter, θ, given the data, x. There are a number of different variations of the ABC algorithm depending on how the the approximate posterior distribution is sampled.

The simplest ABC algorithm is the ABC rejection sampler (Pritchard et al. 1999). In this method, parameters are sampled from the prior and data simulated through the data generating model. For each simulated data set, a distance from that of the desired behaviour is calculated, and if greater than a threshold, ε, the sample is rejected, otherwise it is accepted.

Algorithm 1 ABC rejection algorithm

1: Sample a parameter vector θ from prior $\pi(\theta)$
2: Simulate the model given θ
3: Compare the simulated data with the desired data, using a distance function d and tolerance ε. if $d \leq \varepsilon$, accept θ

The main disadvantage of this method is that if the prior distribution is very different from the posterior, the acceptance rate is very low (Toni et al. 2009). An alternative method is the ABC Markov Chain Monte Carlo (MCMC) developed by Marjoram et al. (2003). The disadvantage of this method is that if it gets stuck in an area of low probability it can be very slow to converge (**Sisson:wf**).

The method used here is based on Sequential Monte Carlo, which avoids both issues faced by the rejection and MCMC methods. It propagates the prior through a series of intermediate distributions in order to arrive at an approximation of the posterior. The tolerance, ε, for the distance of the simulated data to the desired data is made smaller at each iteration. When ε is sufficiently small, the result will approximate the posterior distribution (Toni et al. 2009).

ABC SMC can identify the parameter values within a predefined range of values that can achieve the desired behaviour. It works by first sampling at random from the initial range set by the user, i.e. form the prior distribution of values. Each sample from the priors is called a particle. It then simulates the model given those values and compares that to the target behaviour. If the distance between the simulation and the target behaviour is greater than a predefined threshold distance ε, then the parameter values that produced that simulation are rejected. This is repeated for a predefined number of samples which are collectively referred to as a population. Each particle in a population has a weight associated with it, which represents the probability of it producing the desired behaviour. At subsequent iterations the new samples are obtained from the previous populations and the ε is set to smaller value, thus eventually reaching the desired behaviour. The algorithm proceeds as follows:

Algorithm 2 ABC SMC algorithm

1: Select ε and set population t = 0
2: Sample particles (θ). If t = 0, sample from prior distributions (P). If t > 0, sample particles from previous population.
3: If t > 0: Perturb each particle by ± half the range of the previous population (j) to obtain new perturbed population (i).
4: Simulate each particle to obtain time course.
5: Reject particles if d > ε.
6: Calculate the weight for each accepted particle. At the first population assign a weight equal to 1 for all particles. In subsequent populations the weight of a particle is equal to the probability of observing that particle divided by the sum of the probabilities of the particle arising from each of the particles in the previous population:
7: $w_t^{(i)} = \begin{cases} 1, & \text{if } n = 0 \\ \frac{P(\theta_t^{(i)})}{\sum_{j=1}^{N} w_{t-1}^{(j)} K_t(\theta_{t-1}^{(j)}, \theta_t^{(i)})}, & \text{if } n>0. \end{cases}$

2.6.2 Particle sampling

For the first population, particles are sampled from the priors. Random samples are taken from the distribution specified by the user for each parameter.

For subsequent populations particles are sampled from the previous population. The weight of each particle in the previous population dictates the probability of it being sampled. The number of samples to be drawn is specified by the user in the input file.

2.6.3 Perturbation

Each sampled particle is perturbed by a kernel defined by the distribution of the previous population, as developed by Toni et al. (2009).

$$K_p(\theta|\theta*) = \theta * + U(+s_p, -s_p), \text{ where:} \tag{2.15}$$

$$s_p = \frac{1}{2}\big(max(\theta_{p-1}) - min(\theta_{p-1})\big) \tag{2.16}$$

If the θ* falls out of the limits of the priors then the perturbation is rejected and repeated until an acceptable θ* is obtained. This method is successful in perturbing the particles by a small amount in order to explore the parameter space, but can be slow to complete.

2.6.4 Particle simulation

Each particle is simulated using cuda-sim (Zhou et al. 2011). The model is provided by the user in SBML format and is converted into CUDA® code by cuda-sim. The model in CUDA® code format can then be run on NVIDIA®. CUDA®. GPUs. This allows the user to take advantage of the speed of parallelised simulations without any CUDA® knowledge.

2.6.5 Weight calculation

For the first population the weights are all given a value of 1, and then normalised over the number of particles. For subsequent populations the weights of the particles are calculated by considering the weights of the previous population (Toni et al. 2009). The weights are then normalised over the total number of particles.

$$w_t^{(i)} = \frac{P(\theta_t^{(i)})}{\sum_{j=1}^{N} w_{t-1}^{(j)} K_t(\theta_{t-1}^{(j)}, \theta_t^{(i)})} \text{ for n} > 0 \tag{2.17}$$

This algorithm is implemented on a simple example for illustration. A simple model was used, consisting of one species, A converting to another, B. The model is described by two differential equations, where A is the reactant and B the product, produced at a rate p.

$$\frac{d[B]}{dt} = p[A] \tag{2.18}$$

$$\frac{d[A]}{dt} = -p[A] \tag{2.19}$$

The priors were set to $p \sim U(0, 10)$. Initial conditions for A and B were set to 1 and 0 respectively. The data to which the model was compared to was generated by simulating the same model with the parameter set to 1, as shown in Figure 2.1.

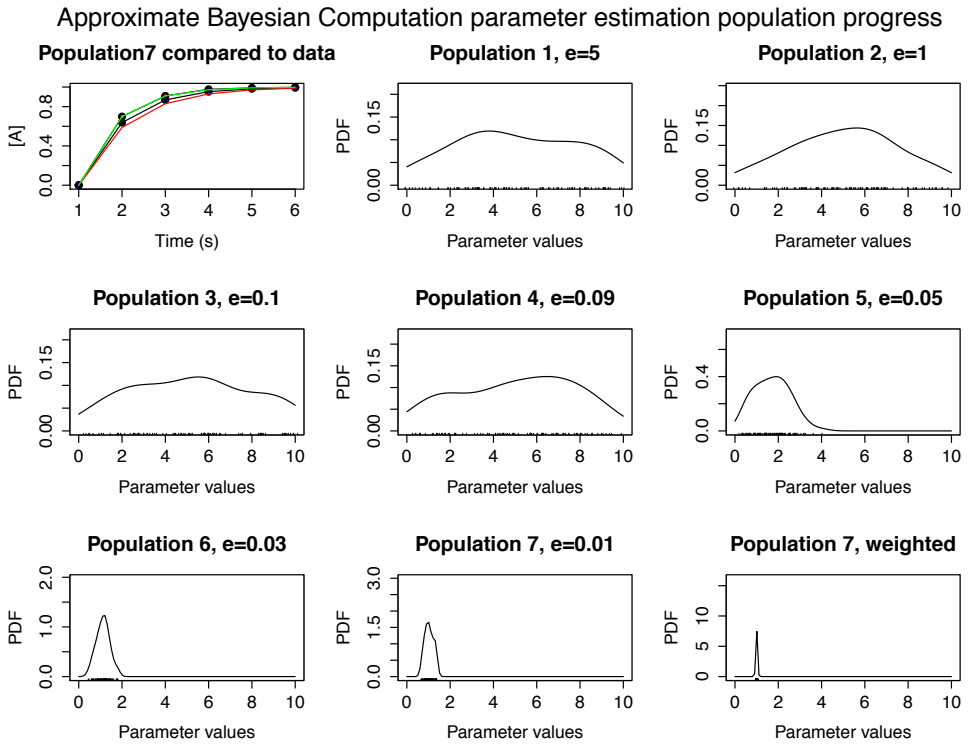

Figure 2.1 ABC SMC parameter inference. The posterior parameter is equal to 1 and its time course shown in red in the top left panel. The blue time course is that of the final population, green is the upper quartile and red is the lower quartile range of values. The progress of the selection process can be seen the ε schedule proceeds from the top left to the bottom right. The bottom far right panel is a density plot of ε= 0.01 with their weights taken into account.

Figure 2.1 demonstrates, using a simple example, that ABC SMC is capable of fitting a model to the data. During the course of 7 populations, the accepted distance ε of the simulated particles to the data is incrementally decreased. This leads to a final population where the distance of the data to the particles is very small, and there is a good agreement between the two. The algorithm concludes with a set of parameter values that produced this behaviour, which approximate the posterior distribution. The posterior distribution found in this model is in good agreement with the parameter value used to generate the data. This example successfully demonstrates the effectiveness of the ABC SMC algorithm in fitting models to data.

2.7 Flow Cytometry

Flow cytometry detects the fluorescent intensity levels in individual cells. It can also provide physical information about the size and granularity of a cell via the forward and side scattering respectively. An overview of flow cytometry is shown in Figure 2.2. A laser excites the fluorochrome present in the bacterial cells. The fluorochromes emit a signal that is detected by channels in the optics. The signals are then all collected and analysed. A sample typically consists single cell measurements of 10^4-10^5 cells. Flow cytometry is a powerful tool for synthetic biology as it can measure multiple parameters in single cells, and process up to 35,000 cells sec^{-1} (*Attune NxT Acoustic Focusing Cytometer* 2015).

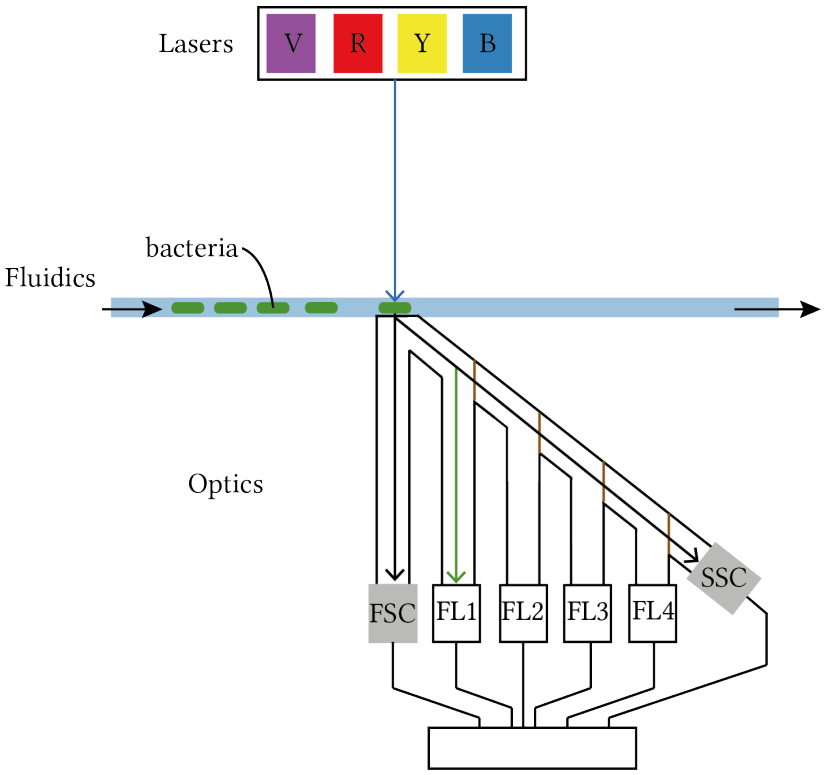

Figure 2.2 : Flow cytometry. A laser excites the fluorescent proteins present in each cell. The cytometer has up to 4 lasers, violet (V), red (R), yellow (Y) and blue (B). The detectors in the optics, FL1-4 pick up the signals. The cytometer also picks up size and granularity information via the forward scatter (FSC) and side scatter (SSC) detectors.

3 Postive feedback loops can increase the robustness of a genetic toggle switch

3.1 Introduction

In this chapter I examine whether adding feedback loops to the genetic toggle switch increases its parametric robustness to parameter fluctuations. To do this, I use ABC SMC to estimate the parameter values that allow the toggle switch model to behave like a switch. I then study the effect that adding feedback loops has to the toggle switch bistability, and finally I use model selection to select the most robust switch model out of the ones considered.

Structurally this chapter is organised as follows: In the first section I examine the genetic toggle switch with no added feedback loops. I use a parameter scan to find the parameter values that make it bistable and then use ABC SMC for parameter inference for a switch-like behaviour. In the subsequent section I examine the effect that the addition of feedback loops to the genetic toggle switch has on its stability, and select the switch architectures that are capable of bistable behaviour. Finally, I use ABC-SysBio model selection to select the most robust model out of the bistable switches.

3.2 Background

3.2.1 The bistable genetic toggle switch

Gardner, Cantor, & Collins (2000) constructed the first synthetic genetic toggle switch. Their model consisted of two mutually repressing transcription factors, and

is defined by the following ODEs:

$$\frac{du}{dt} = \frac{a_1}{1 + v^{\beta}} - u \tag{3.1}$$

$$\frac{dv}{dt} = \frac{a_2}{1 + u^{\gamma}} - v, \tag{3.2}$$

where u is the concentration of repressor 1, v the concentration of repressor 2, a_1 and a_2 denote the effective rates of synthesis of repressors 1 and 2 respectively, β is the cooperativity of repression of promoter 1 and γ of repressor 2. This model is capable of bistable behaviour when a_1 and a_2 are balanced and when β, γ are > 1 (Gardner, Cantor, & Collins 2000).

3.2.2 Phase space and bifurcation analysis

First, I study the model given in Equations 3.1- 3.2 by conducting a bifurcation analysis in order to confirm that it is capable of bistable behaviour. A bifurcation analysis is used to determine the properties of a system in parameter space (Alon 2007). Here I used the PyDSTool (Clewley 2012), a python package used for the analysis of dynamical systems.

The parameters chosen here for the bifurcation analysis are within the range suggested by Gardner, Cantor, & Collins (2000). a_1 and a_2 are set to 10, and β, γ set to 2. A vector plot shows that the system has two steady states as shown in Figure 3.1B. Both states were found to be stable by examining the eigenvalues of the system at each steady state using Mathematica (Mathematica 2016).

I further study the system by conducting a bifurcation analysis, where all parameters remain constant to the values shown above, and only one parameter (a_1) is varied. The bifurcation analysis shows that by varying the parameter for the effective rate of synthesis of repressor 1 while all other parameters remain constant, the system is bistable when $5 \geq a_1 \leq 31$.

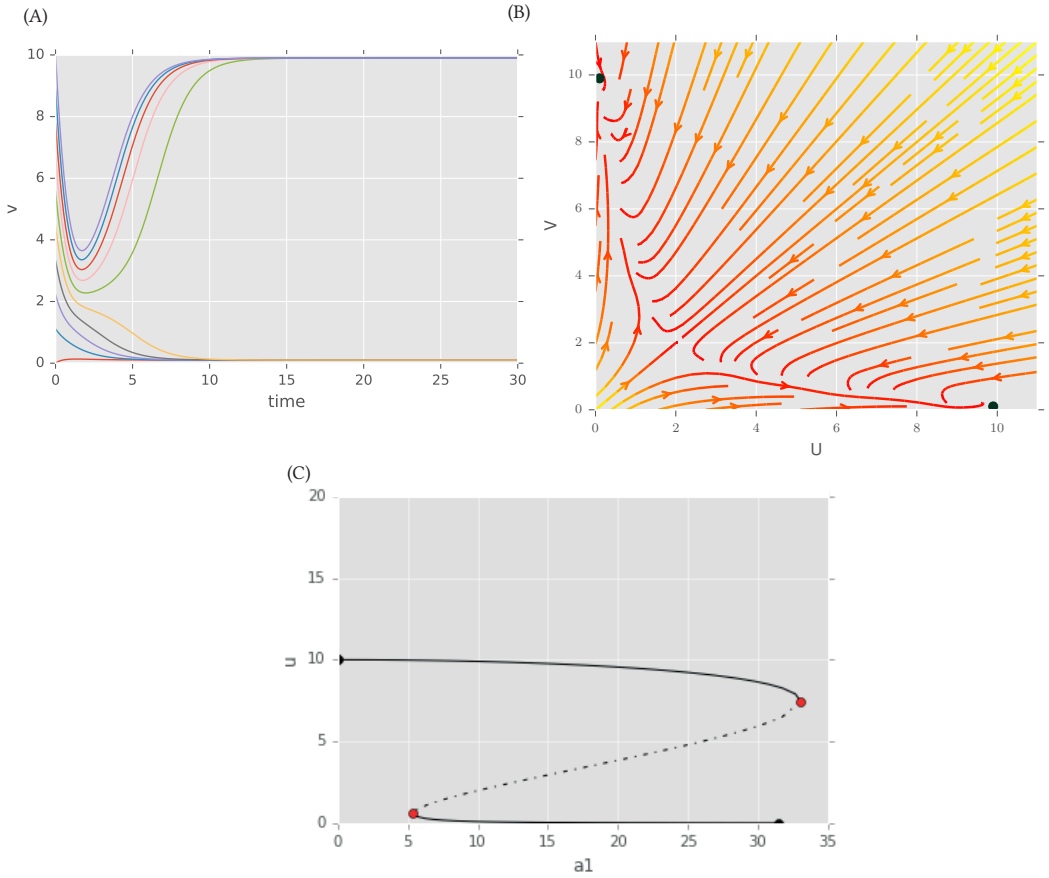

Figure 3.1 : The Gardner, Cantor, & Collins (2000) toggle switch is capable of bistable behaviour given a_1 and a_2 = 10, and β, γ = 2. (A) The timecourse of the simulated model using multiple initial conditions. (B) The vector plot of the Gardner switch shows there are two stable steady states. (C) A bifurcation diagram shows that the system is bistable when $5 \geq a_1 \leq 31$.

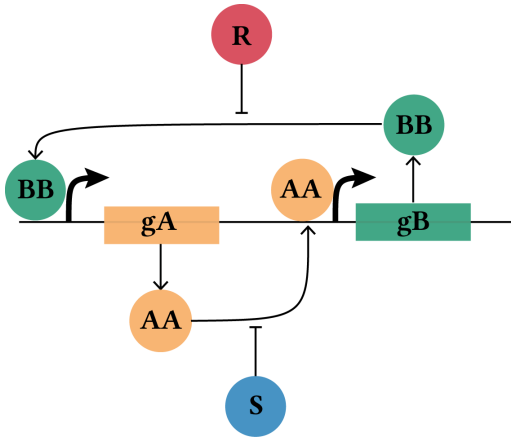

Figure 3.2 : An illustration of the model used in the parameter scan.

3.3 Designing a simple synthetic switch

The model used in Section 3.2.2 could be solved analytically and I demonstrated that it is bistable for the parameters given. Nevertheless, for a switch to be useful in synthetic biological applications, it must be capable of behaving like a switch over a range of parameter values, as these fluctuate within the cellular environment. Therefore, in this section I study the parameter ranges that can give rise to a bistable switch. This indicates whether the bistability of the switch models is robust to small parameter fluctuations.

In order to study the switch system in a more realistic way, I developed an extension to the Gardner, Cantor, & Collins (2000) switch. This new set of switches does not use the quasi-steady state approximation (QSSA) that is often used in modelling the toggle switch. Using mass action, this changes the two-equation system used in Equations 3.1 into a system of 18 equations. The equations describing the system are shown below and illustrated in Figure 3.2. The ODEs are given in Appendix(XXX). The system consists of two genes, gA and gB. The products of the genes homodimerise and mutually repress each other. A symmetric model, where the parameters for equivalent reactions are set to be the same, was used for simplicity.

3.3.1 Parameter scan for model stability

In order to determine the range of parameter values for which the above model is bistable, I developed a parameter scanning algorithm. The algorithm is outlined in Algorithm 3 below. This method involves the scan of parameter values as well as

Table 3.1 Simple mass action switch

Equation	Description
$gA \xrightarrow{ge} gA + A$	gene expression
$gB \xrightarrow{ge} gB + B$	
$A + A \xrightarrow{dim} A2$	dimerization
$B + B \xrightarrow{dim} B2$	
$A2 \xrightarrow{dim\_r} A + A$	monomerization
$B2 \xrightarrow{dim\_r} B + B$	
$gA + B2 \xrightarrow{rep} B2gA$	repression
$gB + A2 \xrightarrow{rep} A2gB$	
$B2gA \xrightarrow{rep\_r} B + gA$	dissociation
$A2gB \xrightarrow{rep\_r} A2 + gB$	
$A \xrightarrow{deg} \emptyset$	degradation
$B \xrightarrow{deg} \emptyset$	

initial conditions for AA and BB. Parameter values are sampled randomly from a uniform distribution. Here, each set of samples will be referred to as a particle. For each particle, latin hypercube sampling is used to sample initial conditions (McKay, Beckman, & Conover 2000). The uniform priors of the two species in consideration represent a rectangle space, which is subdivided into equal parts. Then a random sample is drawn from each sub-part, as illustrated in Figure 3.3. This is used to ensure that the whole space is sampled uniformly. Latin hypercube sampling is done in two dimensions, in order to sample initial conditions for the two dimers, AA and BB.

Algorithm 3 Parameter scan algorithm

1: Select multiple sets of parameter values from a random uniform distribution between 0 and 10.
2: **for** each set of parameter values **do**:
3: Select multiple sets of initial conditions for the two dimers via latin hypercube sampling
4: **for** each set of initial conditions **do**
5: Integrate ODEs of the model
6: Solve system to steady state
7: **end for**
8: **end for**

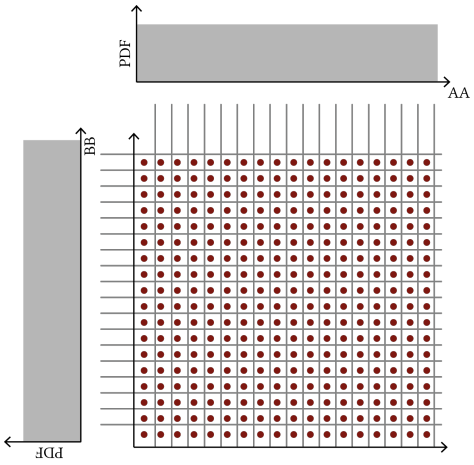

Figure 3.3 Latin hypercube sampling ensures that the whole space is sampled evenly. For the two species concerned, A and B, we assume uniform distributions, shown in grey. The joint space of the two distributions is divided into smaller equal parts and a random sample is drawn from within each subspace.

After each particle was solved to steady state, the value of each dimer at the last timepoint was taken, for all initial conditions sampled. These were used to make a phase diagram for each particle, which consists of the steady state value of one dimer plotted against the other. The parameter scan uncovered the presence of tristable, bistable and monostable systems given a different set of parameter values. An example of the phase plots for each of the three types of stabilities found during the scan are shown in Figure 3.4.

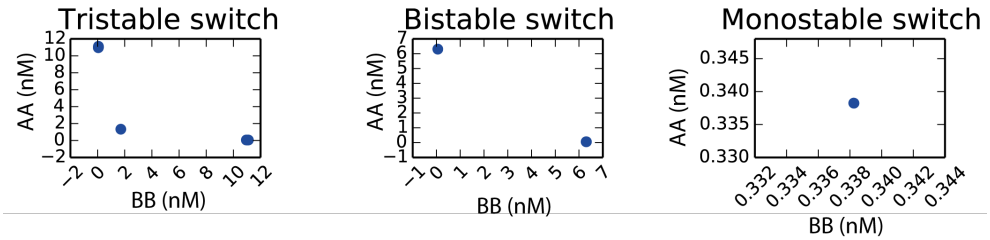

Figure 3.4 : An example for each type of switch found during the parameter scan. Each graph represents the steady state values of one dimer plotted against the other, from one parameter set and 100 initial conditions.

There were a total of 217 monostable, 16 bistable and 6 tristable switches out of 400 sampled parameter sets. The remaining samples did not reach steady state. The parameter values that produced each stability are shown in Figure 3.5.

From the histograms in Figure 3.5 it can be seen that the parameter for gene expression (ge) tends to be relatively high when bistability arises whereas the parameter for the reverse reaction of repression (rep_r) tends to be low. Degradation (deg) also tends to be low. From this analysis I showed that the toggle switch is capable of bistable behaviour for a range of parameter values.

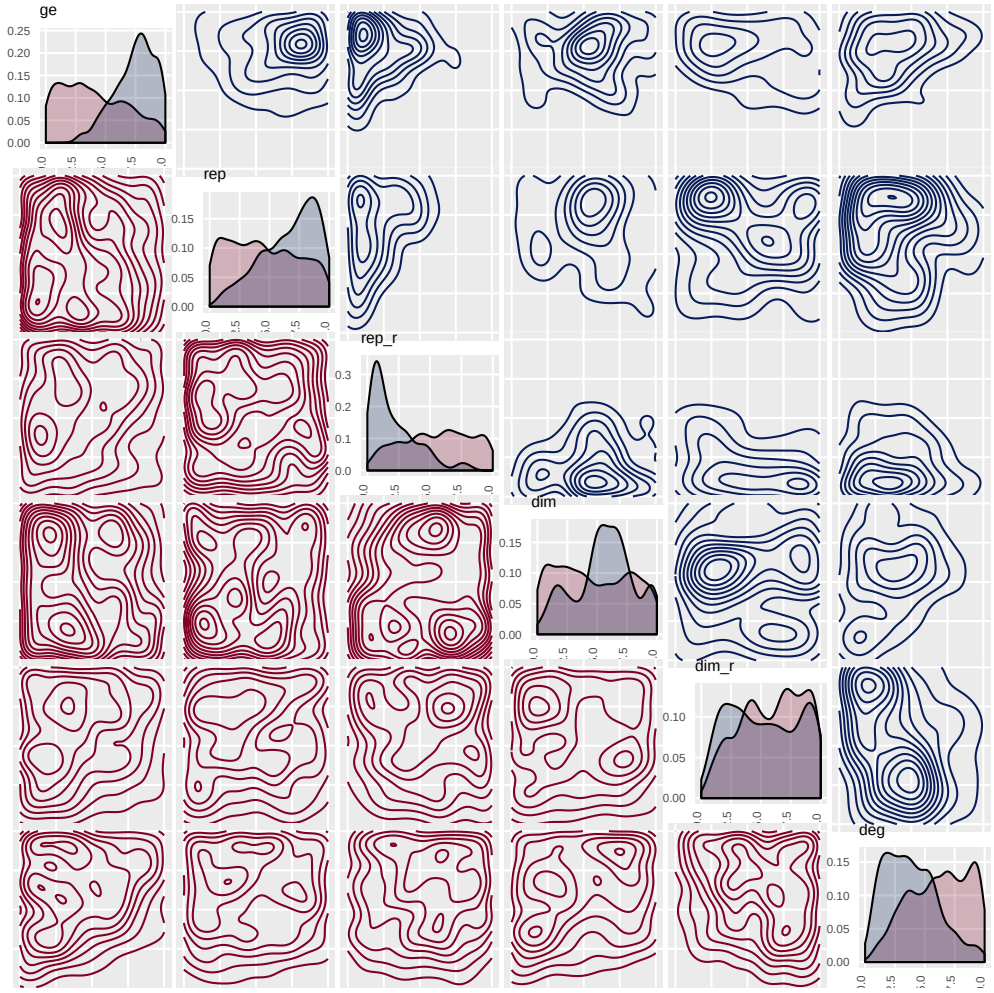

Figure 3.5 The distribution of parameter values that resulted in monostable, bistable and tristable switches in the parameter scan. Each graph represents the distribution of the values of one parameter.

3.3.2 Toggle switch parameter inference

In this section, I extend the analysis carried out in Section 3.3.1 to study the problem of how to rationally design a synthetic biological system to perform a behaviour of choice. In order to address this question I use a Bayesian approach, known as Approximate Bayesian Computation and described in Section(XXX), implemented in a software package, ABC-SysBio (Liepe et al. 2010).

This approach is capable of approximating the posterior distribution that gives rise to the behaviour of choice (Toni et al. 2009). By simulating the model in question, this approach can identify an approximate posterior distribution via a series of intermediate distributions. This method can be used for the rational design of synthetic biological systems by defining some design objectives to which the model is fitted to (Barnes et al. 2011). By specifying the inputs to the system and the outputs required, the posterior of the model that can produce this behaviour can be identified.

Here I use ABC-SysBio to fit a model to the design objectives of a switch-like behaviour. I extend the model used in Section 3.3.1 by adding two inducers to the system, S and R. S removes the A homodimer (AA) from the system by binding to it thus removing the repression on gene B. R removes BB from the system with the same mechanism. These inducers represent the stimuli that will turn the switch ON and OFF in a biological setting. The following equations are added to the existing set of equations for the mass action switch:

The range of parameter values shown to produce a bistable switch in Section 3.3.1 were used as priors and are shown in Table 3.3.

Table 3.2 Toggle switch inducer equations

Equation	Description
$S + A2 \xrightarrow{rep\_dim} SA2$	Inducer repression
$R + B2 \xrightarrow{rep\_dim} RB2$	
$SA2 \xrightarrow{rep\_dim\_r} S + A2$	Inducer dissociation
$RB2 \xrightarrow{rep\_dim\_r} R + B2$	
$R \xrightarrow{deg} \emptyset$	Inducer degradation
$S \xrightarrow{deg} \emptyset$	

Table 3.3 The prior distributions used for the standard toggle switch parameter inference. The values indicate the lower and upper limits of a uniform distribution.

ge	rep	rep_r	dim	dim_r	deg	rep dim	rep_dim_r	deg_sr
6-9	4-10	1-4	4-10	2-7	2-5	0.05-0.1	0.01-0.05	0.01-0.05

3.3.3 Design specifications

The following were defined as the design specifications for a bistable genetic toggle switch. The two inducers, S and R, are used as inputs to flip the switch ON and OFF respectively. The required output is the switch flipping from the OFF state to the ON state and then to the OFF state again (Figure 3.6).

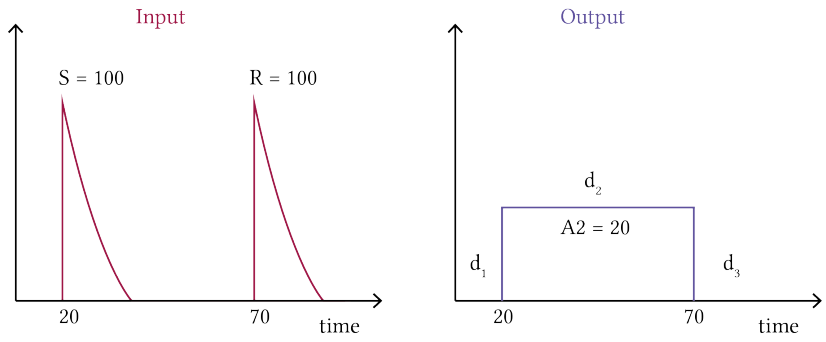

Figure 3.6 : Design specification for ABC parameter inference. The input to the system consists of an event turning on the stimulus (S) at t=20 and another turning on the repressor (R) at t=70. The output specification is the response to the switch to these stimuli.

3.3.3.1 Distance function

In order to fit the switch model to the behaviour specified above, a distance function must be defined. The distance function defines the quantity that is minimized at each successive iteration of ABC SMC. Three distances are measured, one for each state of the switch, OFF-ON-OFF. Each distance is the sum of the distances of the simulated protein levels to the desired protein levels at each timepoint. All three distances must reach the minimum threshold for the process to be complete.

$$d_1 = \sum_{i=0}^{20} (s_i - t_1)^2 \tag{3.3}$$

$$d_2 = \sum_{i=21}^{70} (s_i - t_2)^2 \tag{3.4}$$

$$d_3 = \sum_{i=71}^{100} (s_i - t_3)^2, \tag{3.5}$$

where i represents the timepoints, s_i the simulation result at each timepoint and t_1, t_2, t_3 represent each target behaviour. t_1, t_2, t_3 were set to 0, 20, 0 respectively.

3.3.4 Results

The results of the parameter inference of the toggle switch are shown in Figure 3.7. The model was shown to successfully behave like a switch within the parameter range used here. The resulting timecourse of the last population matches the design specifications. It can be seen from the posterior distribution that gene expression rate (ge) must be high relative to the prior. Repression (rep) and degradation must both be low and the rate of dimerisation (dim) must be high relative to the prior. The posterior constitutes the specifications that can be used when building a synthetic switch in the lab, as the appropriate components can be tweaked for a successful circuit. More generally, the posterior distribution shows that the parameter space that can give rise to this behaviour is limited. A very small portion of the prior is capable of producing the desired design specifications. In the next section I will examine wether the addition of feedback loops can increase the volume of the posterior that produces the required behaviour.

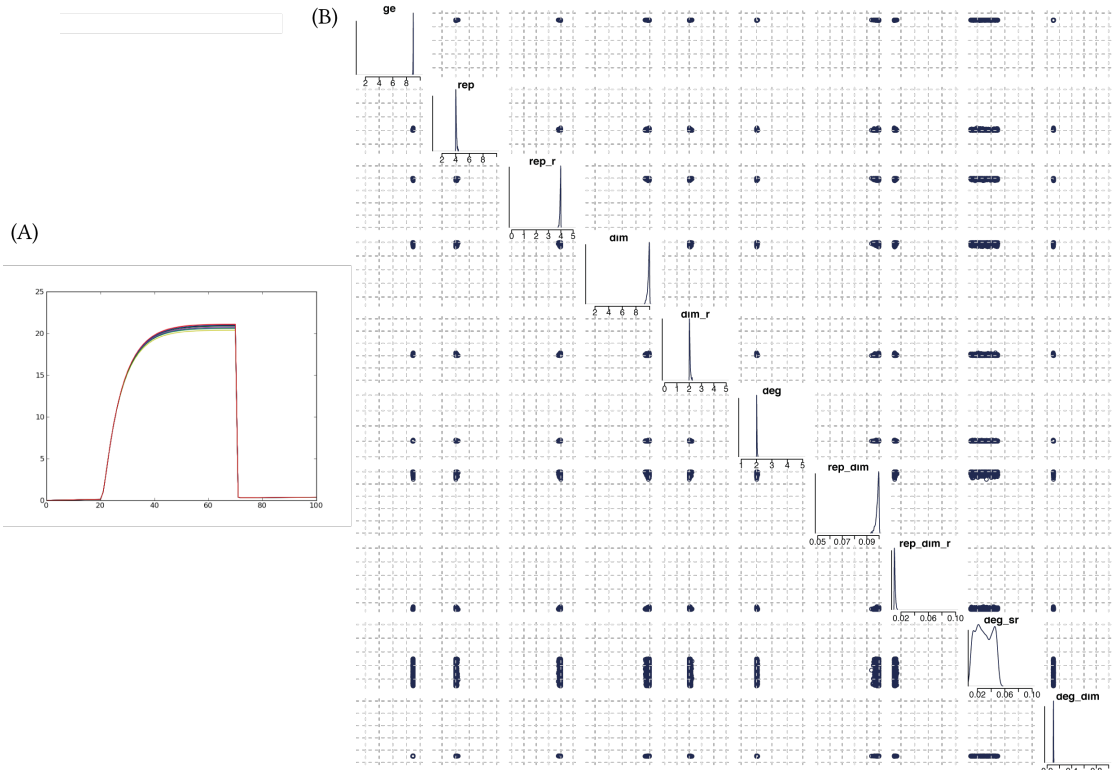

Figure 3.7 (A) The time series of the final population (for final $\epsilon = 2$) of the standard toggle switch ABC-SMC parameter inference. The stimulus, that represses A2, is added at t=20 and the repressor, that represses B2 is added at t=70. (B) The posterior distribution of the toggle switch.

3.4 Designing a more robust genetic toggle switch

In this section I examine whether the addition of feedback loops to the toggle switch can increase its robustness to parameter fluctuations. Here I define a robust system as a device that can withstand fluctuations in parameter values and still produce the desired behaviour (parametric robustness). Feedback loops are well known key regulatory motifs (Brandman et al. 2005).

As was shown in Section 3.3, the posterior distribution of the simple toggle switch was narrow, and thus the behaviour of choice will not be robust. Here I examine whether adding feedback loops to the genetic toggle switch can increase parametric robustness for the desired design specifications.

3.4.1 Models of the genetic toggle switch

Both positive and negative feedback loops were considered. Therefore 7 models were examined for their capability to behave like a switch. The simple toggle switch, switches with positive autoregulation in either or both nodes and switches with negative autoregulation in either or both nodes. The models considered are illustrated in Figure 3.8.

In order to study each model analytically I built extensions to the (Gardner, Cantor, & Collins 2000) toggle switch in order to incorporate positive and negative feedback to the system. The models for the double autoregulation models are shown below. For the single autoregulation models, the unnecessary autoregulation term is set to 0.

Double negative autoregulation:

$$\frac{du}{dt} = \frac{a_1}{1 + v^{\beta} + k_u^{\delta_u}} - u \tag{3.6}$$

$$\frac{dv}{dt} = \frac{a_2}{1 + u^{\gamma} + k_v^{\delta_v}} - v, \tag{3.7}$$

Double positive autoregulation:

$$\frac{du}{dt} = \frac{a_1 + k_u^{\delta_u}}{1 + v^{\beta}} - u \tag{3.8}$$

$$\frac{dv}{dt} = \frac{a_2 + k_v^{\delta_v}}{1 + u^{\gamma}} - v, \tag{3.9}$$

where k represents the effective binding of the transcription factor to its own promoter and γ represents the polymerisation of the bound transcription factor.

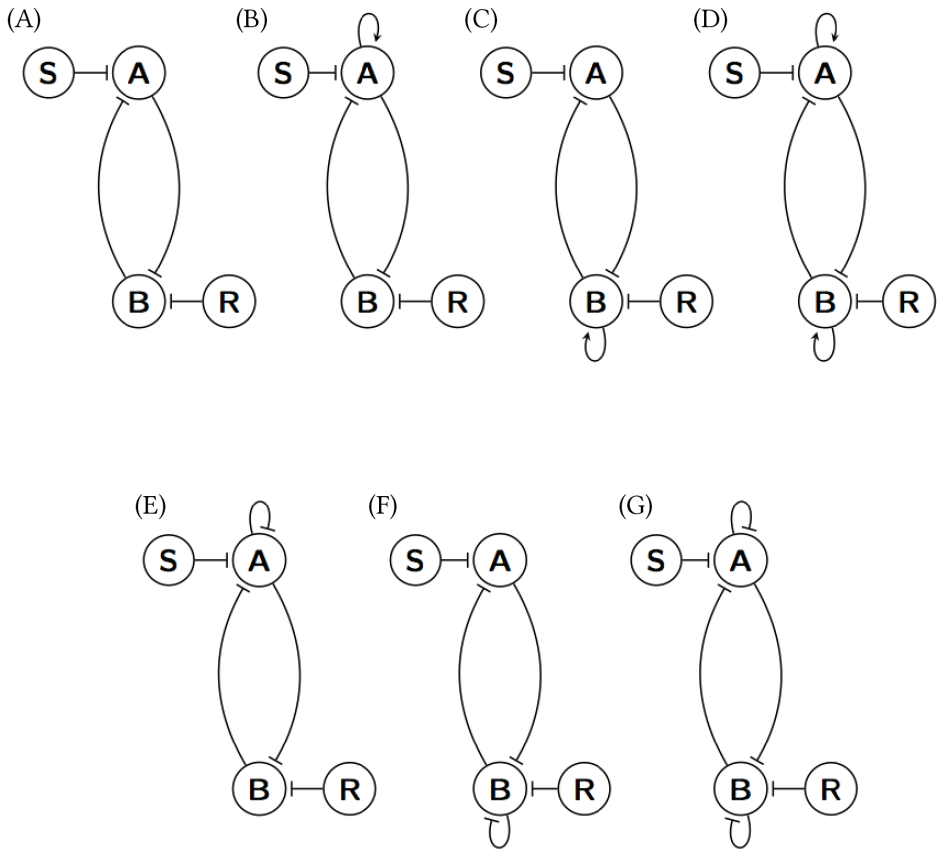

Figure 3.8 : Toggle switch designs considered for model selection. 7 models were used: The simple toggle switch (A), the switches with positive autoregulation on either or both nodes (B, C, D) and the switches with negative autoregulation on either or both nodes (E, F, G)

3.4.1.1 Autoregulation switches phase plots

In order to use these models for model selection, it must first be determined whether they are capable of behaving like a switch. ABC SMC model selection is used to select models that can produce the same behaviour, over a greater range of parameter values. If a model is not capable of producing the desired behaviour for the prior range, then it will not be used for model selection.

I used the PyDSTool (Clewley 2012) in order to determine whether each of the 7 switches is capable of bistable behaviour. The same analysis as Section 3.2.2 was used to identify the steady states of the systems and their stabilities. As shown in Figure 3.9, both single and double positive autoregulation is consistent with bistable behaviour. Two stable states were found for both cases when $k = 2$ and $\delta = 1$.

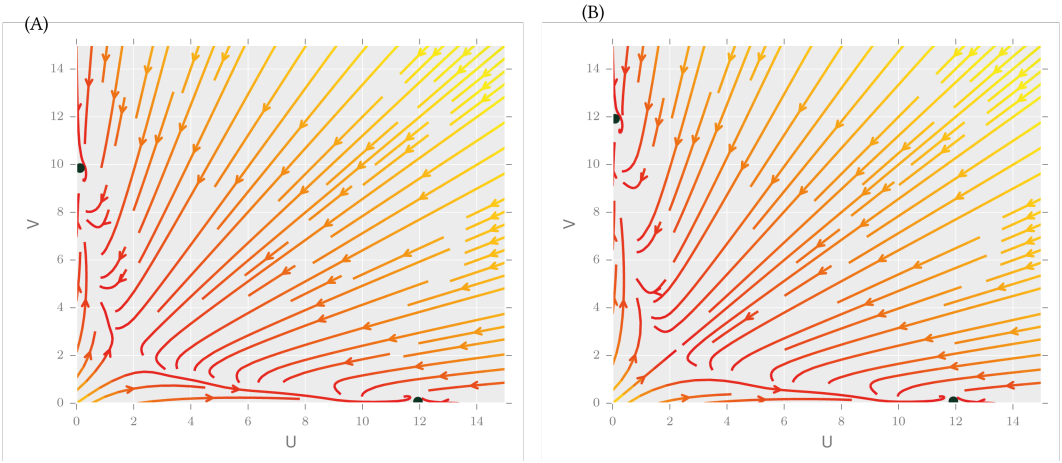

Figure 3.9 : Both single and positive autoregulation is consistent with bistable behaviour. (A) Single positive autoregulation. Two stable steady states are found in the single positive autoregilation model at (u, v) = (0.069, 11.94) and (u, v) = (9.85,0.122). (B) Double positive autoregulation. The model with double positive autoregulation has two stable steady states at (u, v)=(0.122, 9.85) and (u,v)=(11.94, 0.069).

On the other hand, negative autoregulation was not consistent with bistable behaviour for the parameter values used here. The vector plot of the switch with single negative autoregulation show that there is one stable steady state when the levels of the unregulated protein are high and the levels of the negatively regulated protein low. The vector plot for the switch with double negative autoregulation shows one stable steady state when the levels of both proteins are low.

Negative autoregulation occurs when the protein binds to its own promoter and

represses production. This means that the protein levels rise to a lower level than what would be expected for an unregulated system (Alon 2007). This means that, in the case of the double negative autoregulation, the concentration levels of either protein are not sufficient to dominate over the other and create bistability. In the case of the single negative autoregulation, only the protein without autoregulation is able to reach sufficient levels to dominate the system, and thus the only steady state of the system is when the unregulated protein is high.

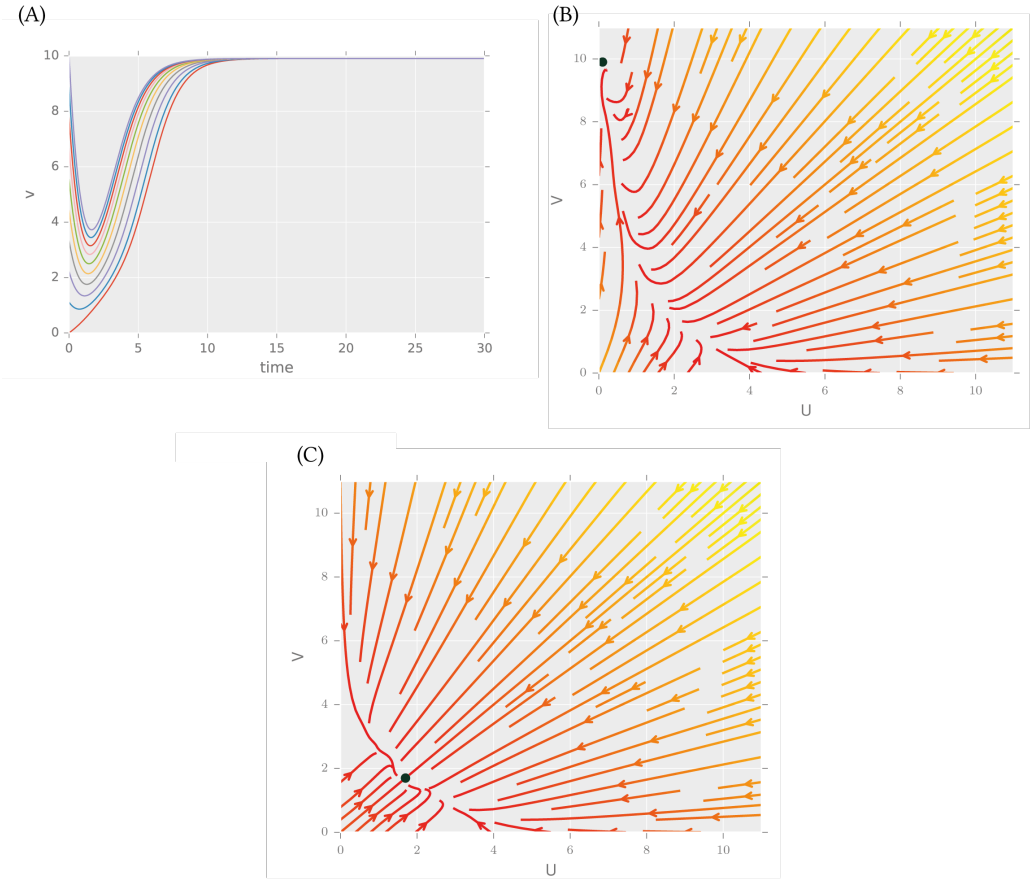

Figure 3.10 :(A) Timecourse of the single negative autoregulation switch, when k_u=2 with varying initial conditions. The system is monostable. (B) One stable steady state is found for the single negative autoregulation switch when k_u=2 at u=0.099 and v=9.903. (C) The vector plot of the switch with double negative autoregulation has one steady state at u=1.699 and v=1.699.

The models shown here to be capable of bistable behaviour will be used in the following section for model selection, to determine whether the addition of positive feedback loops increases the robustness of the system to parameter fluctuations.

3.4.2 ABC for model selection

Bayesian model selection is used to directly compare the competing designs using ABC-SysBio (Liepe et al. 2010). This method enables the simultaneous inference of the kinetic parameters and the structure of the models in order to select the model producing the same behaviour over a greater parameter range. ABC SMC can be used for model selection by adding the model as a parameter to the selection process. The algorithm samples models as well as parameters at each iteration. When the last ε is reached, the algorithm will have concluded to a posterior distribution of parameters for each model, a subset of the prior distribution that can give the best rise to the data. The model that performs over a greater posterior parameter range is selected as the most robust (Toni et al. 2009). The process is outlined in Figure 3.11.

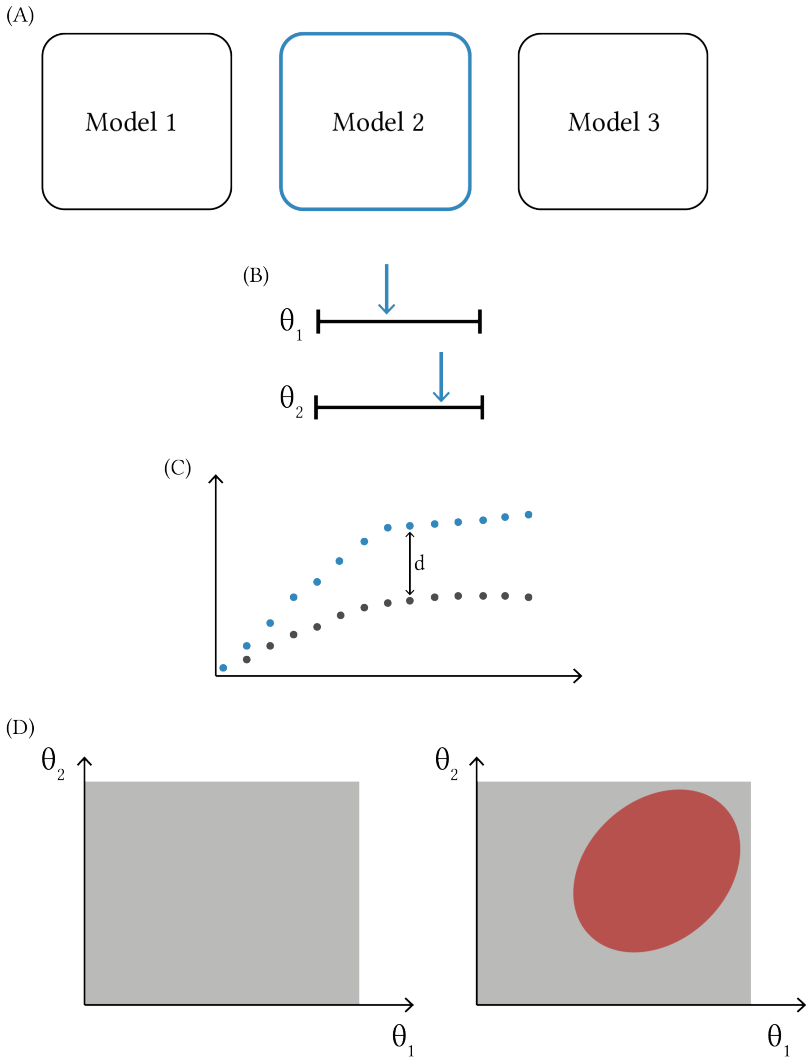

Figure 3.11 : ABC SMC model selection. (A) A model is sampled, and the (B) parameter values for that model are sampled. The system is simulated and the result compared to the desired behaviour. (C) If the distance between the two is smaller than the current accepted threshold, then the samples are retained. If not they are discarded. (D) This method can select the model that can produced the desired behaviour over a greater parameter range (red ellipse).

In order to select for the more robust toggle switch, the standard toggle switch was compared to switches with positive autoregulation in one or both nodes, which were shown to be capable of bistable behaviour in Section 3.4.1.1. The mass action models were used for model selection, in order to represent the system in a more realistic way. The equations shown in Table 3.4 are added to the equations of the simple toggle switch used in Section 3.3.2. Toggle switches with autoregulation on both nodes have both sets of equations added. The same design specifications as used in Section 3.3.2 were used.

Table 3.4 Autoregulated switches additional equations

Equations	Description
Positive autoregulation A	
$A2 + gA \xrightarrow{aut\_1} A2gA$	dimer self-association
$A2gA \xrightarrow{aut\_2} A + A2gA$	self-induced expression
$A2gA \xrightarrow{aut\_3} A2 + gA$	dimer self-dissociation
Positive autoregulation B	
$B2 + gB \xrightarrow{aut\_1} B2gB$	dimer self-association
$B2gB \xrightarrow{aut\_2} B + B2gB$	self-induced expression
$B2gB \xrightarrow{aut\_3} B2 + gB$	dimer self-dissociation

Given the models shown above and the parameter priors shown in Table 3.5, ABC SMC model selection was carried out.

Table 3.5 The prior distributions used for model selection. The values indicate the lower and upper limits of a uniform distribution.

ge	**rep**	**rep_r**	**dim**	**dim_r**	**deg**	**rep dim**	**rep_dim_r**	**deg_sr**	**aut_1**	**aut_2**	**aut_3**
1-10	1-10	0-5	1-10	0-5	1-5	0.05-0.1	0.01-0.1	0-1	0-10	0-2	0-10

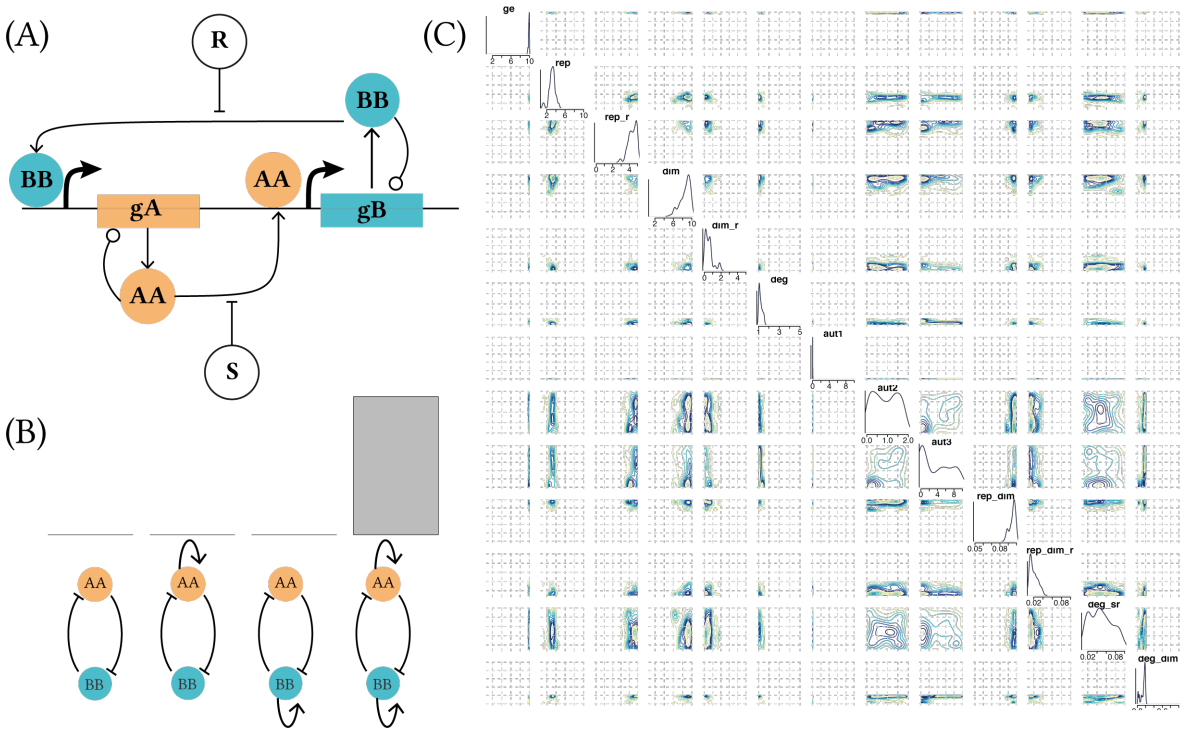

Figure 3.12 : (A) The toggle switch models considered. Interactions denoted by a circular arrow head can be positive of zero. (B) The toggle switch with positive autoregulation on A was found to be the most robust to parameter fluctuations. Three repeats of the model selection were carried out, and the median values are shown here. Upper and lower quartile error bars are included but are too small to be visible. (C) The posterior distribution of the toggle switch with positive autoregulation on A and B.

The results are shown in Figure 3.12. The toggle switch with positive autoregulation on both nodes was found to be the most robust model. This indicates that although all the models considered are capable of behaving like a switch, the model with double positive autoregulation could do that over a greater parameter range.

3.5 Discussion

Here I developed a more realistic model for the genetic toggle switch, using mass action. This model does not use the QSSA. In Chapter(XXX) I explore this model further and show that the QSSA is not necessary. I showed that this model is capable of bistable behaviour, within a given parameter range. I further studied this by using ABC SMC parameter inference to fit the toggle switch to a switching behaviour of choice. The parameter inference revealed the range of parameter values that can produce the behaviour of choice. These parameter values can be used to design a synthetic toggle switch that will behave in the specified manner.

Further, I showed that negative autoregulation is not consistent with bistable behaviour for the parameter values examined here. Adding small levels of single negative autoregulation to the system caused to revert to monostability. The only steady state was when the unregulated protein was high and the negatively regulated protein was low. This makes sense, as the negatively regulated protein cannot reach high enough levels to repress the other protein and dominate the system, whereas the unregulated protein is free to reach a higher steady state. In the case of double negative autoregulation, neither protein is free to reach sufficient levels to dominate the other, and the only steady state was when the levels of both proteins are low. This indicates that the system reaches a deadlock situation where both proteins are repressed and cannot reach a higher steady state. The models used here are deterministic and simplified down to two equations. Stochastic dynamics where noise is added to the system or a more complex model could be capable of overcoming this deadlock situation. More specifically, if transcriptional or translational bursting is included, the protein that receives the first boost can dominate the system on time and escape the deadlock situation (Strasser, Theis, & Marr 2012).

Finally, I showed that the addition of positive feedback loops make the genetic toggle switch more robust to parameter fluctuations. This means that the model was capable of producing the desired behaviour over a greater parameter range. This indicates that small fluctuations in parameters in the cellular environment will not affect the system's ability to be bistable, and thus makes it more suitable for use

in synthetic biological applications where a very constrained parameter set can be too restrictive. This makes it a better candidate for building new synthetic devices based on the toggle switch design.

The volume of the posterior distribution of even the most robust switch out of the ones examined here was still not large. This means that the behaviour of choice is still constrained, even after the addition of the positive feedback loops. A caveat of the analysis used here that has to be considered is that the parameter space is not searched for simply combinations that can produce a bistable switch. The behaviour that is required is very specific, and it is probable that the plethora of constraints put to the system result in the discard of parameter combinations that create a bistable switch. Firstly, a specific steady state level is required, for both the ON and OFF stages. When the switch is OFF, the protein levels must be as close to zero as possible, and when the switch is ON, the protein levels are required to approach 20nM. This requirement will discard any switches that have a higher or lower ON state, or a slightly higher OFF state. Additionally, the design specifications used here dictate that the time to reach steady state has to be quick. There is not much transition time allowed between the ON and OFF states, and the protein is required to reach steady state within a few timepoints. This also results in the exclusion of systems that reach steady state slower, but still act as a bistable switch. Therefore, the switches examined here, follow a very constrained behaviour. In the next Chapter I develop a method that is more flexible in the type of switches it can analyze.

3.6 Summary

In this chapter I studied the Gardner, Cantor, & Collins (2000) toggle switch and showed that it is bistable. I further studied the genetic toggle switch model using mass action. I identified the parameter ranges that produce a bistable behaviour and could be used as prior distributions for parameter inference. Further, I studied the effect of adding feedback loops has on the robustness of the genetic toggle switch. I found that the switch with double positive autoregulation is the most robust to parameter fluctuations. In the next chapter I address some of the shortcomings of the method used here by developing a new algorithm, StabiliyFinder.

4 Dynamics of multi-stable switches

4.1 Introduction

In this chapter, I aim to uncover the underlying principles that govern the stability of a given switch. To do this, I developed an algorithm, called Stability Finder, that can find the parameter value ranges that can produce the desired stability in a given model. I use this algorithm to examine a variety of switch architectures using different modelling abstractions.

Structurally, this chapter is organised as follows: In the first section I examine the current understanding of the stability landscape of the genetic toggle switch. Then, I discuss the development of Stability Finder, justify the choices made and the drawbacks of this method. In the sections following I apply Stability Finder to a variety of models and finally I discuss the implications our findings have to the overall understanding of the toggle switch stability.

4.2 Contributions to this Chapter

This phase plot analysis of the stability of the switch steady states shown in Figure 4.6 was carried out by Mae Woods, PhD.

4.3 Background

A circuit must be robust to a fluctuating cellular environment and its response and sensitivity must be able to be fine tuned in order to orchestrate a network of circuits that function together. A robust circuit can tolerate the compound stochasticity that a chain of circuits brings, and fine tuning of its response and sensitivity enables the researcher to make it sensitive to an upstream signal as well as influence a downstream subsystem. Parts can be fine tuned by developing component libraries (Lu,

Khalil, & Collins 2009), but this will be of little use if the required parameter ranges for parts to make a functional complex network are unknown, and will only perpetuate the cycles of trial-and-error. A computational method to find the range of parameter values that will produce the behaviour of choice is crucial to the design process by enabling the informed selection of appropriate parts from the libraries. For example, if it is known that gene expression must be low for a given stability, one can select a weak promoter or a low copy plasmid for the desired construct.

Both analytical and computational approaches have been deployed for the study of the toggle switch. Analytical approaches are limited to simpler models and thus require a number of assumptions to be made. The system under consideration has to be reduced to very few equations and parameters in order to make the system solvable. This requires assumptions to be made about the system that cannot always be justified, such as the quasi-steady state approximation (QSSA). The QSSA assumes that the binding/unbinding processes are much faster than any other process (**Loinger:2007vo**), thus the bound intermediate is assumed to always be in steady state. The QSSA assumption is met *in vitro* but often does not hold *in vivo* and its misuse can lead to large errors and incorrectly estimated parameters (Pedersen, Bersani, & Bersani 2007). Moreover, it is generally not possible to solve even simple stochastic models analytically, and these methods are restricted to deterministic models. The computational and graph-theoretic approaches developed for the study of multistationarity generally focus on deciding on whether a given system is incapable of producing multiple steady states (Conradi et al. 2007; Banaji & Craciun 2010; Feliu & Wiuf 2013). For example, Feliu & Wiuf (2013) developed an approach using chemical reaction theory and generalised mass action modelling (Feliu & Wiuf 2013). No approach exists that can handle both deterministic and stochastic systems in an integrated manner.

For this purpose, I developed a computational framework based on sequential Monte Carlo that takes a model and determines whether it is capable of producing a given number of (stable) steady states and the parameter space that gives rise to the behaviour. Uniquely, this can be done for both deterministic and stochastic models, and also= complex models with many parameters, thus removing the need for simplifying assumptions. This framework can be used for comparing the conclusions drawn by various modelling approaches and thus provides a way to investigate appropriate abstractions. I have made this framework into a python package, called Stability Finder.

I use this methodology to investigate genetic toggle switches and uncover the

design principles behind making a bistable switch, as well as those necessary to make a tristable and a quadristable switch (4 steady states). The number of stable steady states will be referred to as the desired stability of the model in this thesis. I also demonstrate the ability of Stability Finder to examine more complex systems and examine the design principles of a three gene switch. The examples I used demonstrate that Stability Finder will be a valuable tool in the future design and construction of novel gene networks.

4.4 Stability Finder algorithm

To investigate the multistable behaviour of systems, I had to make a number of extensions to existing approaches. Firstly, a wide range of initial condition samples are required in order to determine the stability of a system. For a given set of parameter values, sample points are taken across initial conditions using latin hypercube sampling (McKay, Beckman, & Conover 2000), and the ensemble system simulated in time until steady state. As a distance function I use the desired stability of the simulated model. An overview of the algorithm is given in Section 4.4.1.

4.4.1 Algorithm overview

The Stability Finder algorithm is summarised below. Stability Finder is available as a Python package, and can be downloaded from https://github.com/ucl-cssb/StabilityFinder.git.

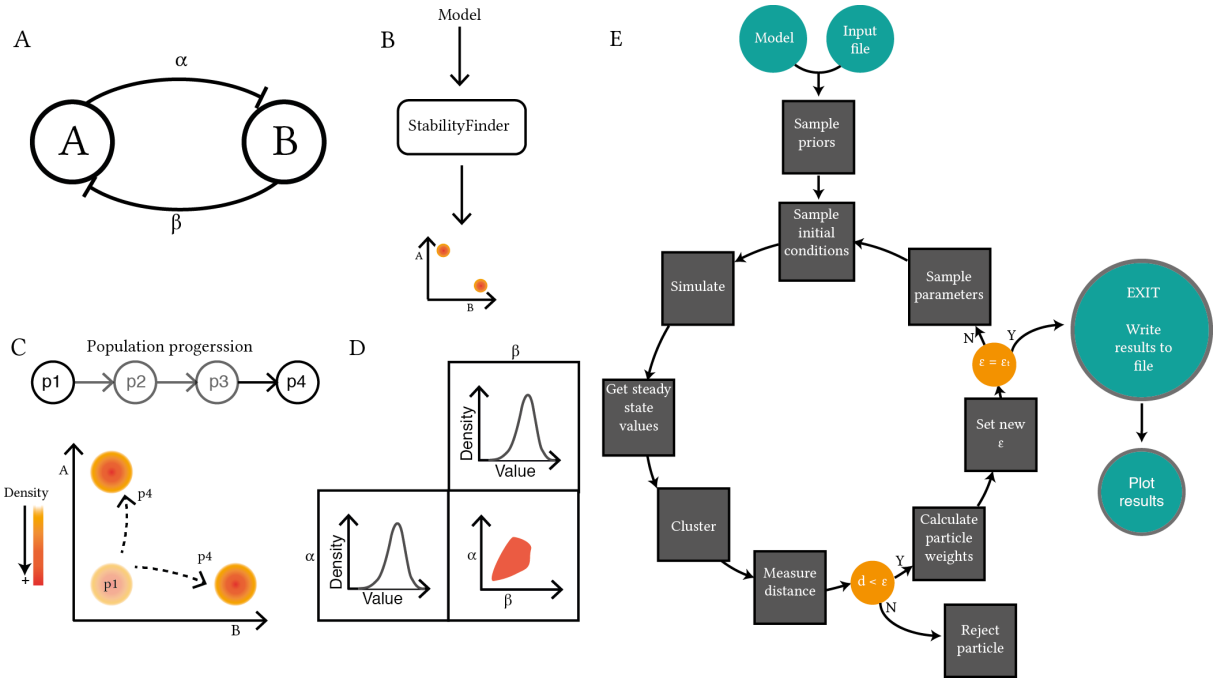

Figure 4.1 : Using sequential Monte Carlo to examine system stability. The algorithm takes as input a model (A) and evolves it to the stability of choice (C) via intermediate populations. In this example model shown in A, There are two species and two parameters. For the model to be bistable, the phase plot of the two species of interest must have two distinct densities, as shown in (C). The parameter space of the model is searched through our algorithm until the resulting simulations give rise to bistability. The parameter values for the model that demonstrated the desired behaviour are given as an output (D). The output consists of the accepted values for each parameter, as well as each density plotted against the other. This allows us to uncover correlations between parameter values. We made this algorithm into a python package, called Stability Finder. The overview of the algorithm is shown in (E).

The user provides an SBML model file (Finney, Hucka, & Le Novere 2003; Hucka et al. 2004) and an input file that contains all the necessary information to run the algorithm, including the desired stability and the final tolerance, ϵ, for the distance from the desired behaviour necessary for the algorithm to terminate. The flow of execution is illustrated in Figure 4.1E. Since the algorithm is computationally intensive, all deterministic and stochastic simulations are performed using algorithms implemented on Graphical Processing Units (GPU)s, which are used for mutli-threaded computation (Kirk & Hwu 2010).

4.4.2 Initial condition sampling

In Stability Finder, latin hypercube sampling is used to sample initial conditions (McKay, Beckman, & Conover 2000). This is used to ensure that the whole space is sampled uniformly. Latin hypercube sampling is done in two dimensions in Stability Finder. This is the same algorithm used for initial condition sampling in Chapter 3, and the reader is referred to Section 3.3.1 for a description of this algorithm. Stability Finder could easily be implemented to be used for a larger number of species, but here it has only been used for stability analyses concerning two species. Stability landscapes involving more that two species are beyond the scope of this thesis.

4.4.3 Clustering methods

Whether the model was simulated using ODEs or the Gillespie algorithm (Gillespie 1977) dictated the method of clustering that I used. For the deterministic models I used an algorithm I developed, that will be referred to as the delta clustering algorithm in this thesis. This algorithm consists of defining the number of clusters by counting a new cluster every time a data point is more than a distance δ away from any existing clusters. The benefits of the delta clustering algorithm are that it is fast and can be used on deterministic solutions, where steady state values tend to be identical if all the particles have reached steady state.

Steady states of stochastic models are clustered using the K-means clustering (Lloyd 1982) and the number of clusters determined using the Gap statistic (Tibshirani, Walther, & Hastie 2001). This method is more suited to stochastic solutions, where the delta clustering method would fail as the steady state solutions tend to be more widely dispersed than in the deterministic case. The detailed algorithms used are shown in Appendix C.

The method used for clustering can be altered by the user if he/she wants to add their own preferred clustering algorithm that might be more appropriate for their specific purposes. For the models I used here, the above methods were successful in clustering the steady state solutions.

4.4.4 Distance function

The distance function is used to compare the desired behaviour to the behaviour observed in each particle (Toni et al. 2009). In Stability Finder the distance function consists of three distances. The first one is the difference between the number of desired clusters and the number of clusters observed in the phase plot. For this distance metric the number of clusters in the phase plot must be calculated. The clustering methods used are outlined in section 4.4.3.

The other two distance metrics used in Stability Finder are the variance within each cluster and the overall, between cluster, variance. The within cluster variance ensures that the clusters are tight, and the between cluster variance is used to ensure the clusters are far apart from each other. In the context of this thesis, the ideal behaviour of a system is tight, widely separated clusters. This means that the genetic system has distinct steady states, and the difference in the protein levels between each steady state is observable.

$$d_1 = C = M_c \tag{4.1}$$

$$d_2 = V_{tot} = \frac{1}{n}\sum_{i=1}^{n}(x_i - \mu) \tag{4.2}$$

$$d_3 = V_{cl} = \underset{m_c=1}{\overset{Mc}{\text{median}}}\{\frac{1}{n_{m_c}}\sum_{c=1}^{n_{m_c}}(x_c - \mu_c)\}, \tag{4.3}$$

where n denotes the total number simulations per particle, x refers to each simulation steady state of a given particle and μ the mean value of all the simulations steady states in each particle. M_c denotes the total number of clusters per particle and m_c refers to each cluster. x_c and μ_c represent the simulation steady state of a given particle in the current cluster and the mean of the current cluster respectively.

Once the distance from the desired behaviour has been calculated, the algorithm rejects any particles whose distance is farther than the current ε. The distances taken into account are the number of clusters (C), the total variance (V_{tot}) and the within cluster variance (V_{cl}) as outlined in Equations 4.2-4.3. In addition to these distances I have included another two checks for the particles. Firstly, Stability Finder checks if the simulation of a particle has reached steady state. If the standard deviation of the last ten time points in the simulation (denoted as SS_{ss}) is larger than a user-specified value, then the particle is rejected. This is to ensure that only particles that have reached steady state are considered. Secondly, there is a check for the minimum level of the steady states (denoted as SS_l). This is to allow the user to select for steady states whose protein levels are above a certain threshold. This has to be added as an additional check as the steady state levels must be experimentally observable if they are to be used to design new systems. Two steady state levels of very low levels would be biologically indistinguishable and thus meaningless in an experimental setup. This check is optional to the user, and can be set to zero if not desired. In summary, a particle must satisfy all of the criteria given below in order to be accepted:

$$M_c \leq \varepsilon_{M_c}$$
$$V_{tot} \leq \varepsilon_{V_{tot}}$$
$$V_{cl} \leq \varepsilon_{V_{cl}}$$
$$SS_{ss} \leq \varepsilon_{SS_{ss}}$$
$$SS_l \leq \varepsilon_{SS_l}$$

4.4.5 Model checking

A problem that can arise by using this method with stochastic simulations is that the behaviour observed may not be the true behaviour but it might be a result of noise. We need to ensure that the resulting behaviour is reproducible. Therefore, I added model checking to the algorithm. Model checking consists of resampling from the posterior distribution and simulating each sample. If the resulting behaviour is the same as what we expected we can be confident that it is the true behaviour of the system and not a result of noise.

Algorithm 4 StabilityFinder algorithm

1: Initialise $t = 0$,
2: $i = 0$
3: **if** t = 0 **then**
4: Sample particle from prior, $\theta^{**} \sim \pi(\theta)$
5: **else**
6: Sample θ^{*} from the previous population $\{\theta^{i}_{t-1}\}$ with weights w_{t-1}.
7: Perturb the particle, $\theta^{**} \sim K_t(\theta|\theta^{*})$ where K_t is the perturbation kernel.
8: **end if**
9: Sample k initial conditions $\{x^{k}_{0}\}$ via latin hypercube sampling.
10: Simulate k datasets to steady state, $\{x^{*k}\}$, from the the model, $x^{*} \sim f(x|\theta, x_0)$
11: Apply clustering in phase space on $\{x^{*k}\}$
12: Calculate the distance $d = \rho(\{x^{*k}\}, y)$.
13: **if** $d \leq \epsilon_t$ **then**
14: $\theta^{i}_{t} = \theta^{**}$. $i = i + 1$
15: **if** $i \leq N$ **then** GoTo step 3
16: **else**
17: Calculate weight for each accepted θ^{i}_{t}
18: $$w^{(i)}_{t} = \begin{cases} 1, & \text{if } t = 1 \\ \frac{\pi(\theta^{(i)}_{t})}{\sum_{j=1}^{N} w^{(j)}_{t-1} K_t(\theta^{(j)}_{t-1}, \theta^{(i)}_{t})}, & \text{if } p \geq 1. \end{cases}$$
19: Normalise weights
20: $t = t + 1$.
21: **if** $t \leq N_t$ **then**
22: GoTo step 3
23: **end if**
24: **end if**
25: **end if**

4.5 Calculating robustness

Unlike Toni et al. (2009), Stability Finder does not have model selection integrated into the method. This is because the purpose of Stability Finder is not necessarily to compare models for robustness but to elucidate the stability a given model is capable of. Nevertheless, robustness analysis is an outcome that Bayesian methods are well suited for. Therefore, here I discuss another algorithm I developed in order to extract robustness information from the results of Stability Finder and apply model selection.

As discussed in Section(XXX), two models can be compared for their robustness using Equation(XXX). This represents the ratio of the robustness measure of each model which in turn is defined as the ratio between the volume of functional region F and the volume of the prior P. In order to calculate the Bayes factor we must first be able to approximate the volume of the viable parameter space. The viable parameter space is the space that approximates the posterior distribution that can give rise to the desired behaviour. I tested two methods of approximating the volume of the viable space, which are outlined in Algorithm 5. The first method is based on the method used by (Hafner et al. 2009), where the volume of the cuboid containing all the viable space is calculated. I modified this part of their method by only including the area of the viable space where the majority of the last population lies. Therefore only the 1^{st} and 99^{th} percentile of the viable space are taken into account. This is necessary in order to exclude outliers in the distribution that would skew the volume calculation significantly. Each parameter represents a side in the cuboid and since the volume of a cuboid is equal to the product of its sides, the volume of the viable space is equal to the product of the ranges of all the parameters. This cuboid method will be prone to overestimating robustness especially in cases of correlation between parameters. This caveat could be alleviated if a Principal component analysis (PCA) (Fukunaga 2013) is done on the data before the cuboid is calculated (Hafner et al. 2009). This would align the axes of the cuboid to the major axes of the distribution. This would still be a crude estimation of the volume, since if the posterior distribution is assumed to be normally distributed the volume would still be overestimated.

Thus we used a second method, where the volume of the viable space was represented by a hyper-ellipsoid, an ellipse in higher dimensions. This method should not be as prone to overestimation of robustness as the cuboid method as an ellipsoid can take correlation into account. For this method the distribution of the viable space is assumed to be normal. The method calculates the covariance matrix of the

distribution, whose volume is given by Equation 4.4. Just as in the cuboid method, the 1st and 99th percentile of the data is ignored.

$$V = \frac{2\pi^{\frac{k}{2}}}{k\Gamma(\frac{k}{2})}\left[\chi_k^2(\alpha)\right]^{\frac{k}{2}}|\Sigma|^{\frac{1}{2}}, \tag{4.4}$$

where k is the number of dimensions, Γ is the Gamma function, α is the confidence interval required and $|\Sigma|$ is the determinant of the covariance matrix.

To validate these methods I compare them to ABC-SysBio model selection (Liepe et al. 2014). ABC-SysBio has been used extensively for model selection (Toni et al. 2009; Toni et al. 2011; Barnes et al. 2011) There is good agreement between the three methods as can be seen in Figures 4.2 and 4.3.

Algorithm 5 Approximating robustness

for each model m of M **do**
  Prior$\sim U(a,b)$
  $V_{prior}^m = \prod_{i=1}^k (i_b - i_a)$
  Get $1^{st} < data < 99^{th}$ percentiles
  if Cuboid calculation **then**
    $V_{post}^m = \prod_{i=1}^k (i_{max} - i_{min})$
  end if
  if Ellipsoid calculation **then**
    Calculate data covariance matrix
    $V_{post}^m = \frac{2\pi^{\frac{k}{2}}}{p\Gamma(\frac{k}{2})}\left[\chi_k^2(\alpha)\right]^{\frac{k}{2}}|\Sigma|^{\frac{1}{2}}$
  end if
  $R^m = \frac{V_{post}^m}{V_{prior}^m}$
  $R_{norm}^m = \frac{R_i^m}{\sum_{i=1}^M R_i^m}$
end for

We use the following two examples used in ABC-SysBio (Toni et al. 2009):

4.5.1 Case study 1: Infectious diseases

For the first case study utilizes the models used in (Toni et al. 2009). As described in Toni et al. (2009), the models describe the spread of an infectious disease through a population over time. The population is made up of susceptible, infected or recovered individuals, denoted as S, I and R respectively. Three models are compared for the robustness of their posterior distributions. The first model (Model 1), is the simplest model of the three. Each individual S or R can be infected once and then it can immediately infect other individuals (Toni et al. 2009).

$$\dot{S} = \alpha - \gamma SI - dS \tag{4.5}$$
$$\dot{I} = \gamma SI - vI - dI \tag{4.6}$$
$$\dot{R} = vI - dR, \tag{4.7}$$

where a denotes the birth rate, d the death rate, γ the infection rate, and v the recovery rate.

The second model, Model 2, includes a time delay between an individual getting infected and being infectious. δ denotes the rate of transition of a non-infectious infected individual to an infectious one.

$$\dot{S} = \alpha - \gamma SI - dS \tag{4.8}$$
$$\dot{L} = \gamma SI - \delta L - dL \tag{4.9}$$
$$\dot{I} = \delta L - vI - dI \tag{4.10}$$
$$\dot{R} = vI - dR, \tag{4.11}$$

Finally the third model, Model 3, extends Model 1 and includes the recovered individuals being able to become susceptible again. This is denoted by rate e.

$$\dot{S} = \alpha - \gamma SI - dS + eR \tag{4.12}$$
$$\dot{I} = \gamma SI - vI - dI \tag{4.13}$$
$$\dot{R} = vI - dR - eR, \tag{4.14}$$

The three models are simulated using ODEs. In ABC-SysBio model selection is used. Parameter inference is also used for each model separately without the use

of model selection. I used the two methods outlined in Algorithm 5 to calculate the robustness of the posterior distributions of all three models. This robustness measure was then compared to the result of ABC-SysBio model selection. As shown in Figure 4.2, there is good agreement between the three measures of robustness. The posterior distributions of all three models are also shown in Figure 4.2.

4.5.2 Case study 2: Population growth

The second example I will use to demonstrate the effectiveness of the methods used here for robustness calculation is a population growth model. This is another example used in ABC-SysBio (Toni et al. 2009).

The data was obtained by simulating an immigration-death model shown in Equation 4.15. This model (referred to as Model 1) and a model of logistic growth are compared for robustness of their posterior distributions. Model 1:

$$\frac{dI}{dt} = \alpha - \beta I \tag{4.15}$$

Logistic growth, model 2:

$$\frac{dI}{dt} = \gamma - I(\delta - \epsilon I) \tag{4.16}$$

As in Section 4.5.1, two analyses were carried out on these two models. First, ABC-SysBio model selection was used to find the most robust model. Then parameter inference was done on each model. The resulting posterior distributions (shown in Figure 4.3), were compared for robustness using the cuboid and the ellipsoid approximation methods. All three robustness measures find that Model 1 is the most robust model. The analysis was repeated for ODE, MJP and SDE simulations, all arriving to the same result of Model 1 being the most robust. The results are shown in Figure 4.3.

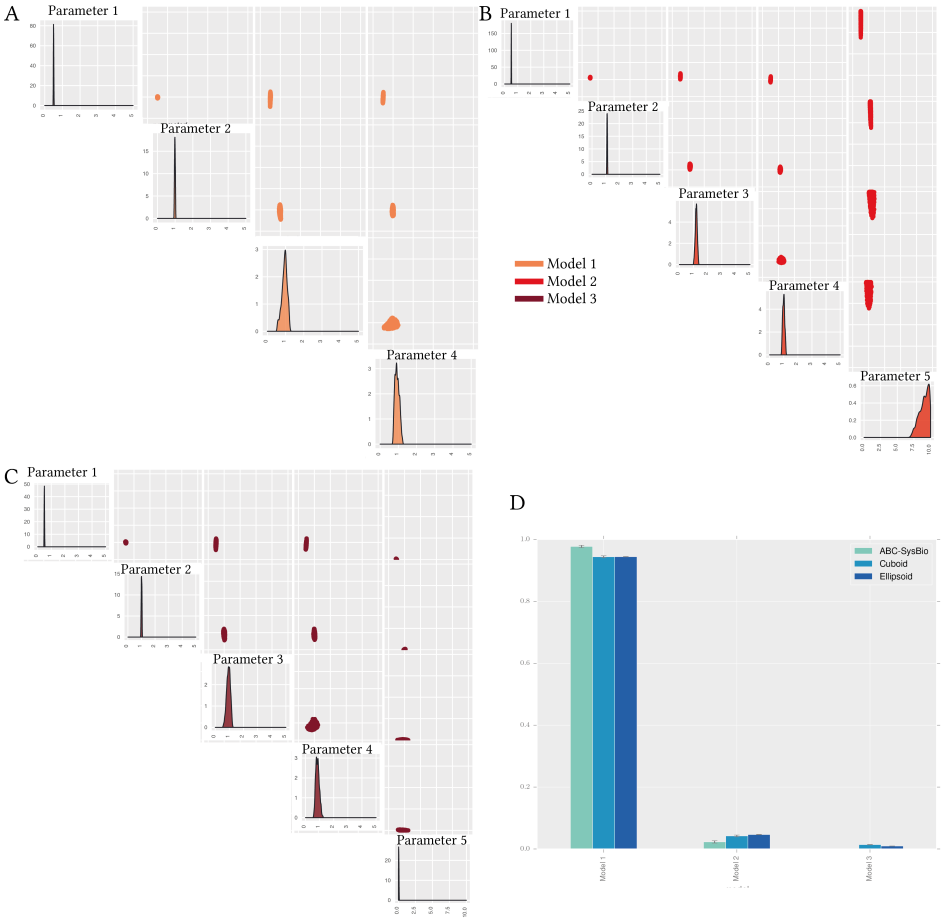

Figure 4.2 : Robustness analysis of the three models for the spread of infectious diseases. (A-C) The posterior distributions of the three models compared. (D) I use three methods to calculate robustness, ABC-SysBio model selection, the volume of the hyper-cuboid approximation of the posterior distribution and the volume of the hyper-ellipsoid approximation of the posterior distribution. Each analysis was repeated three times. The height of the bars indicate the mean robustness from the three repeats and the error bars represent the standard deviation. There is good agreement between all three methods. All three methods show that Model 1, the simplest model, is the most robust model.

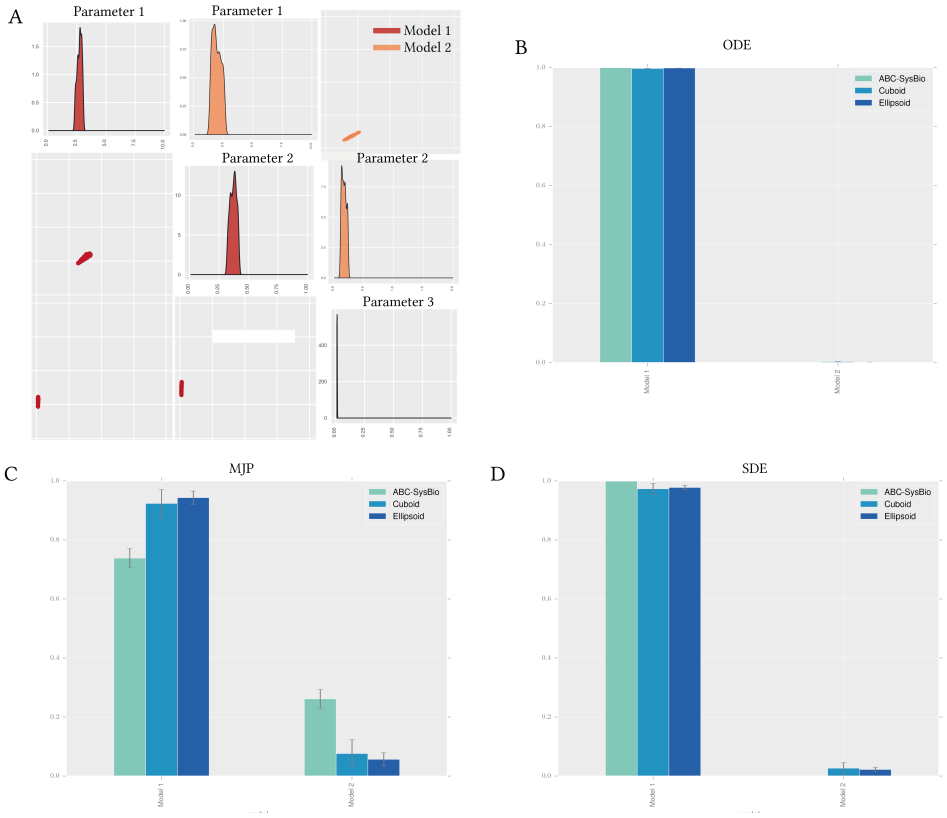

Figure 4.3 : Robustness comparison of two population growth models. (A) The posterior distributions of the two models. (B-D) The models were simulated using ODE, MJP and SDE. Both the cuboid and the ellipsoid approximations agree with ABC-SysBio model selection results. Each analysis was repeated three times. The height of the bars indicate the mean robustness from the three repeats and the error bars represent the standard deviation.

The two case studies used above show that the cuboid and the ellipsoid approximation of model robustness agree with the results obtained from ABC-SysBio model selection. A point I must draw attention to is that for ABC-SysBio model selection where model selection is incorporated in the process, each model is also considered a particle with an associated weight (Toni et al. 2009). If a model is performing poorly it does not proceed in the algorithm and is dropped when the weight falls low enough so that the model is not sampled (Toni et al. 2009). This can save time in the analysis as computational resources are not wasted on 'dead' models, models that perform the required behaviour poorly. Using Stability Finder for model selection, each model must reach the given final ε in order for the cuboid and ellipsoid methods to be valid. This means that time and computational power will be spent on models that are potentially a bad fit, or that have posterior distributions so small compared to the prior that it will take a long time for Stability Finder to find it. Despite this, the results agree between the all three methods of model selection. This shows that the requirement for all models to reach the final ε does not affect the results for the models used in the above case studies. The potentially wasted computational resources on 'dead' models is a compromise we make in order to be able to run the models separately, as model selection is not the primary purpose of Stability Finder.

4.6 Applications of Stability Finder

In this section I apply Stability Finder to switch models in order to find the design principles underlying their stabilities. First I apply it to a simple model with known results, the Gardner, Cantor, & Collins (2000) toggle switch. This model can serve as a test for Stability Finder, as the conditions for bistability are derived in Gardner, Cantor, & Collins (2000).

4.6.1 Testing StabilityFinder

Gardner, Cantor, & Collins (2000) constructed the first synthetic genetic toggle switch (Gardner, Cantor, & Collins 2000). Their model consisted of two mutually repressing transcription factors, as shown in Figure 4.4A, and in the deterministic case is defined by the following ODEs:

$$\frac{du}{dt} = \frac{a_1}{1 + v^{\beta}} - u \tag{4.17}$$

$$\frac{dv}{dt} = \frac{a_2}{1+u^\gamma} - v, \tag{4.18}$$

where u is the concentration of repressor 1, v the concentration of repressor 2, a_1 and a_2 denote the effective rates of synthesis of repressors 1 and 2 respectively, β is the cooperativity of repression of promoter 1 and γ of repressor 2. Gardner, Cantor, & Collins (2000) studied the deterministic case and concluded that there are two conditions for bistability for this model; that a_1 and a_2 are balanced and that β, γ >1 (Gardner, Cantor, & Collins 2000). I test Stability Finder by using it to find the posterior distribution for which this model exhibits bistable behaviour. Therefore, the desired behaviour is set to two steady states, and using a wide range of values as priors as shown in Table 4.1, I used Stability Finder to find the parameter values necessary for bistability to occur. The posterior distribution calculated by Stability Finder for the Gardner deterministic case is shown in Figure 4.5A.

Table 4.1 Gardner switch priors in the deterministic and stochastic cases

Parameters (μM/h)				Species (μM)	
a_1	β	a_2	γ	s_1	s_2
0-60	0-5	0-60	0-5	0-100	0-100

These results agree with the results reported by Gardner, Cantor, & Collins (2000). For this switch to be bistable a_1 and a_2 must be balanced while β and γ must both be >1, as can be seen in the marginal distributions of β and γ in Figure 4.5A.

I next applied Stability Finder to the case of the Gardner switch under stochastic dynamics using the same priors as the deterministic case, and again searched the parameter space for bistable behaviour. The posterior distribution is shown in Figure 4.5B. We can see that the conditions on the parameters required for bistability in the deterministic case generally still stand in the stochastic case. There appears to be slightly looser requirements on the parameters of the stochastic model (wider marginal distributions). Some difference between the deterministic and stochastic posteriors is expected as different clustering algorithms are used for the stochastic and the deterministic cases. The Gap statistic is used in the case of the stochastic case, as it is capable of dealing with noisier data whereas a simpler and faster algorithm is used for clustering the deterministic solutions. These results demonstrate that Stability Finder can be used to find the parameter values that can produce a desired stability and can be confidently applied to more complex models.

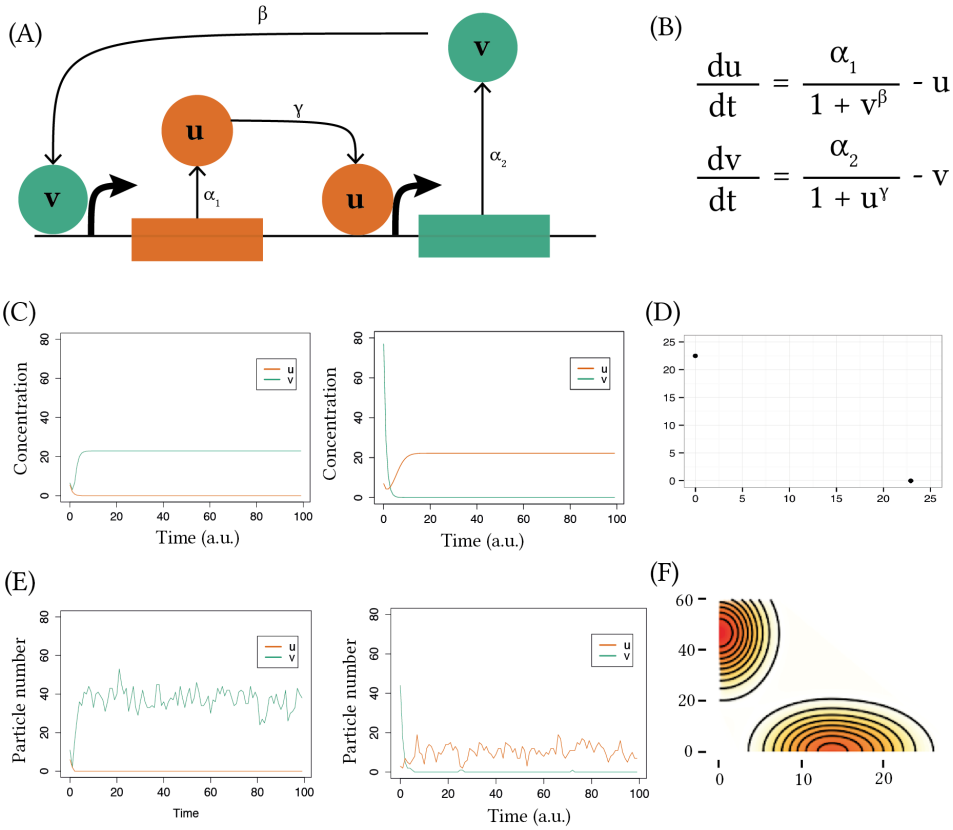

Figure 4.4 The Gardner switch model used to test Stability Finder. The Gardner model (A) consists of two mutually repressing transcription factors. It can be reduced to a two-equation system (B) , where u and v are the two transcription factors, a_1,a_2 are their effective rates of synthesis, u,v are their concentrations and β, γ represent the cooperativity of each promoter. (C) Two samples of deterministic simulated timecourses of the Gardner switch and (D) The resulting phase plot. (E) Two samples of timecourses of the stochastic simulations and (F) the resulting stationary distributions.

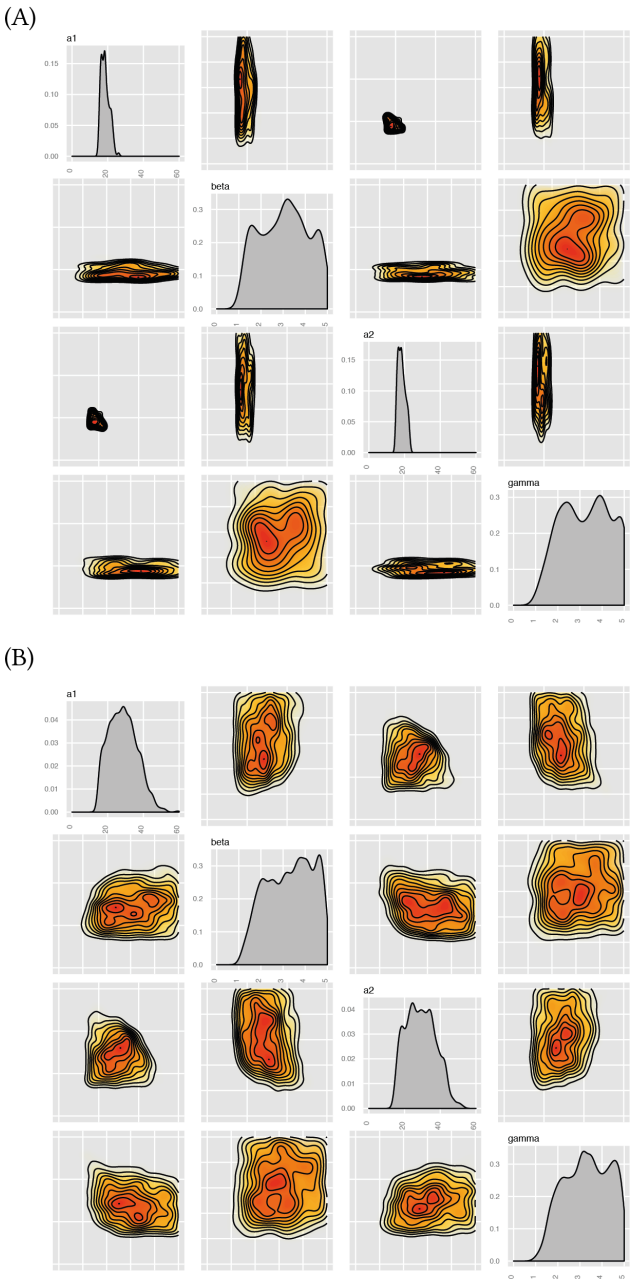

Figure 4.5 Elucidating the stability of the Gardner switch. The Gardner model has four parameters, for which I want to find the values for which this system is bistable. I use Stability Finder to find the posterior distribution of the bistable Gardner switch, deterministically (A) and stochastically (B). The posterior distributions are shown as the density plots of each parameter as well as each one plotted against the other.

4.6.2 Lu toggle switch models

Next I analyzed an extension of the Gardner switch model developed by Lu, Onuchic, & Ben-Jacob (2014). I use these models as they are of increased complexity from the Gardner model. Lu, Onuchic, & Ben-Jacob (2014) considered two types of switches, the classic switch consisting of two mutually repressing transcription factors (model CS-LU), as well as a double positive switch DP-LU. The CS-LU switch was found to be bistable given the set of parameters used, while the DP-LU switch was found to be tristable (Lu, Onuchic, & Ben-Jacob 2014). The CS-LU model used in their study is given by the following system of ODEs.

$$\dot{x} = g_x H^S_{xy}(y) - k_x x \tag{4.19}$$

$$\dot{y} = g_y H^S_{yx}(x) - k_y y, \tag{4.20}$$

where:

$$H^S_I(x) = H^-_I(x) + \lambda_I H^+_I(x) \tag{4.21}$$

$$H^-_I(x) = 1/\left[1 + (x/x_I)^{n_I}\right] \tag{4.22}$$

$$H^+_I(x) = 1 - H^-_I(x), \tag{4.23}$$

and the DP-LU model is given by

$$\dot{x} = f_x(x,y) = g_x H^S_{xy}(y) H^S_{xx}(x) - k_x x \tag{4.24}$$

$$\dot{y} = f_y(x,y) = g_y H^S_{yx}(x) H^S_{yy}(y) - k_y y, \tag{4.25}$$

g_I represents the production rate, k_I the degradation rate, n_I the Hill coefficient, x_I the Hill threshold concentration and λ_I the fold change of the transcription rates, and $I \in \{xy, yx, xx, yy\}$.

For the parameter values used in the Lu study, the CS-LU switch exhibits three steady states, two of which are stable and one is unstable. The CS-LU switch exhibits five steady states, of which three are stable and two are unstable. Bifurcation diagrams of the two Lu models are shown in Figure 4.6.

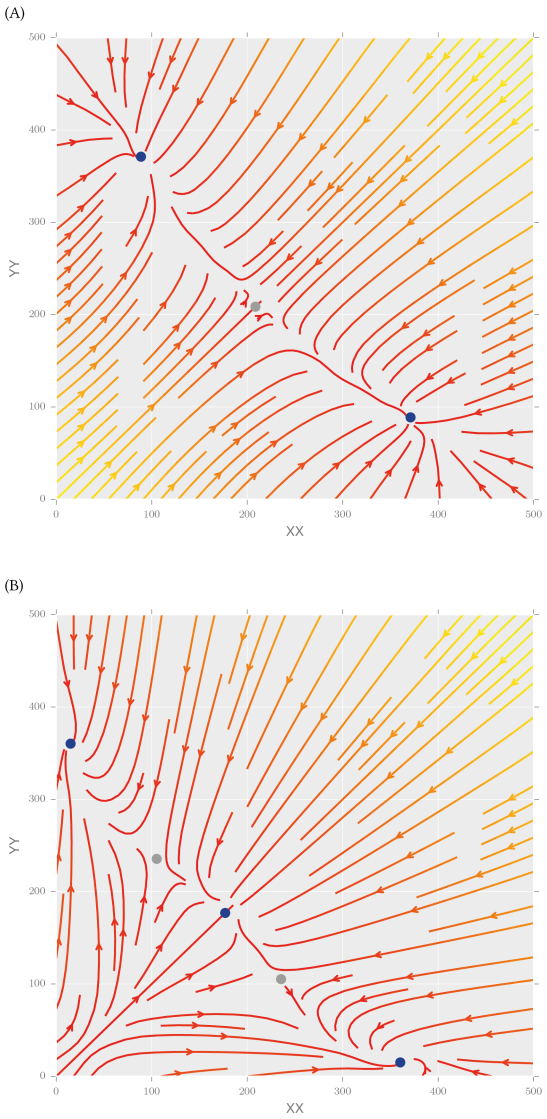

Figure 4.6 Stream plot of the vector plot of the (A) CS-LU and DP-LU switches. The colours indicate the magnitude of the vectors, with yellow indicating high and red low values. The blue points represent stable steady states and the grey points represent unstable steady states.

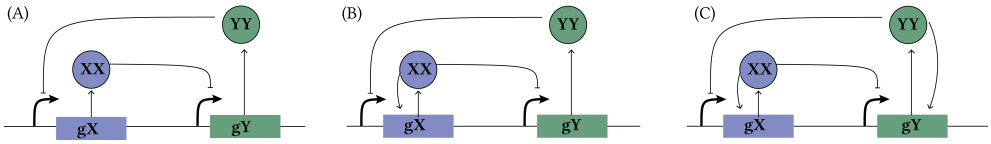

Figure 4.7 The three LU toggle switch models. (A) CS-LU, (B) SP-LU and (C) DP-LU.

4.6.2.1 Extending the Lu models

I start the analysis of the Lu models by extending their analytical approach to solving the system. I use Stability Finder to explore a larger parameter space which allows us to distinguish between rare events and robust behaviours. The advantage of using Stability Finder over solving the system analytically is that the full parameter space is explored rather than solving the system for a single set of parameters. This allows us to deduce model properties that could not otherwise be identified. Robustness to parameter fluctuations can be explored, as well as parameter correlations and restrictions on the values they can take while still producing the desired behaviour.

It is known that that the addition of positive autoregulation to the classical toggle switch can induce tristability (Lu, Onuchic, & Ben-Jacob 2014). Here I investigate the interplay of positive autoregulation on the values of the other parameters in the model. I extended the analysis presented in Lu, Onuchic, & Ben-Jacob (2014) by including the switch with single positive autoregulation (model SP-LU), where an asymmetry of positive feedbacks is present between the two genes. The three switches considered in this analysis are shown in Figure 4.7.

The SP-LU switch is modelled using the following ODE system

$$\dot{x} = g_x H^S_{xy}(y) H^S_{xx}(x) - k_x x \tag{4.26}$$

$$\dot{y} = g_y H^S_{yx}(x) - k_y y. \tag{4.27}$$

Using Stability Finder with priors centred around the parameter values used in the original paper (see Table 4.2), we can identify the most important parameters for achieving the models' stability. The phase plots of the final populations of the models are shown in Figure 4.8 and the posterior distribution of these models are shown in Figure 4.9A. We find that the parameters representing the rates of degradation of the transcription factors in the system (k_x,k_y) must both be large in relation to the prior ranges for bistability to occur. Protein degradation rates have been shown to be important for many system behaviours including oscillations (**Woods:2015vu**).

Table 4.2 Priors of the classical(CS-LU), single positive (SP-LU) and double positive (DP-LU) models.

Parameter	Symbol	CS-LU (nM/min)	SP-LU (nM/min)	DP-LU (nM/min)
Production rate	gx	30-50	1-2	1-100
	gy	30-50	20-25	1-100
Degradation rate	kx	0-0.5	50-55	0-1
	ky	0-0.5	48-52	0-1
Hill coefficient	nxy	1-5	30-35	0-10
	nyx	1-5	0.1-0.2	0-10
Hill thresholds concentration	xxy	100-300	2-3	100-1000
	xyx	100-300	0.4-0.6	100-1000
Transcription rate fold change	lxy	0-0.5	0.02-0.04	0-1
	lyx	0-0.5	0.02-0.04	0-0.2
Hill coefficient	nXX	-	25-30	0-10
	nYY	-	0.01-0.02	0-10
Hill thresholds concentration	xXX	-	0.4-0.5	50-500
	xYY	-	1-3	50-500
Transcription rate fold change	lXX	-	65-72	1-20
	lYY	-	0.02-0.04	1-20

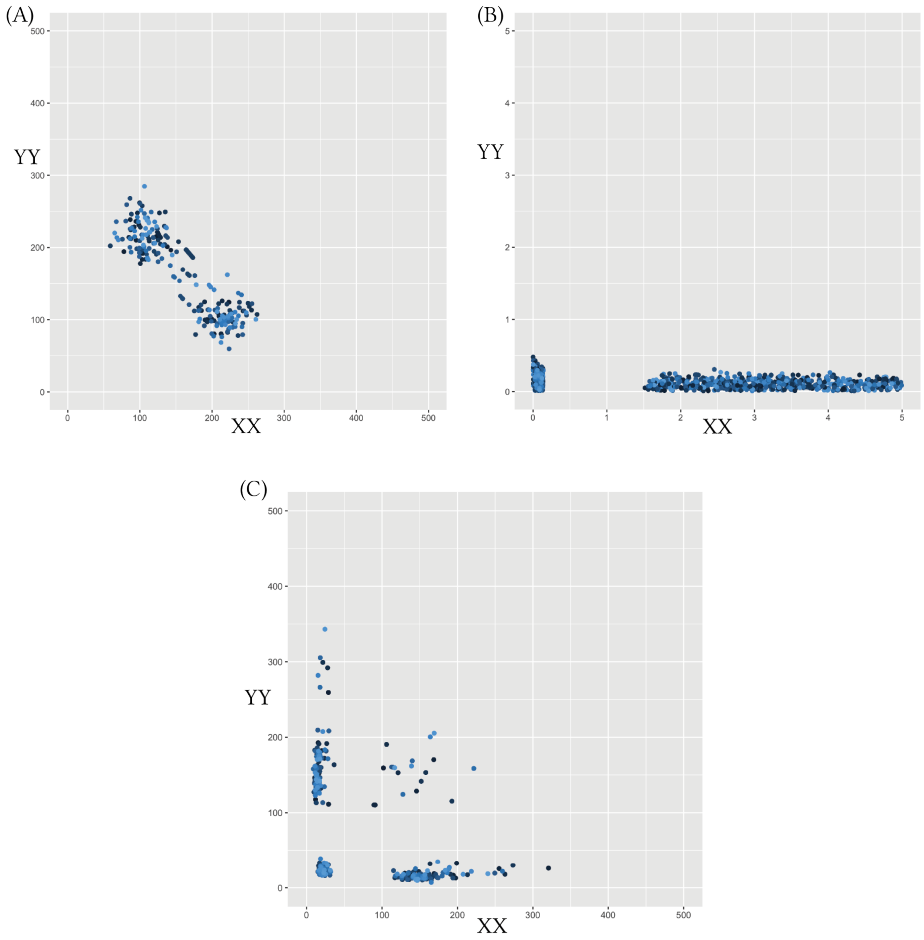

Figure 4.8 : The phase plots of 100 particles from the last population of the three Lu switches. (A) The bistable CS-LU (B) The bistable SP-LU and (C) The tristable DP-LU. We find two types of tristable behaviour, one where the third steady state is zero-zero and one where the third state is high (non-dead).

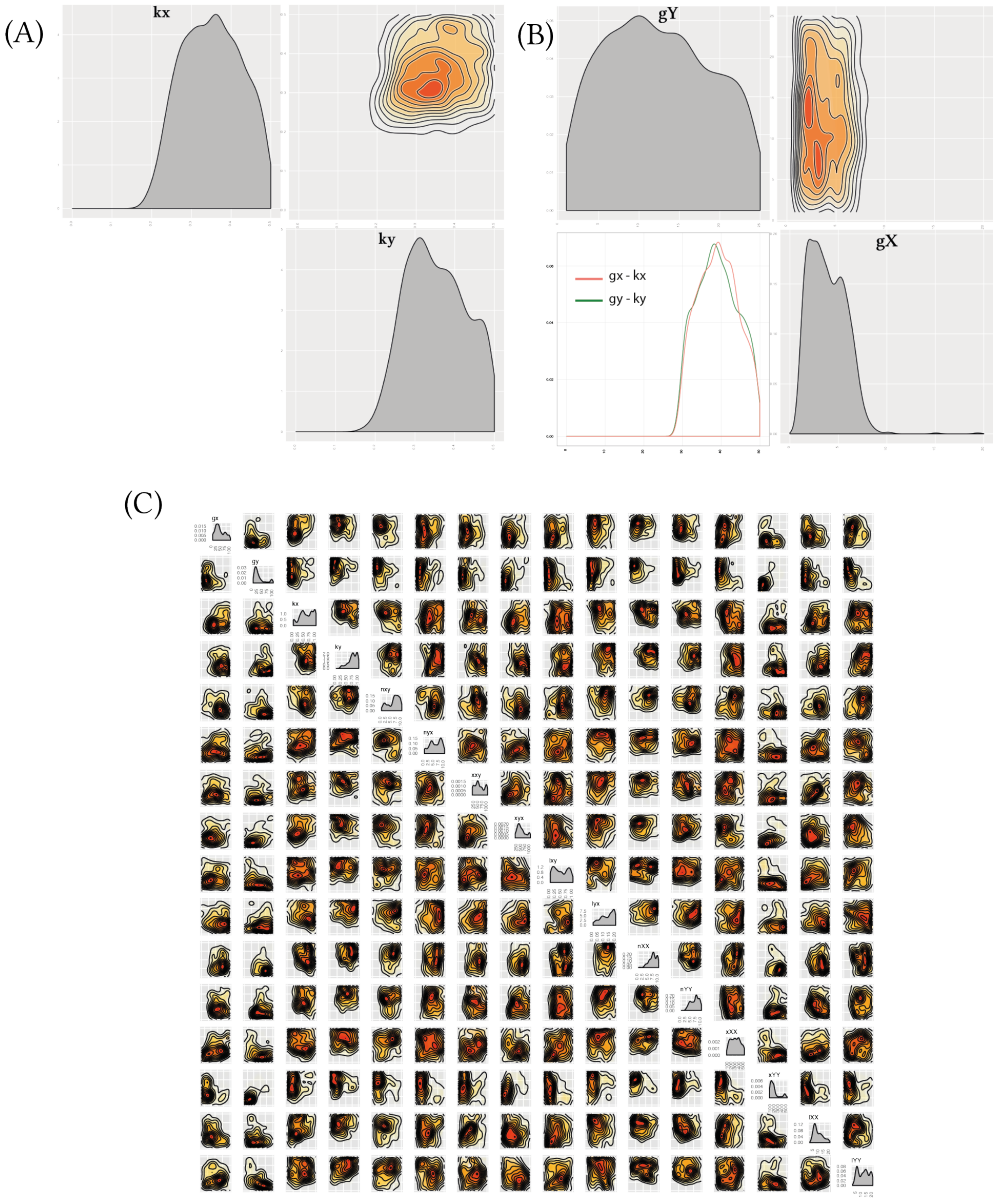

Figure 4.9 : The three variants of the Lu models. (A) The CS-LU switch is bistable. The most restricted parameters for this behaviour are kx and ky which both have to be high relative to the prior while the net protein production for X and Y must be balanced. (B) The extended Lu model with a single positive autoregulation on X. This model is bistable when gx is small, but the net production of protein is equal for the two nodes. (C) The Lu model with double positive autoregulaiton is tristable, and its posterior distribution shown here.

We find that the switch with single positive autoregulation is capable of bistable behaviour as seen in Figure 4.9B, but this is only possible when the strength of the promoter under positive autoregulation, gx, is small (Figure 4.9). There appear to be no such constraints on the strength of the original, unmodified, promoter, gy.

Upon examination of the DP-LU model, we also find that tristability in the switch is relatively robust, as tristability is found across a large range of parameter values, with no parameters strongly constrained. Two types of tristable behaviour are identified, one where the third steady state is at (0,0) and and one where the third steady state has non-zero values, as seen in Figure 4.8. This result agrees with previous work by Guantes & Poyatos (2008), who found that a switch can exhibit two kinds of tristability, one in which the third steady state is high (III_H) and one in which it is low (III_L) (Guantes & Poyatos 2008).

4.6.2.2 Multistability in the Lu models

The DP switch is capable of both bistable and tristable behaviour as well as 4 coexisting states under deterministic dynamics (quadristability) (Guantes & Poyatos 2008). It is of great interest to understand the conditions under which these three behaviours occur. We carried out a bifurcation analysis of the DP switch using the PyDSTool (Clewley 2012) in order to get an indication of the stabilities this model is capable of, and at which parameter ranges these are found.

Since the Lu models can be solved analytically, we can obtain the bifurcation diagram of the DP-LU by keeping all parameters constant apart from gene expression (gx). The result shown in Figure 4.10B, the system can exhibit 2, 3 or 4 steady states depending on the value of the gene expression rate. We observe that if $100 \leq gx \leq 120$ the system exhibits four steady states, if $9 \leq gx \leq 10$ the system is tristable and if $10 \leq gx \leq 100$ the system is bistable. I use the whole range tested above ($0 \leq gx \leq 140$) as prior distributions in Stability Finder and searched parameter space for 2, 3 and 4 steady states.

Using Stability Finder we obtain a more complex picture of the parameter space that can produce each behaviour. This is because, unlike the bifurcation analysis, Stability Finder does not require any of parameters to be fixed. Since there are no such restraints on the value each parameter can take we obtain a bigger range of parameters that can produce each behaviour than the ranges found during the bifurcation analysis. The priors used for each analysis are identical and include the whole range of values found in the bifurcation diagram, varying only the required number of steady states. In addition, unlike the bifurcation analysis the values for

gx and gy are not forced to be equal in the analysis done on Stability Finder.

Using StabilityFinder, we obtained posterior distributions for bistable, tristable and quadristable behaviours in the DP-LU model and then compared the posterior parameter distributions (Figure 4.10). Upon examination of the posterior distributions for all three switches we observe that a subset of the posterior parameter values is different under the three behaviours. We find differences in the univariate distribution of the parameters for gene expression, gx, as highlighted in Figure 4.10, box 1. This parameter must be small for a quadristable switch to occur but there are no such restraints for a bistable or a tristable switch. Furthermore, parameter xXX must be small for three and four steady states to be achieved but there are no such restraints for a bistable switch, as can be seen in Figure 4.10, box 2.

We also find a difference in the bivariate distributions in the posterior. Most notably, we find that parameters xXX and gX are tightly constrained in the tristable and the four steady state cases, where both parameters are required to be small, but less so in the bistable case (Figure 4.10, box 3). Another notable difference is between parameters xXX and nXX shown in Figure 4.10, box 5, where they are constrained and in the tristable and four steady state cases but not the bistable case. Interestingly, we also find parameter correlations conserved between the three behaviours, as seen in Figure 4.10, box 4, where parameters lXX and gx, positive autoregulation and gene expression are negatively correlated in both cases. This highlights the importance of treating unknown parameters as distributions rather than fixed values when studying the parameter values of a model, as they are capable of uncovering not only the ranges and values needed but also the correlations between parameters that would not have otherwise been detected.

I further analyse these models by studying the phase plots resulting from simulating the particles from the posterior distribution to steady state. The phase plots from 100 particles from each posterior are shown in Figure 4.11. We find that there is a strong conservation on the locations of the steady states between each particle. This indicates that the steady states in a two-node toggle switch tend to be symmetrical. This gives rise to the patterns seen in Figure 4.11. This is especially evident in the quadristable switch. For every steady state at (0,0) there is another steady state on its diagonal, at $XX = YY$. All the combinations of these two steady states form the straight line seen in Figure 4.11C. This indicates that two of the four steady states exist where XX=YY. The other two exist where one of the two proteins dominates the other.

This same principle can be seen in the bistable and the tristable switches. In the

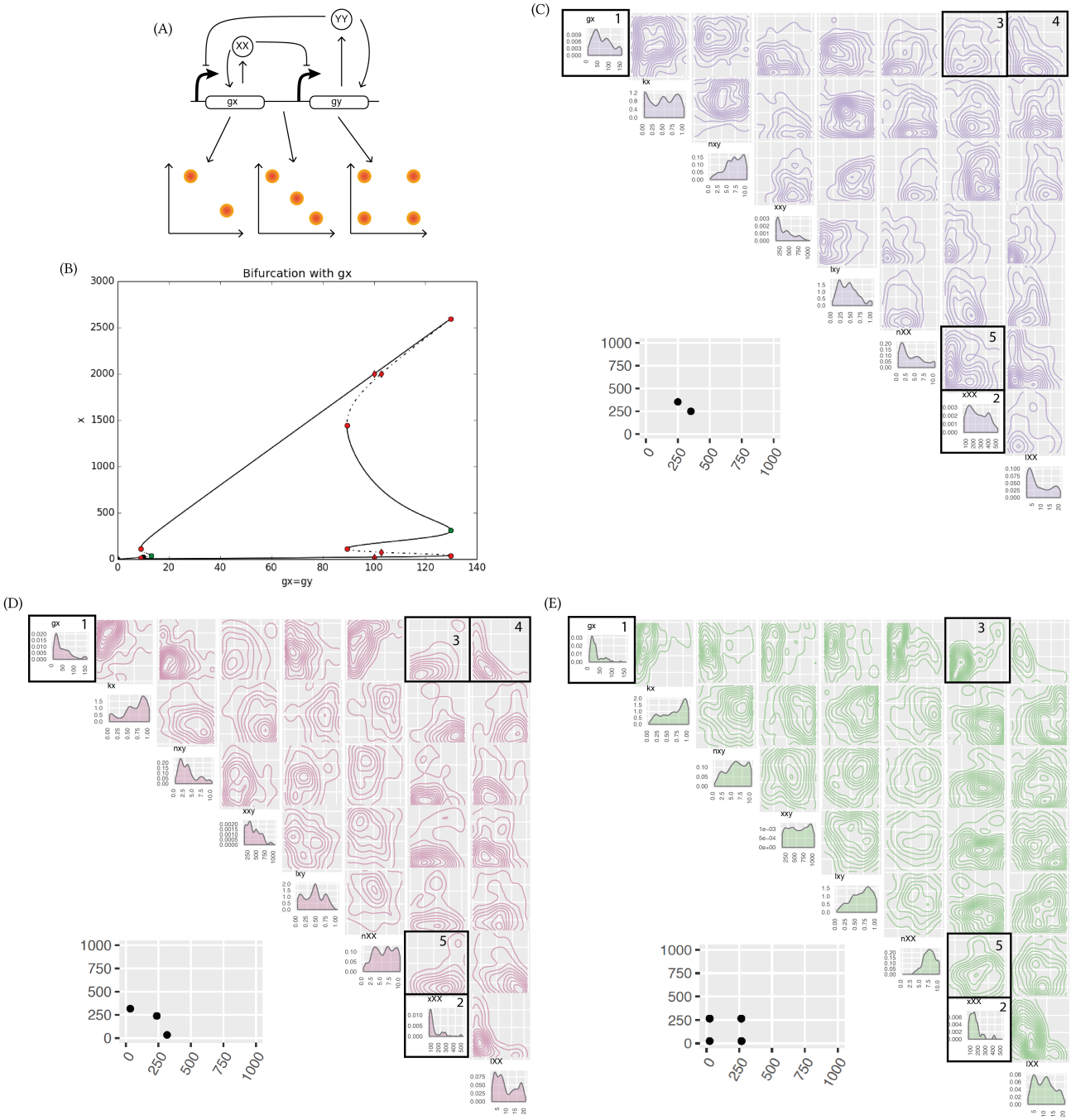

Figure 4.10 : Design principles of multistable switches. (A) Using the Lu model with added positive autoregulation we uncover the design principles dictating if a switch will be bistable, tristable, or will have 4 steady states. (B-D) By considering the bivariate distributions of the parameters we can uncover the differences in the parameters of a bistable switch compared to a tristable switch, compared to a quadristable switch . The posterior distribution of the bistable switch is shown in purple, of the tristable switch in pink and of a quadristable in green. The bivariate distributions for which a difference is observed between the stabilities are in black boxes. An example of a phase plot from each behaviour is shown next to the corresponding posterior distribution.

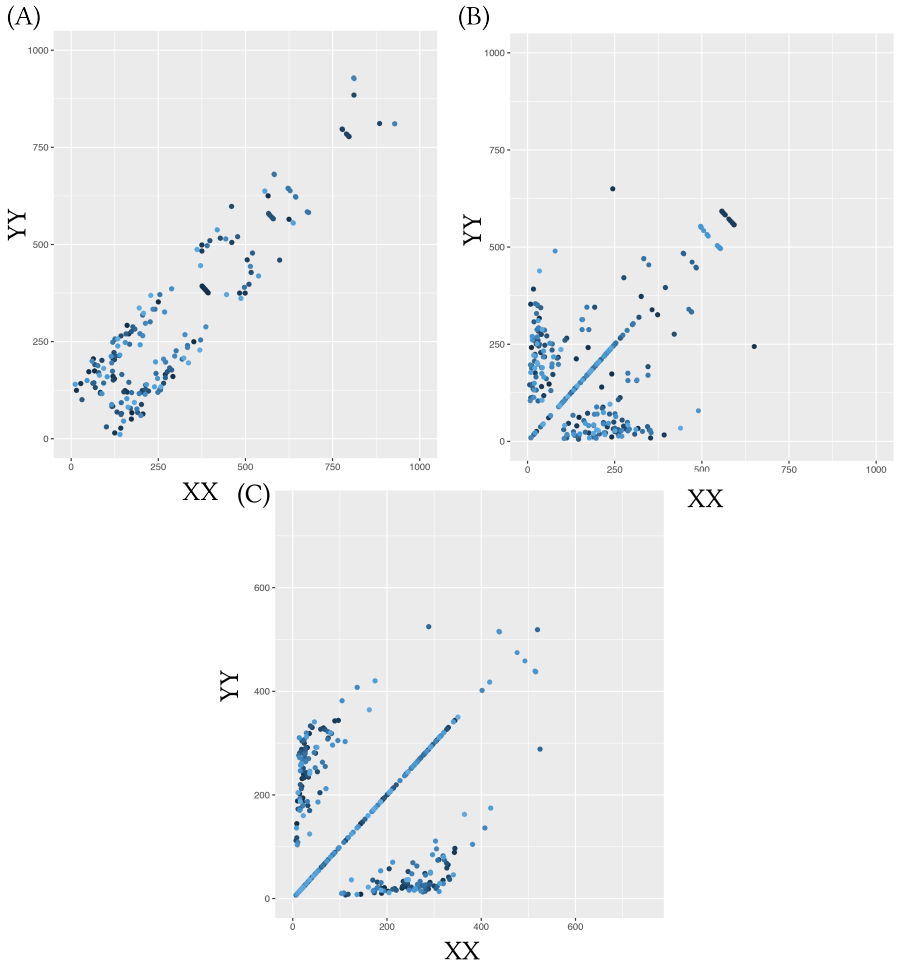

Figure 4.11 : The phase plots from 100 particles from each posterior. (A) The acrshortcs-lu (B) SP-LU and (C) DP-LU. Each particle is represented by a different shade of blue. We find a strong conservation on the location of the steady states between particles.

bistable switch the two steady states are also symmetrical and one never completely dominates the other. For the tristable case we observe that two of the steady states exist where the levels of one protein is much larger than the other, and a third steady state exists where $XX = YY$. We also observe that the third steady state is not necessarily a 'dead' state, but they can exist over a range of values for XX and YY.

4.6.2.3 Extending the Lu switch to three nodes

To further demonstrate the flexibility of Stability Finder I investigated a system capable of higher stabilities. Multistability is found in differentiating pathways, like the myeloid differentiation pathway (Ghaffarizadeh, Flann, & Podgorski 2014; Cinquin & Demongeot 2005). I allow for these more complex dynamics by extending the DP-LU model by adding another gene, making it a three gene switch. This new system is depicted in Figure 4.12A. This model has symmetric parameters, which means that the parameters for equivalent reactions (e.g. gene expression) are the same. In Stability Finder I look for six steady states, the output being in nodes X and Y and using the priors shown in Table 4.3. We successfully find that the system is capable of six steady states, as shown in Figure 4.12C.

Table 4.3 Priors used in the three-node switch

Parameter	Symbol	Range (nM/min)
Production rate	gx	3-5
Degradation rate	kx	0-0.2
Hill coefficient	nxy	0-2
Hill thresholds concentration	xxy	140-160
Transcription rate fold change	lxy	0-0.2
Hill coefficient	nxx	2-4
Hill thresholds concentration	xxx	90-110
Transcription rate fold change	lxx	8-12

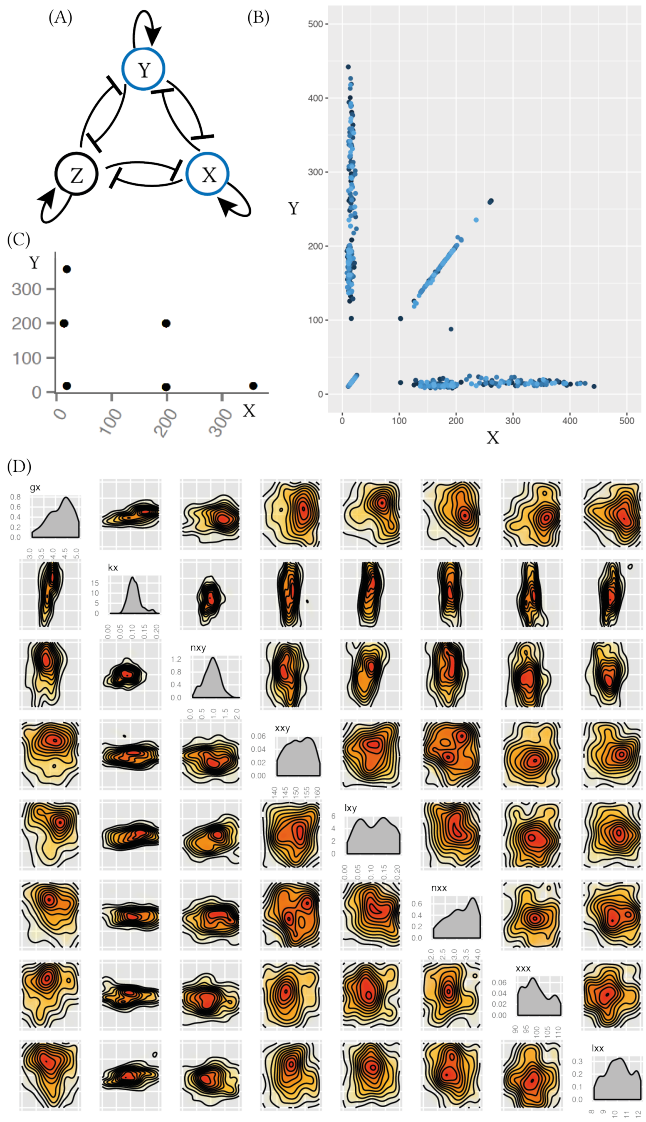

Figure 4.12 : The three-node mutual repression model, with added positive auto-regulation on each node. (A) The model. The model is studied in two dimensions using Stability Finder, for nodes X and Y. (B) The phase plot of 100 particles from the posterior found by Stability Finder. There are 6 steady states. (C) The posterior distribution of the 6-steady state three-node system. Parameters kx and nxy are the most constrained.

We find that the most constrained parameters for this behaviour are again the degradation rate of the proteins, kx. If they are too large or too small the system will not exhibit hexa-stability. Additionally we find that the Hill coefficients for the repressors, nxy, are constrained to be smaller than 1.5 as seen in Figure 4.12D.

Consistently with the results found in section 4.6.2.2, we find that the steady states are symmetric (Figure 4.12B). Each of six steady states exists in symmetry with another one, in tightly constrained regions. This example demonstrates that Stability Finder can be used to elucidate the dynamics of more complex network architectures, which will be key to the successful design and construction of novel gene networks as synthetic biology advances.

4.6.3 Mass action switches

In order to study the switch system in a more realistic way, I developed an extension to the switches used in Sections 4.6.1 and 4.6.2. This new set of switches does not use the quasi-steady state approximation (QSSA) that is often used in modelling the toggle switch. Using mass action, this changes the two-equation system used in Gardner, Cantor, & Collins (2000) and Lu, Onuchic, & Ben-Jacob (2014) into a system of 8 ODEs and 10 parameters in the classical switch case with no autoregulation (model CS-MA). The equations describing the system are shown below.

$$\mathrm{gA} \xrightarrow{\mathrm{ge}} \mathrm{gA} + \mathrm{A}$$
$$\mathrm{gB} \xrightarrow{\mathrm{ge}} \mathrm{gB} + \mathrm{B}$$
$$\mathrm{A} + \mathrm{A} \xrightarrow{\mathrm{dim}} \mathrm{A2}$$
$$\mathrm{A2} \xrightarrow{\mathrm{dim\,r}} \mathrm{A} + \mathrm{A}$$
$$\mathrm{B} + \mathrm{B} \xrightarrow{\mathrm{dim}} \mathrm{B2}$$
$$\mathrm{B2} \xrightarrow{\mathrm{dim\_r}} \mathrm{B} + \mathrm{B}$$
$$\mathrm{gA} + \mathrm{B2} \xrightarrow{\mathrm{rep}} \mathrm{B2gA}$$
$$\mathrm{B2gA} \xrightarrow{\mathrm{rep\_r}} \mathrm{B} + \mathrm{gA}$$
$$\mathrm{gB} + \mathrm{A2} \xrightarrow{\mathrm{rep}} \mathrm{A2gB}$$
$$\mathrm{A2gB} \xrightarrow{\mathrm{rep\_r}} \mathrm{A2} + \mathrm{gB}$$
$$\mathrm{A} \xrightarrow{\mathrm{deg}} \emptyset$$
$$\mathrm{B} \xrightarrow{\mathrm{deg}} \emptyset$$
$$\mathrm{A2} \xrightarrow{\mathrm{deg\_dim}} \emptyset$$
$$\mathrm{B2} \xrightarrow{\mathrm{deg\_dim}} \emptyset$$

For the model with added double positive autoregulation (model DP-MA) the following equations are added to the system:

$$\begin{aligned}
A2 + gA &\xrightarrow{\text{aut 1}} A2gA \\
A2gA &\xrightarrow{\text{aut 2}} A + A2gA \\
A2gA &\xrightarrow{\text{aut 3}} A2 + gA \\
B2 + gB &\xrightarrow{\text{aut 1}} B2gB \\
B2gB &\xrightarrow{\text{aut 2}} B + B2gB \\
B2gB &\xrightarrow{\text{aut 3}} B2 + gB
\end{aligned}$$

The ODEs describing the above switches are shown in Appendix (XXX). These models are too complex to be solved analytically and I use Stability Finder to fit the models to a bistable behaviour. The two models used and the resulting phase plots are shown in Figure 4.13 and the posterior distributions obtained are shown in Figure 4.14.

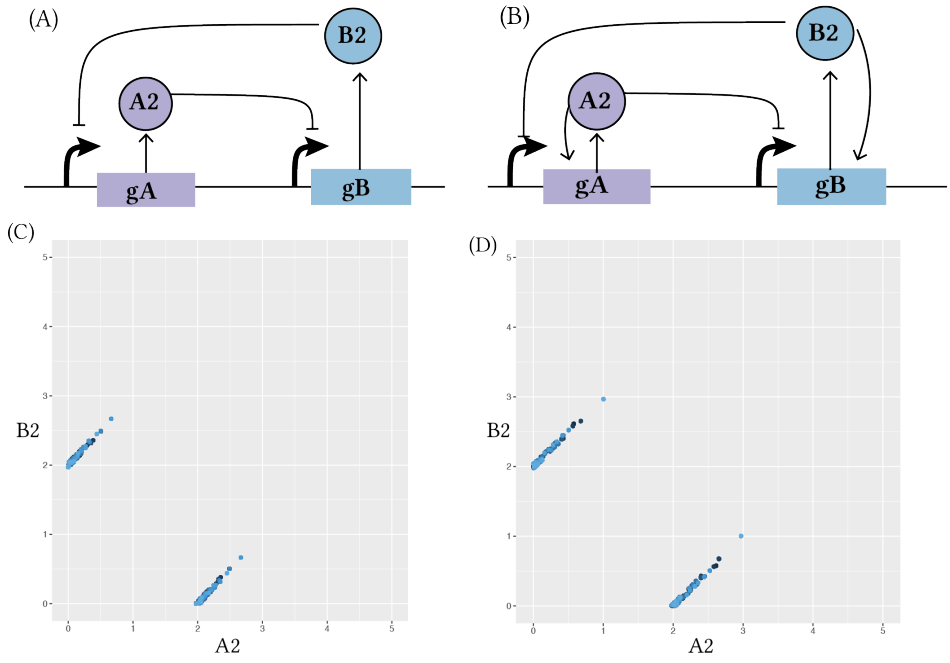

Figure 4.13 : The two mass action switches I developed. (A) The simple switch CS-MA (B) The switch with double positive autoregulation DP-MA. (C, D) The phase plots of 100 particles simulated from the posterior distributions of the bistable mass action switches.

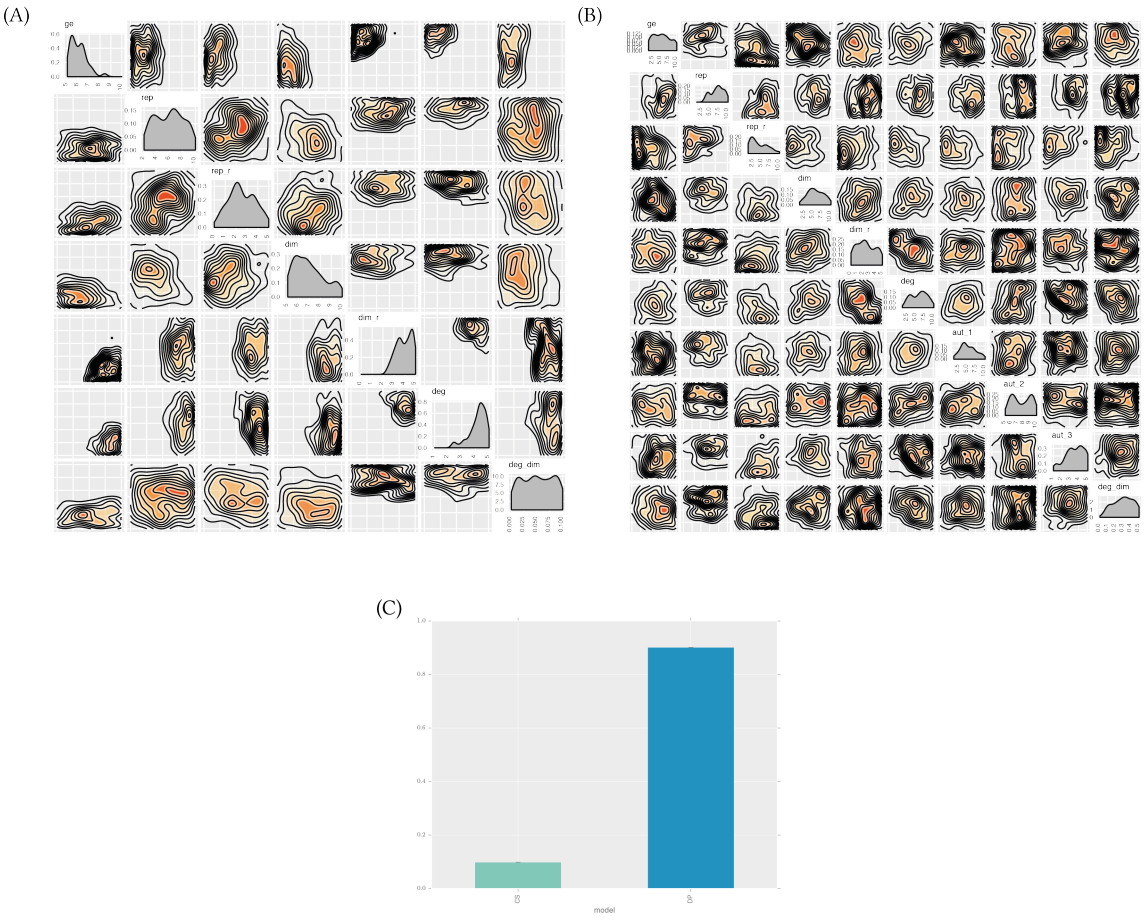

Figure 4.14 : The posterior distributions of the symmetric deterministic (A) CS-MA and (B) DP-MA switches. (C) Robustness comparison of the two models.

By examining the posterior distributions shown in Figure 4.14 we see that the CS-MA is much more constrained that the DP-MA switch. We find that gene expression must be low for bistability to occur in the CS-MA model but there is no such constraint in the DP-MA model. We also find that the monomerization rate dim_r and the monomer degradation rate deg must both be larger than 2. This is not found in the DP-MA model.

Next I compare the two models for robustness using the ellipsoid method described in Section 4.5. We find that the addition of positive feedback loops greatly increases the system's robustness to parameter fluctuations as seen in Figure 4.14C. Adding positive feedback loops to the model allows it to be bistable over a greater range of parameter values. This indicates that small fluctuations in parameters in the cellular environment will not flip the switch and thus makes it more suitable for use in synthetic biological applications where spontaneous and undesired switching might be detrimental. This makes it a better candidate for building new synthetic devices based on the toggle switch design. We identified the parameter region within which these models are bistable, information that is important when building such a device in the lab.

The models used in the above analysis assume the parameters for equivalent reactions are equal. This is a constraint that simplifies the model. When building this model into a synthetic system in the lab, this assumption is not necessarily justified. When choosing promoters to build this synthetic system two promoters can be chosen to have similar strength but their strength will not necessarily be identical. In order to study how this might affect the results, I further eliminate modelling assumptions made in the toggle switch by making the parameters representing gene expression (ge) and repression (rep), as well as the protein degradation parameters asymmetric (independent parameters for each protein, versus fixed to be equal). We find that the features of the posterior distributions of the symmetric and the asymmetric models remain the same. The posterior distribution of the asymmetric CS-MA model was more constrained than the posterior distribution of the asymmetric DP-MA model.

We further study the asymmetric mass action models by examining the QSSA aproximation. As stated above, the QSSA is a common analytical tool for model simplification. By examining the posterior distributions of the asymmetric CS-MA and DP-MA models we find that the QSSA does not necessarily hold for these models (Figure 4.15). That is, the systems function as switches even when the QSSA does not hold. These assumptions, necessary for the reduction of the model, are

therefore not always justified in this case.

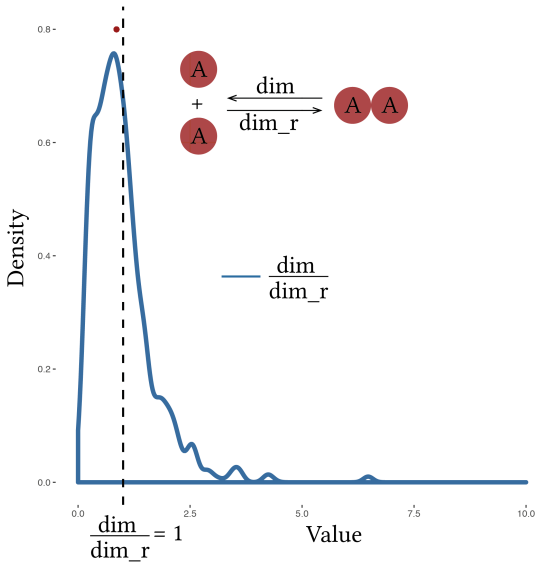

Figure 4.15 : The QSSA that the dimerization reaction (dim) is much faster than its reverse (dim_r) cannot be justified here. The median of the data (red) lies below the x=1 line (black dashed line) which indicates that in the majority of the particles in the posterior $\frac{dim}{dim\_r} < 1$

4.6.3.1 Multistability in the MA switches

To investigate how the level of abstraction affects switch design principles, I expand the analysis under the assumption of mass action kinetics and stochastic dynamics. The asymmetric CS-MA and DP-MA models are simulated using the Gillespie algorithm (Gillespie 1977).

Ma et al. (2012) found that the stochastic fluctuations in a system involving such a small number of molecules, like the toggle switch, uncovers effects that can not be predicted by the fully deterministic case (Ma et al. 2012). We find that in the stochastic case, both the simple switch, CS-MA , and positive autoregulation switch, DP-MA, are capable of both bistable and tristable behaviour. The fact that tristability can occur in the classical model is consistent with the effect of small molecule numbers; if gene expression remains low, it provides the opportunity for small number effects to be observed, and the third steady state to stabilise (Ma et al. 2012). In order to ensure that the tristable switches found in the stochastic case are truly tristable, I re-sample the posterior distributions and simulate to steady state. If the resulting phase plots are tristable then we know that the posterior truly represents tristability.

As can be seen in Figure 4.16, differences in the parameter values are observed between the bistable and tristable switches, in bothCS-MA and DP-MA models. We find that the simple switch is tristable when dimerisation rate is low and bistable when it is high. The degradation of the dimer proteins must have a low rate for bistability but there are no restraints in the case of the tristable switch. For the case of the DP switch, we find that the rates for dimerisation, degradation and dimer degradation are different for the bistable and tristable behaviours (Figure 4.16). The rate of dimerisation must be low for tristability to occur and large for bistability, as observed for the simple switch. The parameter for protein degradation must be low for tristability whereas there are no constraints for the bistable case. Finally, the parameter for dimer degradation must be low for bistability whereas it has no constraints for tristability, as observed in the simple switch. The design principles for both the CS-MA model and the DP-MA model are summarised in Table 4.4

Table 4.4 Design principles of bistable and tristable switches

	CS-MA		DP-MA	
	Bistable	*Tristable*	*Bistable*	*Tristable*
dimerisation	High	Low	High	Low
protein degradation	-	-	-	Low
dimer degradation	Low	-	Low	-

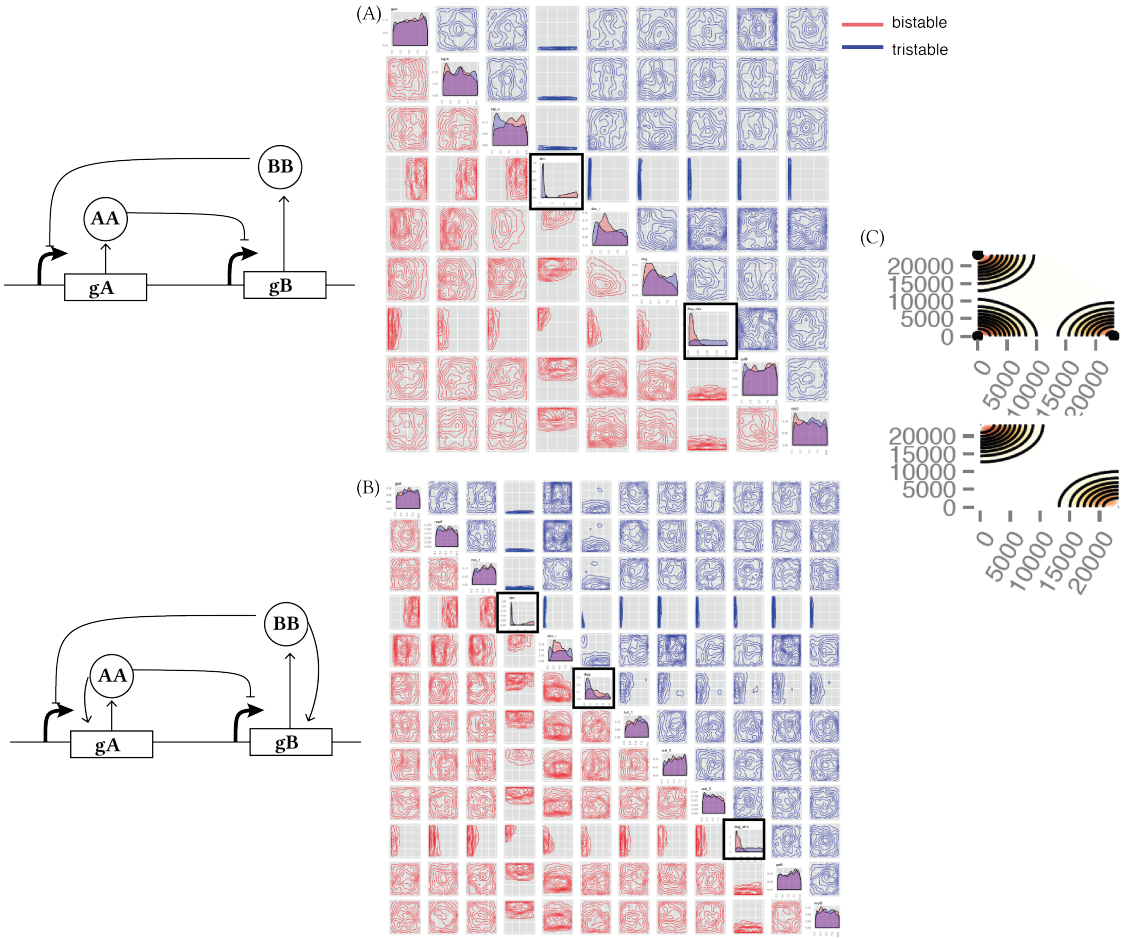

Figure 4.16 : Tristability is possible in the mass action toggle switch models only when simulated stochastically. (A) The simple toggle switch with no autoregulation can be both bistable and tristable. The two posteriors are shown, where the posterior distribution of the bistable switch is shown in red and of the tristable switch in blue. From the posterior distribution we can deduce the the dimerization parameter must be small for tristability to occur but large for bistability. The switch with double positive autoregulation and its posterior distributions for the bistable and tristable case are shown in (B). (C) A sample phase plot of a stochastic tristable and bistable mass action switch.

4.6.3.2 Robustness prior dependence

An important aspect of robustness that I must address is its dependence on the prior distributions. From Equation 2.14 we can expect that the measure for robustness will depend on the size of the prior.

I use the symmetric CS-MA model and Stability Finder to find the posterior distribution that makes this model bistable. This is repeated using the same model, but with larger prior ranges. The priors used are shown in Table 4.5. The posterior distributions are shown in Figure 4.17A and B. We find that for this model, if the prior ranges are very large, the robustness of the model will be much larger. If one of the parameters is able to have a larger value, then the constraints on the rest of the parameters are not necessary any more.

In order to test this observation, I test each parameter separately. All the priors are kept the same as the priors in the very narrow case (Figure 4.17A), except for one. Each run, one of the parameters has priors equal to the priors used in the wide case, Figure 4.17B. The priors are summarised in Table 4.5. Every time the robustness is calculated using the ellipsoid method. Since robustness is only meaningful when used in relation to another model, the robustness of the CS-MA model is compared to the robustness of the DP-MA model used in Figure 4.14. The same posterior distribution of the DP-MA model as used in Figure 4.14 are used every time. For the parameters this model shares with the CS-MA model I used the priors used in the narrow case in Table 4.5, for their shared parameters.

Table 4.5 Priors used for studying the effect of priors to robustness

	Very narrow	Narrow	Wide
ge	5 - 10	1 - 10	1 - 100
rep	2 - 10	1 - 10	1 - 100
rep_r	0 - 5	1 - 10	0 - 10
dim	5 - 10	1 - 10	0 - 10
dim_r	0 - 5	0 - 5	0 - 10
deg	1 - 5	0 - 10	0 - 100
deg_dim	0 - 0.1	0 - 0.5	0 - 1

We can see from Figure 4.17C that when the priors for *ge* are much larger, the Bayes' factor increases significantly. This is due to the fact that when the rest of the parameters are constrained to being within a very narrow range, gene expression must be small for bistability to occur. This has the effect of lower robustness since the prior volume added to the system is much larger than the volume of the

functional region, thus greatly decreasing the robustness.

Robustness of the CS-MA model is increased when the priors for degradation and dimerisation have wider priors (Figure 4.17C). Therefore, if degradation and dimerisation can take larger values, the system becomes more robust.

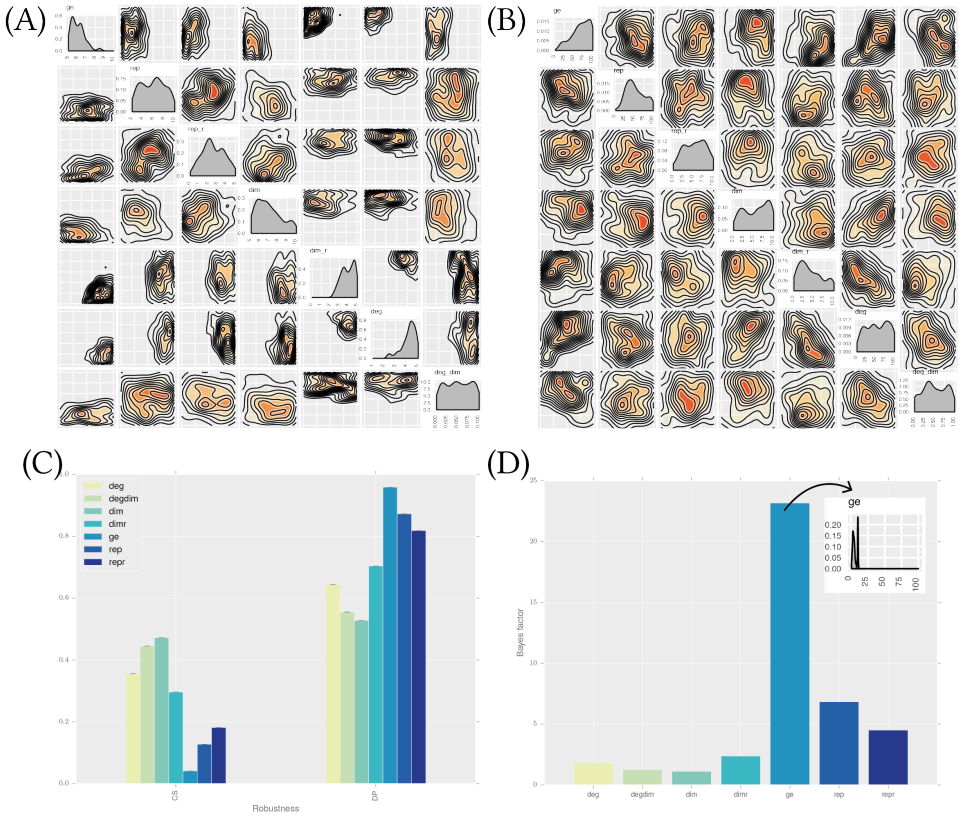

Figure 4.17 : The volume of the priors has an effect on the posterior distribution obtained. (A) The CS-MA model with narrow priors and (B) the CS-MA with wide priors. (C) The increase in robustness seen is due to the gene expression parameter being able to be large.

In Section 4.6.3 we found that the DP-MA is more robust than the CS-MA model. Here I want to test whether this result still remains when the priors of the models are made wider. I change the prior ranges of both models and measure their robustness each time. The results are shown in Figure 4.18A. We find that the robustness measure changed significantly as the priors of the models changed. When both models have very narrow or when both models have wide priors then their robustness measures are very similar. When both models have vary narrow priors the Bayes' factor is equal to 1.32 and when the priors are wide the Bayes' factor is equal to 1.06. In both these cases there is no significant difference in robustness between the two models. When the priors for both models are narrow the Bayes' factor is equal to 2.25. Most notably, when the priors for the CS-MA model are very narrow and the priors of the DP-MA model are narrow, the Bayes' factor is at 9.14.

It is evident that the robustness measure depends on the prior volume. It is therefore useful to think of the Bayes' factor in terms of the difference in the volume of the priors of the models that are being compared. I carry out this analysis for the above priors and the results are shown in Figure 4.18B. Here we see that even though the prior difference is within the same order of magnitude, the Bayes' factor increases significantly. This point corresponds to the case where the priors of CS-MA are very narrow and the priors of DP-MA are narrow, and there is where we see the Bayes' factor between the two models maximise. We can take advantage of this observation to design a switch with double positive autoregulation that is significantly more robust than the Gardner toggle switch. By choosing the parameter values carefully we can maximise the gain of robustness that adding two positive feedback loops gives.

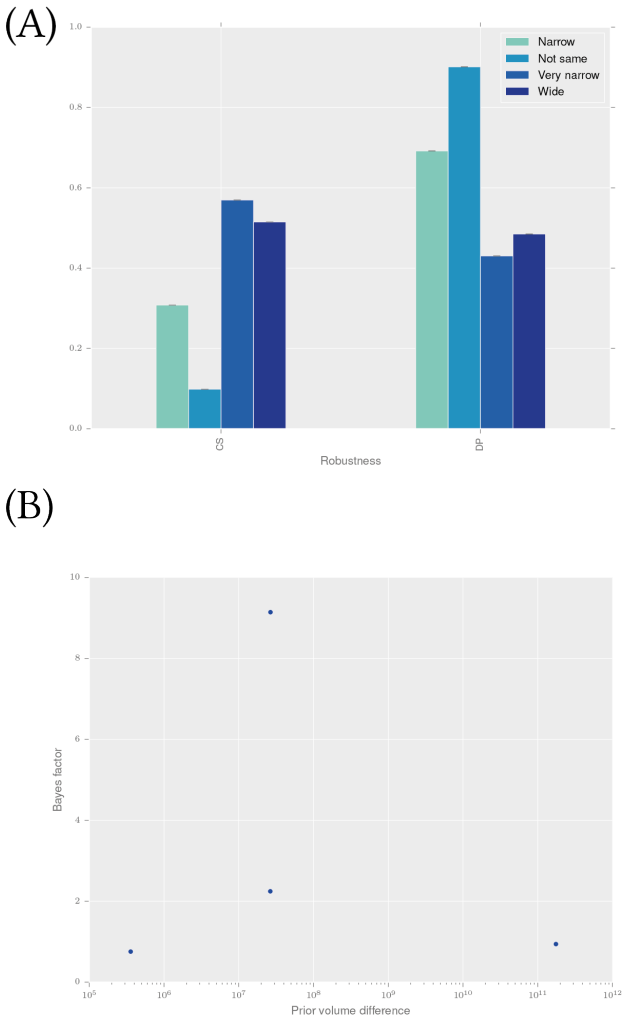

Figure 4.18 : Changing the priors in both models affects the robustness measure. (A) Using different prior ranges for the CS-MA and the DP-MA models yields different robustness for each. (B) The Bayes' factor as a function of the prior volume difference.

4.7 Discussion

Here I developed a novel framework, Stability Finder, that can be used to infer parameter values that can produce a desired behaviour. The novelty in the framework I developed over existing methodology is that complex models can be analyzed assuming both deterministic and stochastic dynamics. I have used Stability Finder to uncover the design principles of a bistable, a tristable and a quadristable switch. I found key parameters that are important in determining the number of steady states a model is capable of. This is critical to the design of novel synthetic switches, where the stability of a system has to be well defined and predictable. A bistable, a tristable or a quadristable switch could each be used for different functions within a synthetic system. Being able to predetermine the stability a system is capable of is vital for the design of new systems. This is especially true when multiple such systems are used together, and the success of the whole system depends on all the parts working as expected.

Tools that can identify parameter regions that give rise to specific behaviours will be key for the success of synthetic biology. In the future, by selecting the system components accordingly, the parameter values can be adjusted *in vivo*. For example, the desired level of gene expression can be accomplished by selecting the appropriate RBS sequence (Salis, Mirsky, & Voigt 2009). Another method to modify the parameter values *in vivo* is to select the promoter to have the strength corresponding to the levels of gene expression and repression desired. Activity of each promoter can be measured and standardised (Kelly et al. 2009) making this process possible. For a system requiring more than one promoter, these can be efficiently selected from a promoter library using a genetic algorithm (Wu, Lee, & Chen 2011). These standardised interchangeable components with known sequence and activity (Kelly et al. 2009; Canton, Labno, & Endy 2008) can be selected and used to construct a desired system and replicate the parameter values found using Stability Finder.

The methodology used here can only be used to study the presence of a given stability and not its absence. If the algorithm is not converging it cannot be concluded that the given model is not capable of the desired stability under these priors. For example, the mass action switches were found to be both bistable and tristable when stochastic effects were taken into account. Using deterministic dynamics the algorithm did not converge using priors within the ranges used in this work. Nevertheless this does not permit the conclusion of absence of tristability in the deterministic classic or double positive mass action switches. The methodology

presented here only permits the interpretation of models that have converged to a given stability.s

The methodology presented here can also be used to study the topology of more complex multistable switches that exist in natural biological systems such as developmental pathways. I also limited this framework to the objective behaviour of a given number of stable steady states. This could be extended to examine systems with a given switching rate or systems robust to a particular set of perturbations, both of which could be of great importance for building more complex genetic circuits.

Importantly I find that the prior distributions used during such an analysis greatly affect the robustness observed. More generally, the assumptions made when building a model can have a significant effect on the predictions made. This is consistent with current understanding (Babtie, Kirk, & Stumpf 2014) and highlight the importance of a programme of experimental work, combined with systems modelling, in order to understand the rules of thumb for abstraction in model based design of synthetic biological systems.

4.8 Summary

In this chapter I discussed the algorithm I developed and demonstrated how it can identify the parameter regions necessary for a model to achieve a given number of stable steady states. I used it to uncover the underlying principles that govern the stability of a given switch.

I first tested Stability Finder on a known switch and then proceeded to apply it to more complex models. I uncovered the design principles that make the Lu switch bistable, tristable or quadristable. I extended the Lu models to a three-node switch and showed how it can achieve 6 steady states.

Furthermore, I built two novel models of the toggle switch which do not use the QSSA and showed that the QSSA cannot be justified in these models. Using these models I studied the effect positive autoregulation has on the robustness of a model. I also studied the effect the priors have on the posteriors and on the robustness of a model. Finally, using stochastic modelling I showed that these switch models are capable of both bistable and tristable behaviour.

In the next chapter I study the genetic toggle switch in the lab and fit the toggle switch models used here to experimental data.

5 Bayesian model fitting applied to flow cytometry data

5.1 Introduction

In this chapter I aim to fit the toggle switch model to experimental data. This chapter is organised as follows: In the first section I provide an overview of the framework developed to fit models to flow cytometry data (ABC-Flow). In the subsequent section I test ABC-Flow on simulated flow cytometry data. Next I use flow cytometry to study the toggle switch experimentally and examine the concentrations of the inducers and the time needed to flip the switch. Finally, I use ABC-Flow to fit a computational model to the experimental data acquired.

5.2 Contributions to this Chapter

The R code used to pre-process the flow cytometry obtained was provided by Alex J Fedorec. The initial development of ABC-Flow was carried out by Chris P Barnes.

5.3 Flow cytometry and model fitting

Computational modelling is well known to aid the understanding of complex systems by fitting experimental data and providing further insights and testable predictions. Experimental data is used to fit the model parameters and then the model can provide further understanding of the system and aid in the design of further experiments. Flow cytometry is used in synthetic biology for BioBrick characterisation (Kelly et al. 2009), enzyme screening (Choi et al. 2014) and industrial bioprocesses (Díaz et al. 2010) among others.

Flow cytometry data presents a challenge to computational modelling as the fluorescence intensity per cell is measured rather than number of proteins. The problem with measuring fluorescence intensity is that it is a relative and not an absolute measurement like the number or concentration of proteins in a system, which would increase the predictive power of computational models (Bower, McClintock, & Fong 2010; Cooling et al. 2010), but this type of biological data cannot be directly measured (Kelwick et al. 2014). The fluorescence intensity values can vary between experiments due to instrument settings so they can only be used in relative terms within the same experiment. Standardization of experimental methods in flow cytometry has aided the effort to reduce variability due to experimental setup (Kelly et al. 2009), but the successful conversion of fluorescence intensity to an absolute measurement of protein $cell^{-1}$ s^{-1} has yet to be made successfully.

Another approach to the problem is converting the model output of GFP $cell^{-1}$ s^{-1} to relative fluorescence intensity. This approach was first developed by Lillacci & Khammash (2013). The converted model output can then be compared to the data output from the flow cytometer. The fluorescent intensity measurements acquired via flow cytometry are treated as a sample from distribution of the fluorescence present in the cell (Lillacci & Khammash 2013). This means that the flow cytometry fluorescence distribution at each time point can be compared to the model fluorescence distribution. Here I expand the method developed by Lillacci & Khammash (2013) in order to be able to apply it to flow cytometry data including two fluorescent proteins simultaneously. This new framework, ABC-Flow, can be used to fit stochastic models to flow cytometry data involving two species, like the genetic toggle switch.

5.4 ABC-Flow algorithm development

The algorithm of ABC-Flow is based on the same ABC algorithm as ABC-SysBio and Stability Finder described in Sections (XXX), adapted to be used for flow cytometry data. The algorithm of ABC-Flow is outlined in Algorithm 6. The modified modules of the ABC algorithm are outlined in the sections that follow.

The user provides an SBML model file and an input file to specify the information needed to run ABC-Flow, such as the epsilon schedule and the priors to the parameters. The user must also provide a data file containing the flow cytometry data to which the model will be fitted. The data files used here were generated from .fcs files, which is the standard output of flow cytometers, using the R bioconductor

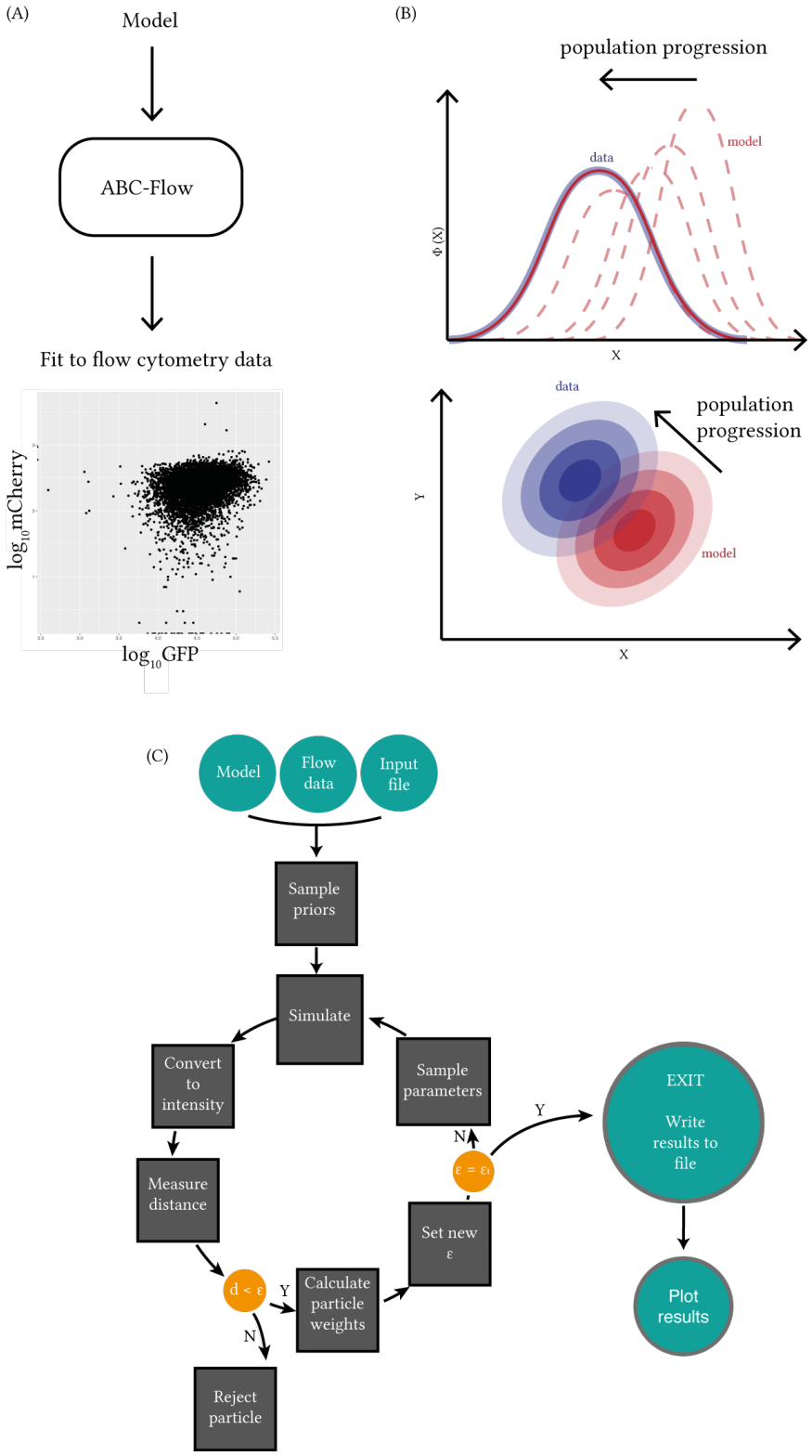

Figure 5.1 Overview of ABC-Flow. (A) ABC-Flow is used to fit models to experimental flow cytometry data. (B) The algorithm can be applied to 1D and 2D flow data. (C) ABC-Flow uses Approximate Bayesian Computation.

packages flowCore (Ellis et al. 2016b). All models are simulated stochastically using the Gillespie algorithm (Gillespie 1977). ABC-Flow simulations are implemented on GPUs. ABC-Flow is available as a Python package, and can be downloaded from https://github.com/ucl-cssb/ABC-Flow.git.

Algorithm 6 ABC-Flow

1: Initialise ϵ
2: population p $\leftarrow 1$
3: **if** p $= 1$ **then**
4: Sample particles (θ) from priors
5: **else**
6: Sample particles from previous population
7: Perturb each particle by $\pm$ half the range of the previous population (j) to obtain new perturbed population (i).
8: **end if**
9: Simulate model using the Gillespie algorithm.
10: Convert signal to intensity:
11: **for** each particle **do**
12: **for** each beta **do**
13: **for** each timepoint **do**
14: **for** each fluorescent protein **do**
15: Intensity = $\mathrm{N}\left(\text{signal}\times\mu, \sqrt{(signal \times \sigma^2)}\right)$
16: **end for**
17: **end for**
18: **end for**
19: **end for**
20: Measure distance to data
21: Reject particles if d $> \epsilon$.
22: Calculate weight for each accepted θ
23: $w_t^{(i)} = \begin{cases} 1, & \text{if } p = 0 \\ \frac{\pi(\theta_t^{(i)})}{\sum_{j=1}^{N} w_{t-1}^{(j)} K_t(\theta_{t-1}^{(j)}, \theta_t^{(i)})}, & \text{if } p \geq 0. \end{cases}$
24: Normalise weights
25: Repeat steps 3 - 15 until $\epsilon \leq \epsilon_T$

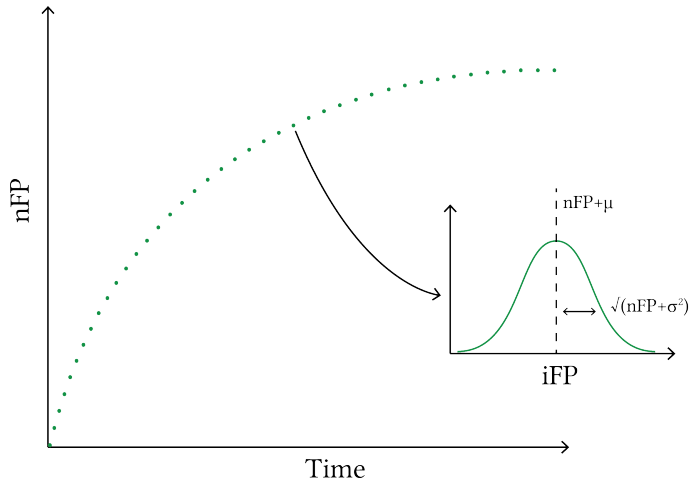

Figure 5.2 Converting the number of fluorescent proteins to the intensity (iFP) is done by drawing from a normal distribution, as shown in Equation 5.1.

5.4.1 Intensity calculation

The units of the result of the stochastic simulations is in the form of number of fluorescent proteins. On the other hand, flow cytometry data units are in the form of fluorescence intensity. For ABC-Flow, the simulation results are converted to intensity in order to be able to compare the data to the simulations. In order to do this two additional parameters are defined, intensity μ and intensity σ, for each fluorescent protein used. To convert the number of fluorescent proteins to intensity, random samples are drawn from a normal distribution:

$$X \sim N\Big(nFP \times \mu, \sqrt{(nFP \times \sigma^2)}\Big), \tag{5.1}$$

where nFP is the number of fluorescent proteins.

These parameters are fitted to the data along with the model parameters.

5.4.2 Distance Calculations

In order to compare the flow cytometry data to the model generated data, I had to develop a distance measure. This distance measure should be able to determine whether two datasets are sufficiently close to each other to be able to be assume that they have been drawn from the same distribution. The measure should also give an estimate of how different the two data sets are, and thus get increasingly

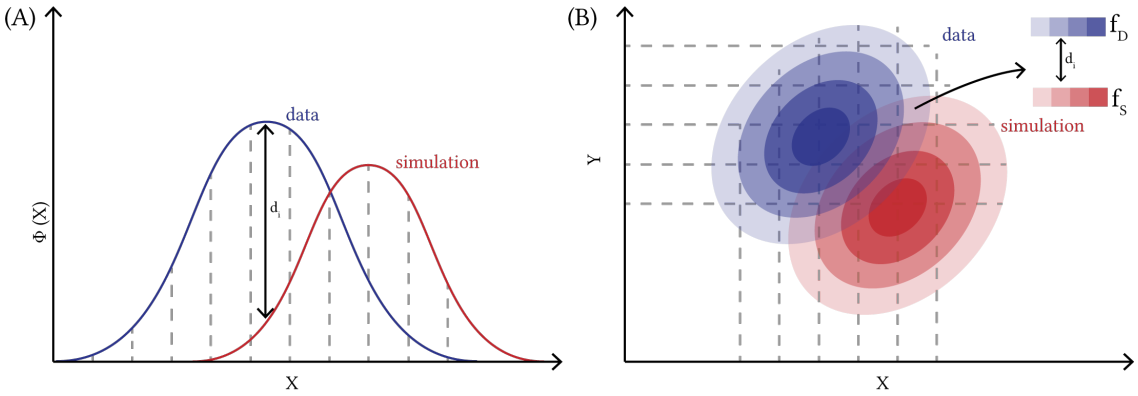

Figure 5.3 Calculating the distance between two distributions in (A) 1D and (B) 2D.

larger as two data sets are drawn from increasingly different distributions. Finally the distance measure should be applicable to one and two dimensional distributions, and be comparable between the two.

5.4.2.1 Kernel distance

In order to measure the distance between the flow cytometry data and the fitted model, the algorithm outlined in Algorithm 7 was developed. The algorithm consists of defining a grid from the minimum to the maximum value of the data. A gaussian kernel was then fit to the flow and simulated data. The distance between the two kernels is given by:

$$d = \sum_{i=xmin}^{xmax} (fD_i - fS_i)^2,$$

where fD_i is the kernel of the flow data at each value of x and fS_i the kernel of the simulaed data. An illustration of the distance calculation is shown in Figure 5.3.

Algorithm 7 Distance calculation

1: Grid ← min(data):max(data):ngrid
2: kD = kernel density estimation(data)
3: kS = kernel density estimation(simulations)
4: fD = kD(xx)
5: fS = kS(xx)
6: $epsilon = \sum((fD - fS)^2)$

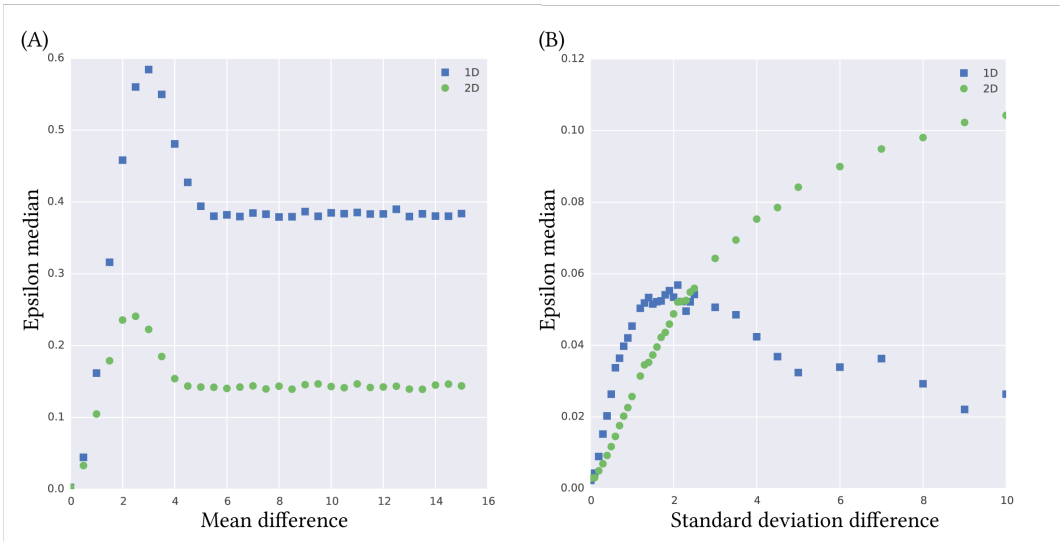

Figure 5.4 (A) The range by which epsilon varies as the difference between the mean of the distributions increases. (B) The median of the epsilon distributions varies by a small amount with increasing difference in the standard deviation of the distributions.

Prior to incorporating this distance calculation in ABC-FLow, it was tested to determine whether it is an appropriate distance to use when comparing distributions. This was done by drawing samples form two uniform distributions with varying mean and standard deviation. Algorithm 7 was then used to calculate the distance between the different distributions.

First, Algorithm 7 was tested by drawing samples from two distributions with an increasingly different mean. This is done to determine the dynamical range of the distance calculation.

From Figure 5.4 we see that the epsilon value does not increase linearly with increasing mean difference of the two distributions. As the difference between the means increases, the epsilon value reaches a peak when the difference is at 3. From that point, as the mean difference increases, epsilon values decrease until they reach a plateau at epsilon = 0.38 in the 1D case and epsilon = 0.14 in the 2D case. Next, I test the distance calculation by comparing bimodal distributions. Two bimodal distributions are generated with increasingly different mean, in 1D and 2D.

Similar to the normal distribution, for the bimodal distributions shown in Figure 5.5 we find that the epsilon values do not increase linearly. There are two peaks

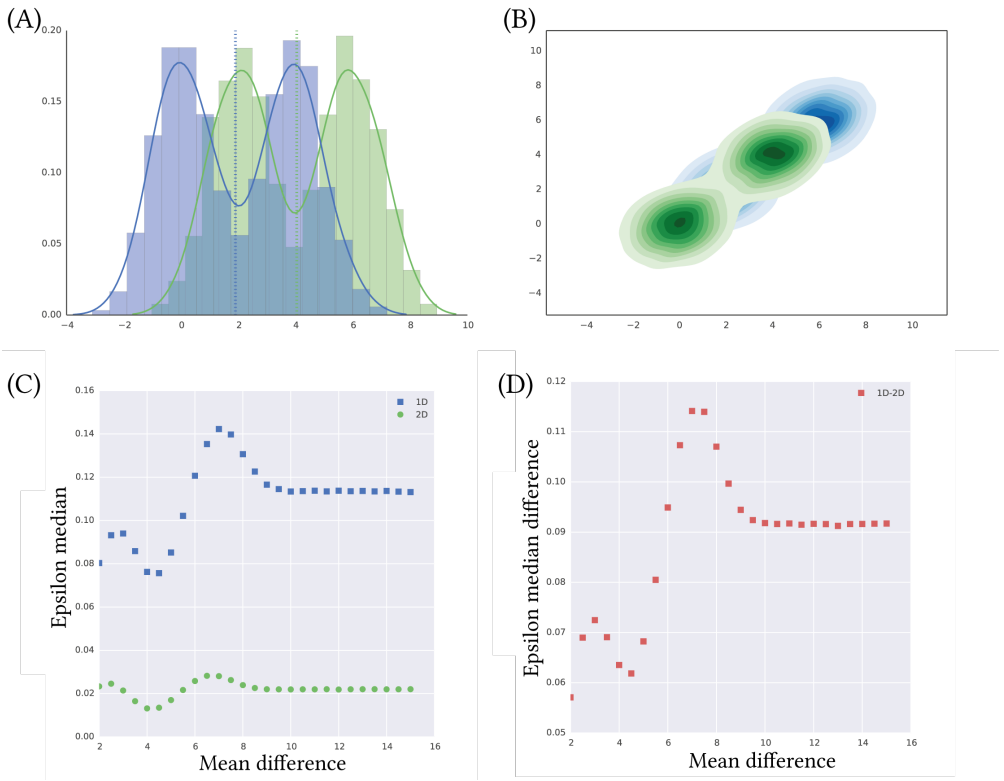

Figure 5.5 Comparing the 1D and 2D distances between bimodal distributions. (A) and (B) show samples of the bimodal distributions compared in 1D and 2D respectively with a mean difference of 4 between simulations and data. (C) The range by which epsilon median and variance varies as the difference between the mean of the distributions increases. (D) The difference between the epsilons calculated in 1D and 2D is not constant.

in the epsilon distribution, one at mean difference = 3 and one at mean difference=6. The epsilon values then decline until they reach a plateau. The epsilon values do not have a large range of values, for neither the 1D or 2D cases. We also find that the difference in the epsilon values between the 1D and 2D cases is not constant.

Finally, I study how these distance functions perform when comparing a bimodal with a normal distribution. A bimodal distribution is generated and a series of normal distributions with increasing mean, in 1D and 2D. From Figure 5.6 we find that epsilon is the lowest when the mean of the normal distribution corresponds to the μ of one of the two peaks in the bimodal distribution and the highest when there is no overlap between the distributions.

From Figures 5.4-5.6 I conclude that Algorithm 7 is not a good measure for distance to be used in ABC-Flow. If Algorithm 7 was used in order to minimize the

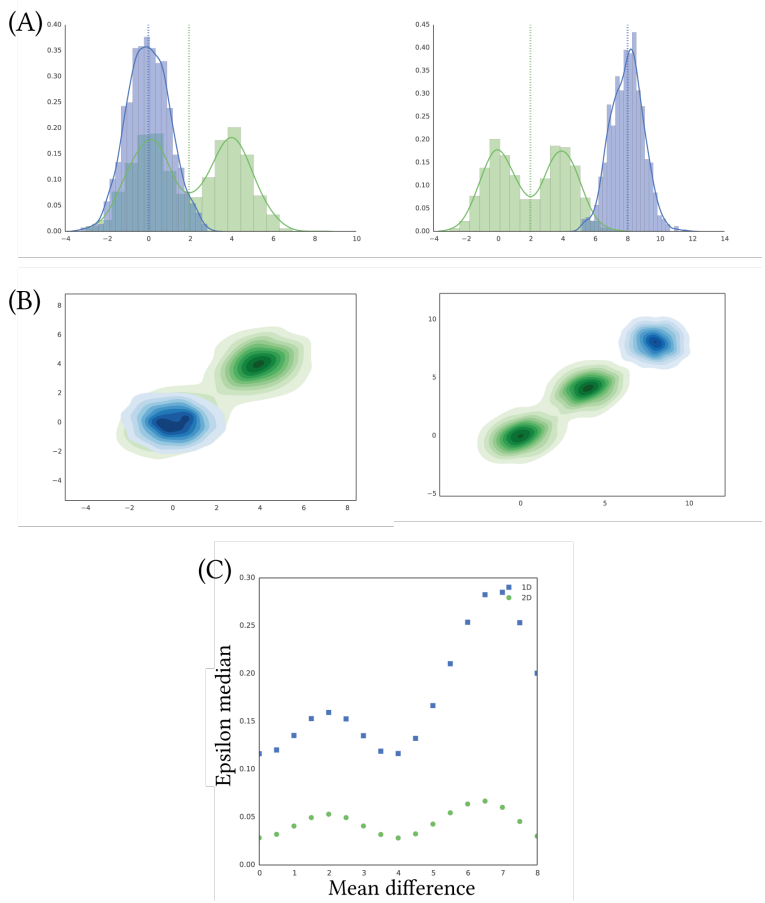

Figure 5.6 Comparing a multimodal to a normal distribution, in 1D and 2D. (A, B) The mean of the normal distribution is varied from equal to the mean of the first peak of the bimodal distribution to beyond the range of the bimodal distribution. (C) Epsilon median and variance are at the lowest when the mean of the normal distribution is equal to the mean of one of the peaks of the bimodal distribution.

distance between two distributions that start off with very different means, the distance between the two distributions will not be sufficiently minimized. This stems from the fact that ABC-FLow works by iteratively making the accepted epsilon smaller. As can be seen in Figure 5.4, if the two distributions have a large difference in the means, (>6) it would not be possible to overcome the peak that is created when the mean difference is at 3. Epsilon values increase before the decrease again, which will be a problem in ABC-Flow. Therefore a different distance calculation was developed.

5.4.2.2 Kolmogorov-Smirnov distance

In order to avoid the problems that arose from the distance calculation described in Section 5.4.2.1 I implemented a different distance calculation for ABC-Flow. I used a Python implementation of the Kolmogorov-Smirnov two sample test for the 1D case (Kolmogorov 1933). The Kolmogorov-Smirnov (KS) test is a non-parametric statistic test that determines whether two data sets were frawn from the same underlying distributions. The KS distance between two distributions is equal to the largest distance between the empirical distribution functions of the two samples, as shown in Equation 5.2.

$$D_{n,n'} = \sup_x |F_{1,n}(x) - F_{2,n'}(x)| \tag{5.2}$$

For the 2D case the distance was calculated by using the 2D Kolmogorov-Smirnov two sample test. The algorithm was developed by Fasano & Franceschini (1987) and the Python implementation developed by Major (2016).

This distance calculation was tested to determine whether it is an appropriate distance function to use in ABC-Flow. Two datasets were drawn from normal distributions with increasingly different means. The KS test was then used to calculate the distance between the data sets. This was carried out in 1D and 2D. The results are shown in Figure 5.7.

The epsilons of the 1D Kolmogorov-Smirnov distance calculation increase with increasing mean difference until it reaches a plateau when the two distributions are very different. This makes it an ideal distance calculation to be used in ABC-Flow. As the epsilon threshold is lowered at each iteration the difference between the two data sets decreases. Therefore the 1D KS statistic was used in ABC-Flow.

The multidimensional Kolmogorov-Smirnov test presents a challenge, as there is no unique way to order the data points to calculate the largest distance. There are $2^d - 1$ ways of ordering the data points and defining a cumulative distribution function, where d is the number of dimensions (Lopes, Reid, & Hobson 2007). This has affected the results of the computed distance seen in Figure 5.7. The variability in the calculation of the distance between data sets originating from distributions with known distance is large relative to the range of values the calculation can take. This was further confirmed when testing this distance on simulated data, where no parameter identifiability was observed (data not shown).

To alleviate the above shortcomings of the multi-dimensional generalisation of the Kolmogorov-Smirnov test, a different distance calculation was used for the 2D

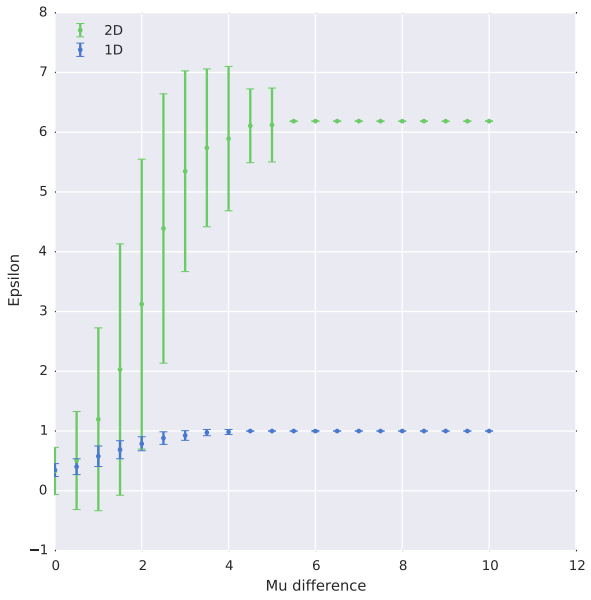

Figure 5.7 The Kolmogorov-Smirnov distance function was tested in 1D (blue) and 2D (green). Two data sets were generated with increasing mean difference, and the Kolmogorov-Smirnov two-sample test was applied to compute the distance between the two.

case. The Kolmogorov-Smirnov test for the 1D case was used in ABC-Flow as it performed well in the testing shown here.

5.4.2.3 Wald-Wolfowitz distance

For the 2D case the distance was calculated by using the multivariate Wald-Wolfowitz test (Friedman & Rafsky 1979). This is a generalisation of the Wald-Wolfowitz test proposed by (Wald & Wolfowitz 1940), a non-parametric test to determine whether two data sets were drawn from the same distribution. This test works by computing the minimum spanning tree of the pooled samples. Any edge whose nodes originated from different samples are removed, and the number of *runs* (R) is then defined by the number of disjointed subtrees (Friedman & Rafsky 1979). If the number of *runs* is small, then the null hypothesis that the two samples originated from the same distribution cannot be rejected. The quantity W for two samples, of length m and n, computed is given by:

$$W = \frac{R - 2\frac{mn}{N} - 1}{\sqrt{\frac{2mn(2mn-N)}{N^2(N-1)}}}, \tag{5.3}$$

where $N = m + n$ and R is the number of *runs*. A Python implementation of the multivariate Wald-Wolfowitz test by Monaco (2014) was used here. This is a variation to the Wald-Wolfowitz test that can be efficiently applied to larger data sets. The Python code used is given in Appendix (XXX).

Here I test this distance calculation in a similar way as Section 5.4.2.1. First, the two data sets are drawn from increasingly different distributions, and the distance between them calculated. As shown in Figure 5.8D, the 2D distance is 0 when the difference between the μ from which the two datasets are drawn from the same distribution. The distance calculation reaches a plateau at epsilon = 140 when the mean difference is 4 or larger. The 1D distance is also shown in Figure 5.8C in order to compare the two calculations, but the 1D distance was computed using the Kolmogorov-Smirnov distance described in Section 5.4.2.2.

To further study the distance calculation used in ABC-Flow, two normal distributions were simulated, with $\mu = 0$ and $\sigma = 1$ and distance between them calculated using the Kolmogorov-Smirnov test in the 1D case and the Wald-Wolfowitz test in the 2D case. Doing this multiple times, the expected variation in distance values for identical distributions can be calculated. This is the error that can be expected when measuring distance in ABC-Flow. As can be seen in Figure 5.9, the range of distance values obtained in the 1D case is small. For the 2D case, the distance values obtained vary more than in the 1D case, but it is still small relative to the range of values that the Wald-Wolfowitz test can take shown in Figure 5.8.

Using the Wald-Wolfowitz test the value of epsilon increases with increasing distance between the distributions with relatively small variability between repeats. Since the 2D Wald-Wolfowitz test performed well in the test carried out above, it was implemented in ABC-Flow as the distance function for the 2D calculations.

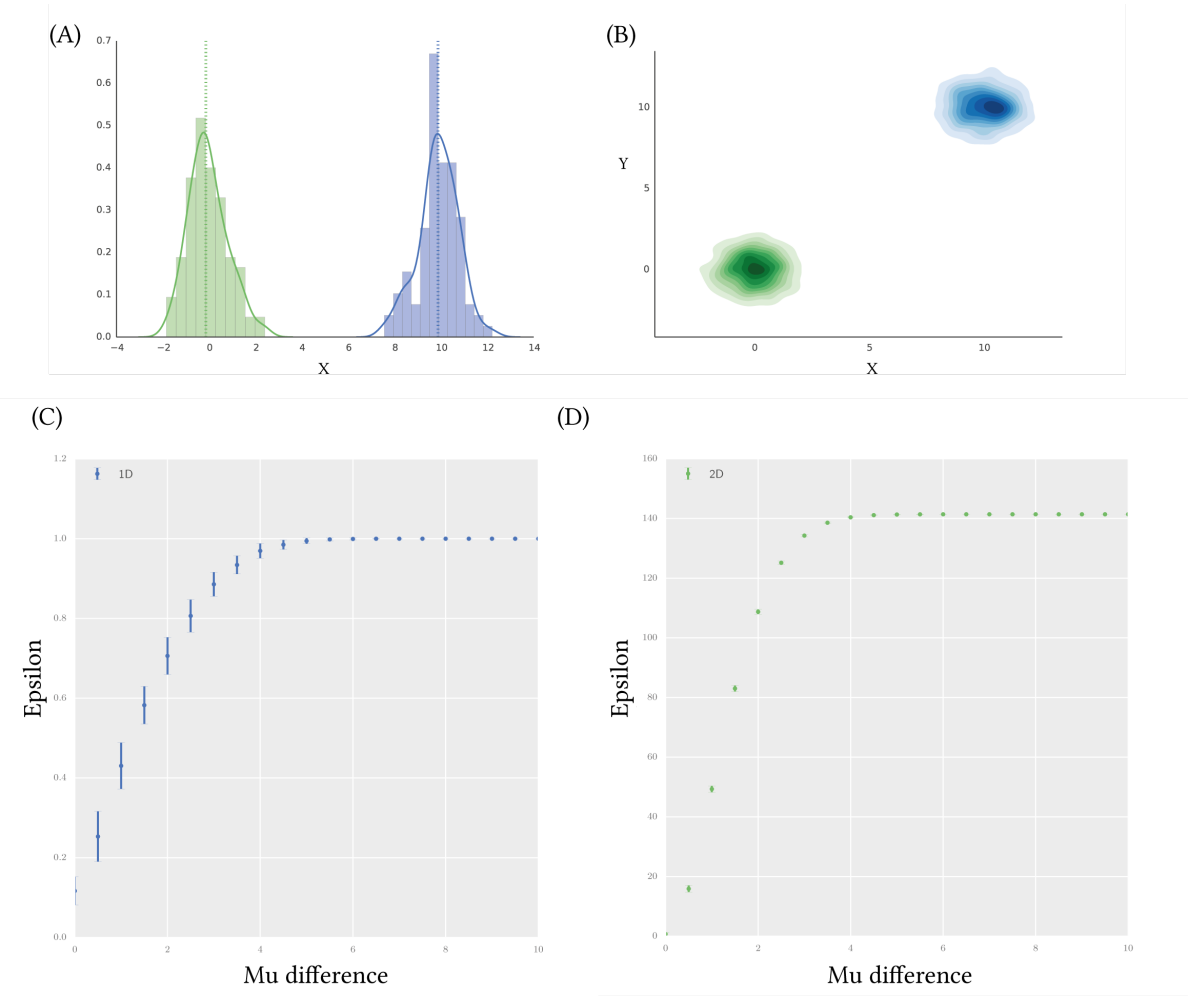

Figure 5.8 The distance calculation for data sets drawn from increasingly different distributions. Two examples are shown of distributions compared in (A) 1D and (B) 2D. (C) As the difference between the means of the two distributions increases, the distance calculation, epsilon, increases. In the 2D case (shown in green) epsilon plateaus at 2.3 and in the 1D case (shown in blue) epsilon plateaus at 1.

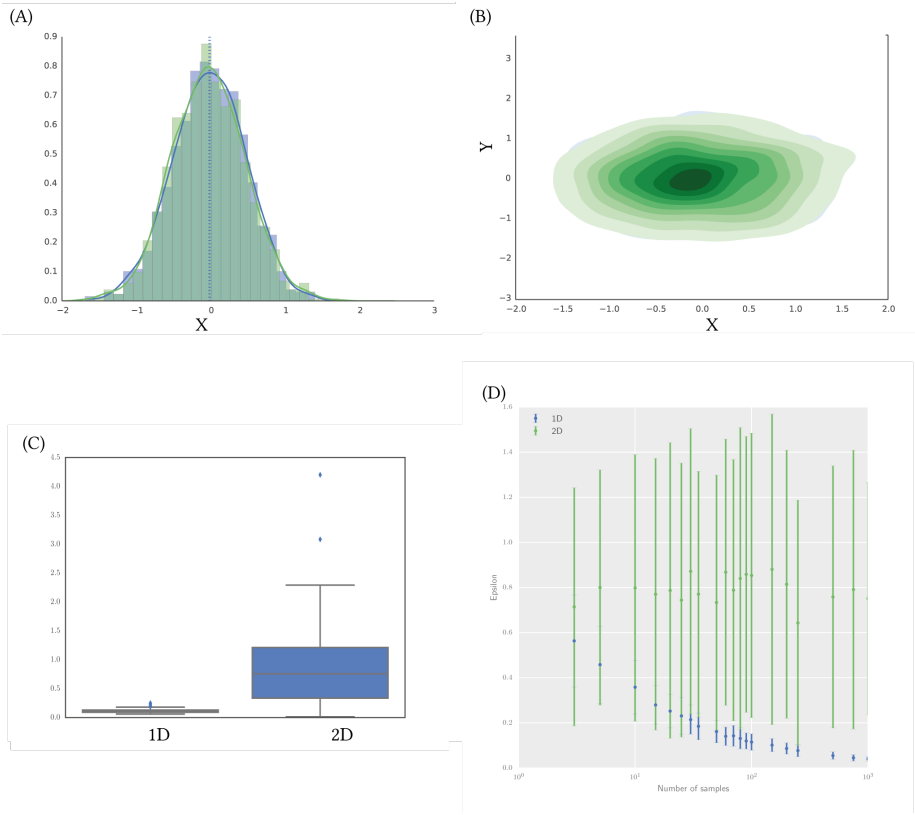

Figure 5.9 The distance between two data sets drawn form the same distribution are compared using the Kolmogorov-Smirnov two sample test. (A) in 1D and (B) in 2D. (C) The distance is calculated for 1000 data sets. A larger variation of values is found for the 2D distance calculation, but still small relative to the overall range of values. (D) As the number of samples in the datasets increase the distance calculation becomes more accurate in the 1D case. It has no effect on the 2D case.

5.5 ABC-Flow model fitting to simulated data

In this section I apply ABC-Flow to simulated data, where the parameter values used to produce the data are known. This analysis will serve as a verification test for ABC-Flow. The model used to produce the simulated data is an extension of the Gardner, Cantor, & Collins (2000) switch. The model consists of two mutually repressing transcription factors. The model used here has additional parameters allowing for gene expression to be leaky as well as include repression from an external stimulus.

In order to produce the simulated data set, an extension of the Gardner, Cantor, & Collins (2000) switch was simulated stochastically using the Gillespie algorithm (Gillespie 1977). The model used is defined by the following hazards:

$$h_1 = u \tag{5.4}$$

$$h_2 = \frac{p_1 \times p_3}{1 + p_3 + v^{p_2}} \tag{5.5}$$

$$h_3 = (1 + a) \times v \tag{5.6}$$

$$h_4 = \frac{p_4 \times p_6}{1 + p_6 + u^{p_5}}, \tag{5.7}$$

where u and v are the two proteins in the system, p_1 and p_4 represent the effective gene expression of u and v respectively, p_2 and p_5 represent the cooperativity of u and v respectively. p_3 and p_4 represent the leakiness of the promoters for each species. parameter α increases the degradation of one of the species, and simulates the addition of a repressor.

Using the time course data generated for one of the fluorescent proteins in the system, u, I use ABC-Flow to fit the model shown above, using priors centered around the parameter values used to produce the data, shown in Table 5.1. The resulting fit is shown in Figure 5.10A. In order to determine whether this is a good fit to the data, QQ plots are produced for each timepoint (Figure 5.10B). A QQ-plot is a plot where the quantiles of two distributions are plotted against eachother. If the distributions are similar, the points will lie on the 45° line x = y line (Wilk & Gnanadesikan 1968).

By examining the data and the fitted models shown in Figure 5.10, we see that at ε= 0.08, there is a good fit of the model to the data using ABC-Flow. The model parameters, as well as the intensity parameters, have been fitted to simulated flow cytometry data. The model has been successfully fitted to the simulated data. This is highlighted in the QQ plots in Figure 5.10B, where the results lie in the x=y line.

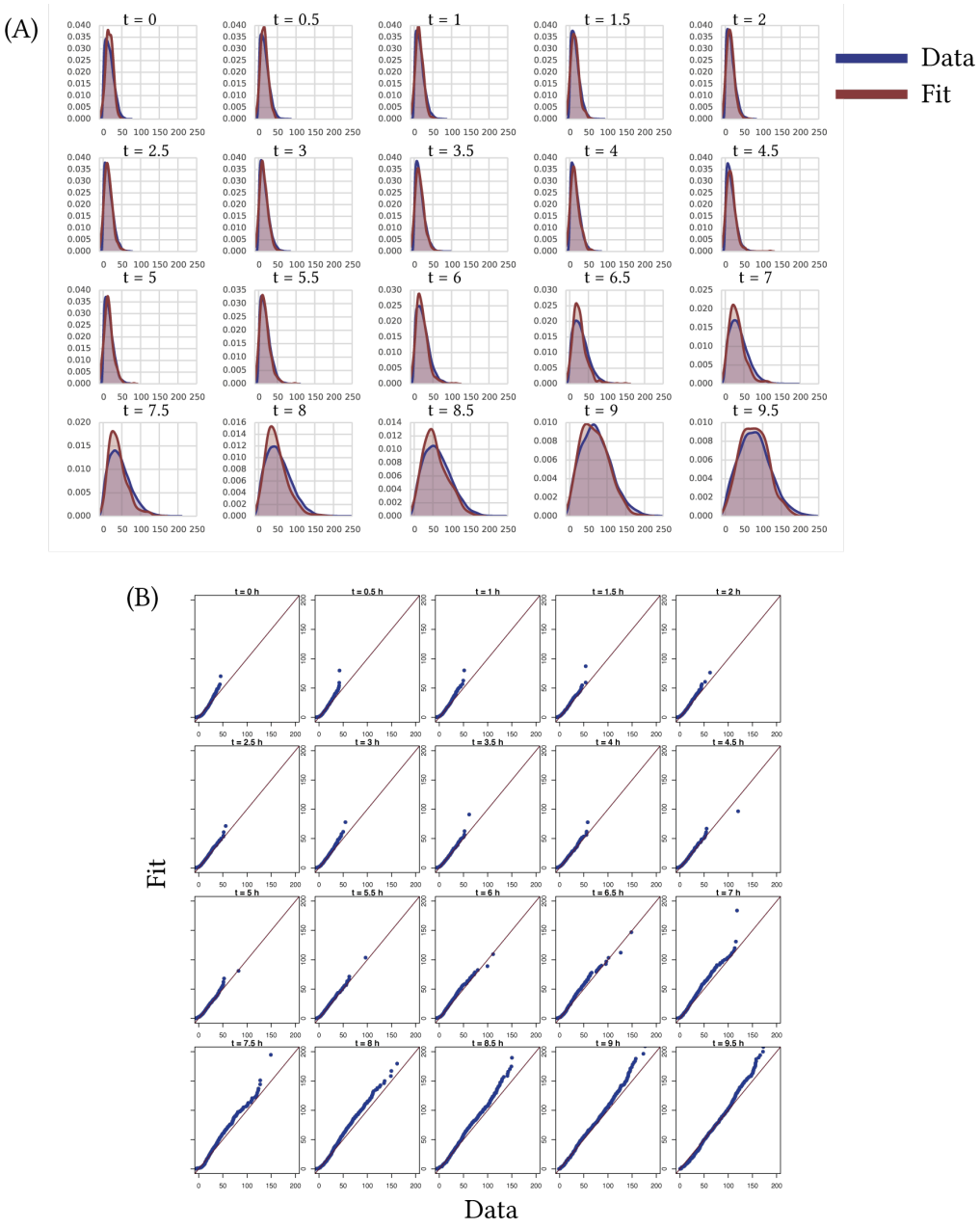

Figure 5.10 (A) 1D ABC-Flow fit (shown in blue) to data (shown in red) produced by simulating the same model. (B) QQ-plot of each time point fit. The quantile of the two distributions are plotted against eachother. If the ditributions are similar, the points would lie on the 45° line x = y, shown in red.

I further test ABC-Flow by using 2D data to fit two species of the model simultaneously. The same data was used as was used in the 1D case, but this time both species u and v were taken into account. This represents both sides of the switch model used.

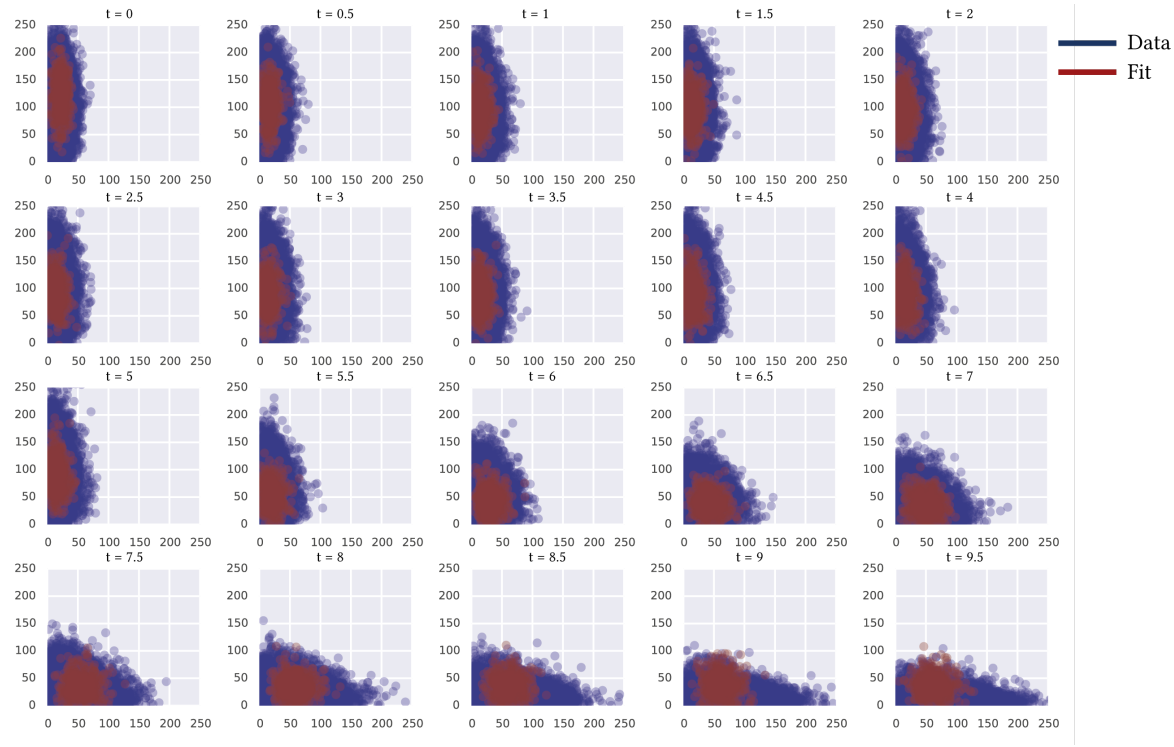

Figure 5.11 2D ABC-Flow fit (shown in blue) to data (shown in red) produced by simulating the same model.

The posterior distributions obtained from each fit are shown in Figure 5.12. We find similar posterior ranges for both the 1D and 2D fits. Both fits identified the parameters necessary to produce the simulated data. From the posteriors we find that the most constrained parameters to produce the switch behaviour in this model are parameters $p2$ and $p5$, the parameters representing the cooperativity of the repressors. They both have to be equal to 2 to produce the observed behaviour in this model. We also find that α, the parameter representing the increased degradation of the repressing species due to the addition of an inducer, is tightly constrained. α is required to be small. Further, we find that μ and σ, the parameters representing the mean and standard deviation of the fluorescence intensity emitted by each fluorescent molecule to be tightly constrained.

These results demonstrate that ABC-Flow can successfully fit a computational

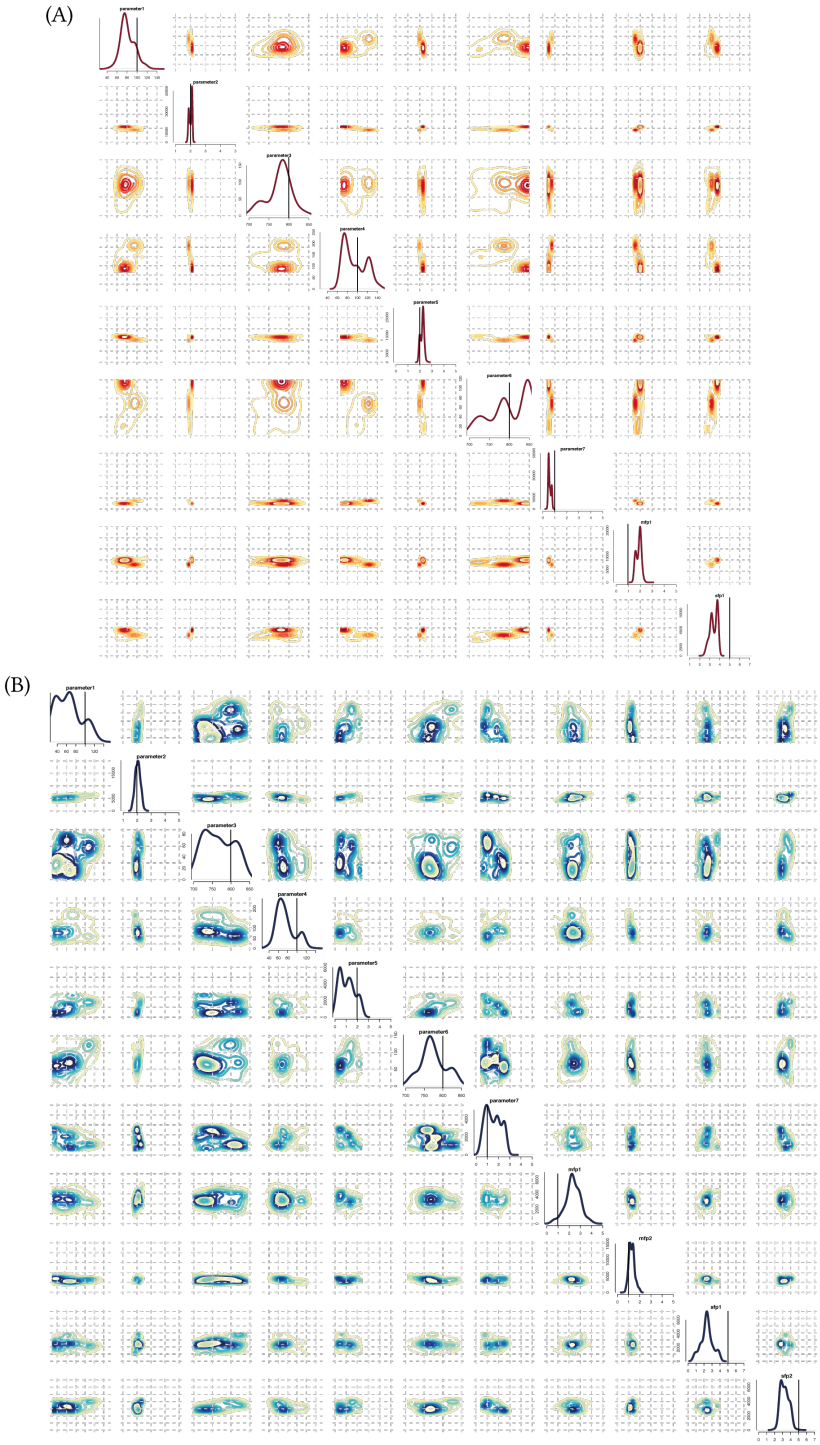

Figure 5.12 The posterior distributions of the 1D (red) 2D (blue) fits to simulated data.

Table 5.1 The priors used for the 1D and 2D ABC-FLow model fitting to simulated data

Parameters		
	1D	2D
p1	30 - 150	30 - 150
p2	1 - 5	1 - 5
p3	700 - 850	700 - 850
p4	30 - 150	30 - 150
p5	0 - 5	0 - 5
p6	700 - 850	700 - 850
p7	0 - 5	0 - 5
Species		
u	9 - 11	9 - 11
v	90 - 110	90 - 110
Intensity parameters		
mean fp1	0 - 5	0 - 5
mean fp2		0 - 5
sigma fp1	0 - 7	0 - 7
sigma fp2		0 - 7

model to flow cytometry data. It can identify the parameter values necessary to produce the observed behaviour. ABC-Flow can now be confidently applied to real flow cytometry data of the genetic toggle switch. This will allow me to fit a computational model to experimental data of the genetic toggle switch and uncover the parameters that are necessary to produce this behaviour. In the following Section I will outline the methods used to obtain the experimental data necessary and the results obtained.

5.6 Toggle switch data collection

In this section I collect experimental data on the genetic toggle switch. Using flow cytometry and the necessary inducers to flip the switch I study the switch flipping over time as well as over different inducer concentrations.

5.6.1 Circuit overview

The toggle switch plasmid I used here was provided by Litcofsky et al. (2012). All the switch components were contained in one plasmid, pKDL071. An overview of the plasmid is shown in Figure 5.13A and the sequence given in Appendix (XXX). The circuit consists of two promoters, Ptrc2 and PLtetO-1 (Lutz & Bujard 1997). Ptrc2 is a constitutive promoter, repressible by LacI. PLtetO-1 is also a constitutive promoter, repressible by TetR, as shown in Figure 5.13B. mCherry (Shaner et al. 2004) and GFP (Shimomura, Johnson, & Saiga 1962) are fluorescent proteins, that were added under the control of the same promoters as the repressors, and thus reflect the levels of TetR and LacI in the system. The plasmid contains kanamycin antibiotic resistance and is high copy (ColE1 origin of replication).

This system is capable of two states, GFP high and mCherry high. When IPTG is added to the system, it represses the repression of TetR and mCherry and thus the cells end up in the mCherry high state. When ATc is added to the system, it represses the repression of LacI and GFP and thus the cells end up in the GFP high state. If no inducer is added to the system it will randomly go to the GFP high or mCherry high states.

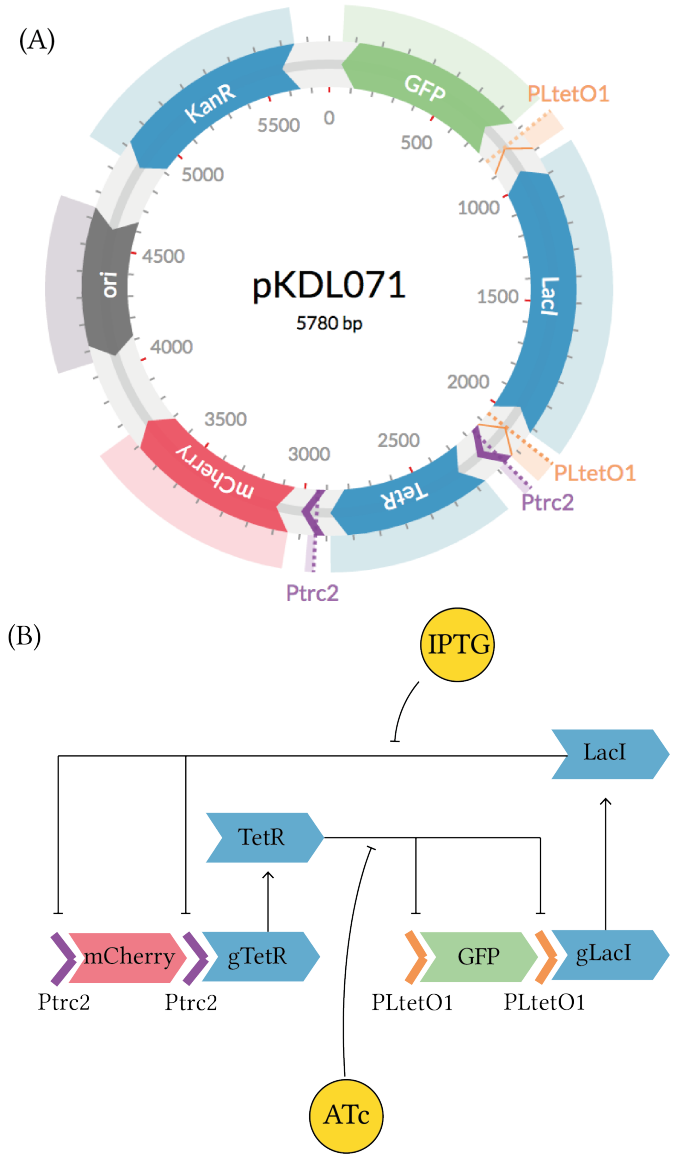

Figure 5.13 : The genetic toggle switch circuit used in this chapter. (A) The plasmid map of pKDL071, the plasmid containing the genetic toggle switch used in Litcofsky et al. (2012) (B) The interactions between each element of the circuit.

5.6.2 Methods

The toggle switch plasmid was provided by the James J Collins lab in the form of a stab culture in *E. coli* K-12 MG1655.

5.6.2.1 *Escherichia coli* culturing conditions

Lysogeny broth (LB) was made by diluting LB in deionized water to a concentration of $25\,\mathrm{g\,L^{-1}}$ and subsequently autoclaved. LB agar plates were made by adding bacteriological agar to the above solution to a concentration of $45\,\mathrm{mg\,mL^{-1}}$ before autoclaving. The solution was then cooled down to 55 °C using a water bath. If antibiotic was required it was added to the correct concentration to the cooled solution. The solution was then aliquoted to plates and left to solidify in room temperature. The plates were stored in the fridge for up to 1 month.

Overnight cultures were made by picking a single colony from a static culture in an agar plate. Each colony was placed in 15 mL Falcon tubes (Fisher Scientific, MA, U.S.A) with 5 mL LB with kanamycin antibiotic at a concentration of $50\,\mathrm{\mu g\,mL^{-1}}$. The tubes were then screwed loosely and taped securely in order to allow for aeration. The falcon tubes were put in an incubator at 37 °C with orbital shaking at 200 rpm for 12-16 hours.

5.6.2.2 Glycerol stock preparation

To preserve the transformed cultures long-term glycerol stocks were made. 5 mL LB and Kanamycin overnight cultures were made as described in Section 5.6.2.1. The cultures were kept on ice and 70 % glycerol was added to the cultures in a ratio of glycerol to culture of 1:7. These were aliquoted into cryovials and transferred to a −80 °C freezer for long-term storage.

5.6.2.3 Revival

For subsequent revival of the frozen cultures, a 1.5 mL eppendorf tube was removed from the −80 °C freezer and put on ice. Small amount was streaked onto an agar plate containing LB and kanamycin. The plates were stored in an incubator at 37 °C overnight. Then the plates were sealed using parafilm and stored at 4 °C for up to two weeks.

5.6.2.4 Plasmid construction

Plasmids were constructed via PCR cloning. PCR primers were chosen to add restriction enzyme sites on the 5' and 3' were needed. Following PCR amplification, the amplified DNA was purified using the Qiagen PCR cleanup kit (Qiagen, Crawley, U.K). Double digests were carried out and the desired fragment isolated via gel extraction. The relevant fragments were subsequently ligated. Following construction, each plasmid was isolated using the QIAprep Spin Miniprep Kit (Qiagen, Crawley, U.K). Plasmid concentration was determined using the Thermo Scientific NanoDrop 1000 Spectrophotometer (Fisher Scientific, MA, U.S.A).

5.6.2.5 Polymerase Chain Reaction

In order to amplify DNA and add the restriction enzyme sites required, a Polymerase Chain Reaction (PCR) reaction was carried out with mutagenic primers. A list of primers can be found in Appendix (XXX). Q5® DNA Polymerase (NEB, MA, U.S.A) was used with its associated buffer, dNTPs and Q5® enhancer, as specified in Table 5.2. PCR reactions were run in a T100^{TM} thermal cycler (Bio-Rad Laboratories, Inc., UK) as per the Q5® recommendations, and as outlined in Tables 5.2 and 5.3.

Table 5.2 PCR recipe

Reagent	Final concentration	50 µL reaction
Q5® buffer 5X	1X	10 µL
dNTPs	200 mM each	1 µL
Forward primer	0.5 µM	2.5 µL
Reverse primer	0.5 µM	2.5 µL
Template DNA	2 µg/50 µL	-
Q5® DNA polymerase	0.02 U µL^{-1}	0.5 µL
Q5® enhancer	1X	10 mL
H_2O	-	to 50 µL

Table 5.3 Thermocycling conditions

Step	Cycles	Temperature	Time
Initiation	1	98 °C	30 s
Denaturation		98 °C	10 s
Annealing	30	72 °C	20 s
Extension		72 °C	2 min
Final extension	1	72 °C	2 min
Hold	1	4 °C	∞

5.6.2.6 Digestion

All enzymes, buffers and Bovine Serum Albumin (BSA) were supplied by NEB. Digestion controls were carried out by adding H_2O instead of DNA in the digestion reaction. Additionally, during agarose gel electrophoresis uncut plasmid was run alongside the digested plasmid in order to detect the difference.

2 µg digests were set up by mixing the plasmid with 0.5 µL of each restriction enzyme, 3 µL 10x buffer and 3 µL 10x BSA. H_2O was added to make the reaction to 20 µL. The recipe used is shown in Table 5.4. The reactions were placed in an incubator at 37 °C for 4 hours. Finally, the solutions were analysed using agarose gel electrophoresis (Section 5.6.2.7).

Table 5.4 Digestion recipe

Reagent	Volume
PstI	0.5 µL
HindIII	0.5 µL
NEB Buffer 2.1	2 µL
BSA	0.2 µL
DNA	1 µg
H_2O	to 20 µL

5.6.2.7 Agarose gel electrophoresis

To make a 0.8% agarose gel, 0.4 g agarose were diluted in 50 mL 1X TAE buffer. It was further dissolved by microwaving for 1-3 minutes. The solution was left to cool for 5 minutes and then 1.5 µL gel red were added. Gel trays were prepared by putting the well comb in place and taping the ends shut. The solution wast then

poured into the prepared gel trays and left to solidify for 20-30 minutes at room temperature.

Agarose gel electrophoresis was carried out by placing the poured gels into the gel tanks. The tank was then flooded with 1X TAE buffer. The DNA was prepared to be analysed by adding 4 µL loading dye to 20 µL sample. A negative control was used with H_2O instead of sample. The DNA ladder of choice was prepared by adding 1 µL H_2O and 1 µL dye to 2 µL ladder. Each sample was added to a well by pipetting. The agarose gel was ran at 90 V until the dye was 80% of the way down the gel, approximately 1 hour.

To purify the fragments from the agarose gel, the gel was placed in a UV box. Using a sterile razor blade, the desired fragment was cut out and placed in a clean eppendorf tube. The DNA was isolated from the gel using the QIAquick Gel Extraction Kit.

5.6.2.8 Ligation

A ratio of 3:1 of insert to recipient plasmid was used, 1 µL T4® DNA ligase (NEB, MA, U.S.A) and 2 µL ligase buffer. H_2O was added to make the reaction up to 20 µL. The controls used for each ligation reaction, are shown in Table 5.5. Control 1 is used to detect competent cell viability, control 2 background due to uncut vector, control 3 contamination and control 4 vector re-circularization.

The ligation reactions were placed at 4 °C for 12 hours. The reactions were then placed at 65 °C for 10 minutes to heat inactivate the T4 DNA ligase enzyme. A transformation was then carried out as per Section 5.6.2.9.

Table 5.5 Ligation controls

	Control 1	Control 2	Control 3	Control 4
Vector	Uncut	✓	✓	✗
Insert	✗	✗	✗	✓
Buffer	✓	✓	✓	✓
H_2O	✓	✓	✓	✓
Ligase	✗	✗	✓	✓

5.6.2.9 Transformation

Thermocompetent *E.coli* Dh5α was transformed with the constructed plasmids. Each ligation reaction was added to 50 µL of thawed competent cells. The cells were sub-

sequently kept on ice for 30 minutes, then placed at a 42 °C water bath for 45 s. The cells were then placed back on ice for 15 minutes. Then 500 µL of Super Optimal broth with Catabolite repression (SOC) were added to each ligation and placed in a 37 °C shaking incubator for 3 hours. 500 µL and 50 µL were subsequently pipetted of each ligation onto petri dishes with LB agar and the appropriate antibiotic. The plates were incubated at 37 °C for 12-16 hours. Two controls were used for the transfection protocol, a positive control with no antibiotic in the LB agar and non-transfected cells and a negative control of non-transformed cells and LB agar with antibiotic. These ensure that the cells are viable and not contaminated respectively.

Finally, the number of colonies were counted on each plate. Individual colonies were then selected from each transfection and grew each separately in 5 mL LB medium for 12-16 hours at 37 °C, 200 rpm. Glycerol stocks were then prepared from each culture, as per Section 5.6.2.2.

5.6.2.10 Colony PCR

In order to determine if the fragment was successfully inserted into the vector DNA plasmid, diagnostic colony PCR was then carried out. Primers were designed that amplified the multiple cloning site of the vector DNA plasmid. These can be found in Appendix (XXX). A PCR master mix was made for the number of colonies to be amplified, 32, with an added 10% to account for pipetting error. GoTaq® Flexi DNA polymerase (Promega Corp., WI, U.S.A.) was used with its associated buffer, dNTPs and $MgCl_2$and H_2O. The recipe for the master mix is shown in Table 5.6.

Table 5.6 Colony PCR master mix recipe

Reagent	Final concentration	Master mix
GoTaq® green Flexi buffer	1X	141 µL
dNTPs	200 mM each	14.1 µL
Forward primer	0.5 µM	1.4 µL
Reverse primer	0.5 µM	1.4 µL
GoTaq® Flexi polymerase	0.02 U μL^{-1}	3.5 µL
$MgCl_2$	1X	42.2 µL
H_2O	-	465 µL

19 µL were then added from the master mix to each PCR tube. Each of the colonies was then lifted from the transformation from the agar plate using a 20 µL pipette tip and added it to a PCR mix by mixing. The pipette tip was subsequently used to

make a scratch into a clean agar plate, and labelled it. A PCR was then carried out according to GoTaq® Flexi polymerase recommendations, and as shown in Table 5.7.

Table 5.7 Thermocycling conditions for colony PCR

Step	Cycles	Temperature	Time
Cell lysis	1	95 °C	10 minutes
Denaturation		95 °C	30 s
Annealing	35	50 °C	1 minute
Extension		72 °C	1 min
Final extension	1	72 °C	5 min
Hold	1	4 °C	∞

Finally a diagnostic agarose gel electrophoresis was carried out as outlined in Section 5.6.2.7.

5.6.2.11 Sequencing

In order to confirm plasmid identity, all plasmids were sequenced using Source Bioscience, Cambridge UK. 10 µL of each plasmid DNA were submitted at a minimum of 100 ng μL^{-1} as per the requirements. Primer sequences were also submitted and manufactured by Source Bioscience. Primers can be found in Appendix (XXX).

5.6.2.12 Inducers

Anhydrotetracycline (ATc) solution was made by diluting ATc from Cayman Chemical Company in 100 % ethanol to a concentration of 1 mg mL^{-1}. Isopropyl-beta-D-thiogalactopyranoside (IPTG) solution was made by dissolving IPTG in deionized water to a concentration of 1 M. The solution was sterilised by passing the solution through a 0.22 µm syringe filter. Both inducers were stored in 1 mL aliquots at −20 °C.

5.6.2.13 Growth rate measurement

Plate reader analysis was carried out in order to measure the growth of *E.coli* over time. Overnight cultures were made using the method shown in Section 5.6.2.1. Overnight cultures were then diluted by a 1:1000 ratio into a 5 mL LB + kanamycin solution. The diluted cultures were grown at 37 °C with shaking at 200rpm for 1 hour. These cultures were then further diluted by a 1:100 ratio. 200 µl aliquots of the dilutions were then transferred to a clear bottom, black-walled 96-well plate.

Wells with only LB and kanamycin were also added in order to be used as blanks. The plate was then sealed using a gas permeable membrane and placed it in BMG FLUOstat OPTIMA plate reader to measure absorbance. The plate reader was set to a constant 37 °C, with 30 seconds orbital shaking at 150 rpm and 4 mm shaking width every ten minutes. Absorbance was measured at 540 nm. Data was exported as a CSV file and analysed using Python.

5.6.2.14 Flow cytometry

Flow cytometry experiments were carried out in order to get fluorescent levels in single cells. Flow cytometry allows us to gather this information for thousands of single cells. Flow cytometry data was exported as FCS files and analysed using the R bioconductor packages flowCore (Ellis et al. 2016b), flowViz (Ellis et al. 2016a) and Ggplot2 (Wickham 2009). Prior to analysis the raw data was processed to remove any debris or instrument noise detected. The data was also processed to removed any doublets, which occurs when more than one bacterial cell passes through the detector at a time. This will skew the data by including datapoint with double the fluorescent intensity that the rest of the population. The pre-processing was done by using the side scattering data. The height and the area of the sample forward scattering distribution is recorded during an experiment. The cells that lie in the diagonal where the area equals the height are single bacterial cells. If the area of the signal exceeds the height it is indicative of a doublet, or cluster of cells, and is removed from the data. This preprocessing was carried out using autoGate, developed by Fedorec (2016).

5.6.2.15 Concentration assays

Concentration assays were carried out in order to determine the concentration of each inducer (ATc and IPTG) at which the switch flips. Separate overnight cultures were prepared as per Section 5.6.2.1 with added IPTG at a concentration of 1 mM or added ATc at a concentration of 100 ng mL^{-1} (Litcofsky et al. 2012). The cultures were then diluted by 1:1000 into fresh LB medium with varying concentrations of the opposite inducer than what the cells were grown in overnight. The concentrations used are shown in Table 5.8. For each concentration. three replicates cultures were made.

The cultures were placed in an incubator at 37 °C, 200rpm for 5 hours. The cultures were then placed in a centrifuge and spun at 13,000rpm for 5 minutes. The supernatant was discarded and replaced it with 1 mL PBS solution. The BD

Table 5.8 Concentrations used for flow cytometry assay

ATc (ng/ml)	IPTG (M)
0.05	1e-7
0.06	6e-7
0.07	1e-6
0.08	6e-6
0.09	1e-5
0.1	1e-3
1.0	0.1

LSRFortessa^{TM} cell analyzer (Becton, Dickinson and Company) was used at the St. Mary's Flow Cytometry Core Facility at Imperial College London for flow cytometry analysis. GFP was excited using the 488 nm laser and detected using the 533/30 filter. mCherry was excited using the 561 nm laser and detected using the 620/10 filter. Data was obtained at n=10000 events per experiment.

5.6.2.16 Time course assays

Time course assays were carried out to measure the time it takes for the switch to flip to each state. Separate overnight cultures of pKDL071 were prepared as per Section 5.6.2.1 with added IPTG at a concentration of 1 mM or added ATc at a concentration of 100 ng mL^{-1} (Litcofsky et al. 2012). Overnight cultures of pSEVA281G and pSEVA281C were also made. The cultures were then diluted by a ratio of 1:1000 into fresh LB medium. Separate cultures for each time point were made, in triplicate. For cultures grown overnight in IPTG, ATc was added at a concentration of 100 ng mL^{-1} and for cultures grown overnight in ATc, IPTG was added at a concentration of 1 mM. All cultures were placed at 37 °C, 200rpm incubator. At 30 minutes, 1 hour and then every hour up to 6 hours flow cytometry was carried out for the corresponding cultures. Triplicates for each induction were removed from the incubator and placed in a centrifuge at 13, 000rpm for 10 minutes. The supernatant was discarded and replaced with 1 mL PBS solution. These cultures were then analysed in an Attune^{TM} NxT Flow Cytometer (Thermo Fisher Scientific) at University College London. GFP was excited using the 488 nm laser and detected using the 533/30 filter. mCherry was excited using the 561 nm laser and detected using the 620/10 filter. Data was obtained at n=10000 events per experiment. pSEVA281G and pSEVA281C cultures were used to set the laser voltages and pKDL071 cultures to detect the bacteria population.

5.6.3 Results

5.6.3.1 pKDL071 plasmid alteration

The pKDL071 plasmid contains all the elements of the switch. The two states of the switch are LacI high and TetR high. These are detected by using the fluorescent proteins that are controlled by the same promoters, and thus mirror the levels of LacI and TetR. The concentration of LacI can be estimated by GFP intensity and TetR concentration by mCherry intensity. In order to detect GFP and mCherry levels within each cell simultaneously, flow cytometry can be used. The lasers needed to

excite GFP and mCherry are 488 nm blue and 561 nm yellow respectively. Since the blue laser was not available for use in the BD Acuri^{TM} C6 or the BD LSRII^{TM} (Becton, Dickinson and Company) flow cytometers available, an alternative construct had to be made in order to be able to detect the levels of both sides of the switch.

In order to alter the switch construct to be able to detect both sides, the mCherry gene was swapped for the YFP gene. The yellow fluorescent protein is excited by the blue laser and could thus be detected using the equipment available. The YFP gene was available from BioBrick registry of standard biological parts as BBa_K592101. PCR cloning was used to introduce the flanking sequences of EcoRV and KasI restriction enzymes in the 5' and 3' ends respectively. The primers used are given in Appendix (XXX). A double digest was performed on plasmids pKDL071 and BBa_K592101, as well as positive and negative controls. Following gel extraction and ligation, the pKDL071-YFP plasmid was complete. The plasmid map is shown in Figure 5.14.

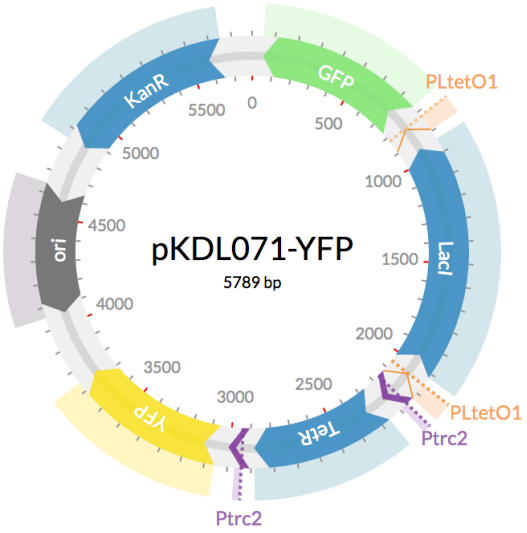

Figure 5.14 : pKDL071-YFP plasmid map.

GFP and YFP have overlapping emission spectra, which have to be compensated during flow cytometry data acquisition (Shapiro 1941). This is because the signal from GFP can be detected at the YFP detector and vice versa. Due to the high level of compensation needed to be carried out and the relatively dim signal given by the bacteria used here, the different stages of the switch, ON and OFF, could not be resolved (data not shown). In order to be able to acquire toggle switch flow cytometry data, an alternative facility was found that was able to detect GFP and

mCherry fluorescence.

5.6.3.2 Control plasmids construction

I constructed two plasmids in order to use them for the flow cytometry mCherry-/GFP experiments. The first plasmid, pSEVA281G contains the promoter PLtetO-1 and GFP and the other, pSEVA281C, contains the promoter Ptrc2 and mCherry from PKDL071, shown in Figure 5.15. These two plasmids were used to determine the appropriate voltages for the lasers that excite GFP and mCherry.

pSEVA281G was constructed by digesting pKDL071 and pSEVA281 using the protocol outlined in Section 5.6.2.6. pSEVA281 is a plasmid backbone containing kanamycin resistance, a high copy origin of replication and a multiple cloning site. The digested fragments were isolated using gel purification (Section 5.6.2.7) and then ligated the isolated fragments (Section 5.6.2.8). *Escherichia coli* Dh5α was then transformed with each plasmid (Section 5.6.2.9).

pSEVA281C was constructed via PCR cloning. PCR was carried out using the pKDL071 plasmid as a template DNA using the protocol outlined in Section 5.6.2.5. Primers were chosen so that Ptrc2 and mCherry were copied and a HindIII restriction enzyme recognition sequence added to the fragment. The rest of the cloning procedure followed as per plasmid pSEVA281G.

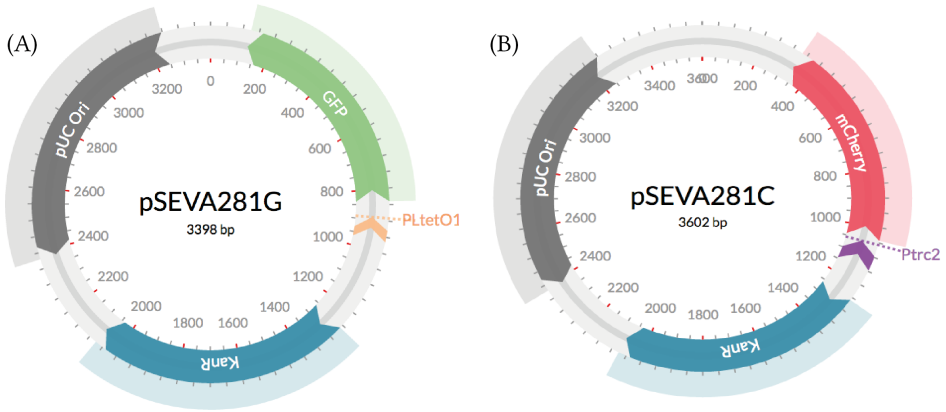

Figure 5.15 : The plasmids used to calibrate GFP and mCherry fluorescence. (A) pSEVA281G plasmid map (B) pSEVA281C plasmid map.

5.6.3.3 Growth rate investigation

I carried out a growth rate analysis to determine whether the ATc or IPTG added to pKDL071 or pSEVA281G *E. coli* cultures affected the growth of the bacteria. Cultures were grown without any inducer overnight as described in Section 5.6.2.13. Assays for the cultures were ran with and without added inducers. As can be seen in Figure 5.16, there is no difference between the conditions. The addition of either ATc or IPTG does not affect the growth rate of *E. coli* K-12 MG1655. Additionally, ATc does not affect the growth rate of *E. coli* Dh5α. Since the addition of ATc flips the switch to the GFP high state, and IPTG to the mCherry high state, we can also conclude that the growth rate of the chassis is not affected by which side of the switch is in the high state. The growth rate of *E. coli* Dh5α was consistently lower than that of *E. coli* K-12 MG1655.

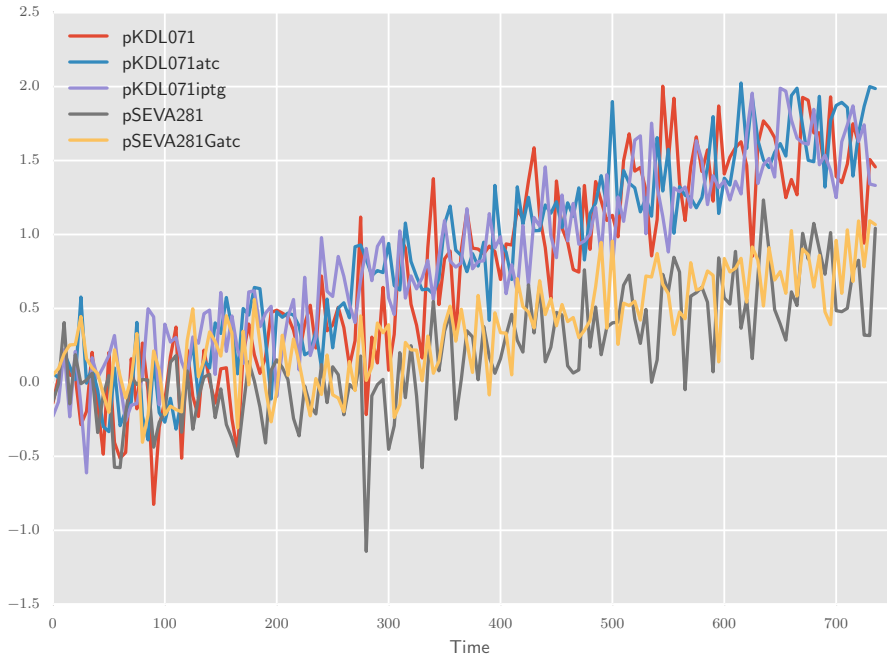

Figure 5.16 : Growth rate analysis of *E. coli* K-12 MG1655 pKDL071 and *E. coli* Dh5α pSEVA281G cultures with and without inducers. The inducers do not affect the growth of the bacteria.

5.6.3.4 Toggle switch concentration assays

Here I aim to identify the inducer concentration at which the pKDL071 toggle switch changes state. In order to do that I carry out a concentration assay using flow cytometry, as described in Section 5.6.2.15. As can be seen in Figure 5.17A, during ATc induction the switch flips to a GFP high state when ATc concentration is at $0.09\,\mathrm{ng\,mL^{-1}}$ or higher. We observe a bimodal distribution at concentrations $0.07\,\mathrm{ng\,mL^{-1}}$ and $0.08\,\mathrm{ng\,mL^{-1}}$, which indicates that the switching has begun at these concentrations. Thats why part of the population has switched to the GFP high state but complete switching is not observed until the concentration of ATc is at $0.09\,\mathrm{ng\,mL^{-1}}$. In the case of IPTG induction (Figure 5.17B) we find that the switch flips to the mCherry high state when the concentration of IPTG is higher or equal to 0.001M. A decrease in GFP fluorescence is also observed. We do not observe a bimodal distribution in this case.

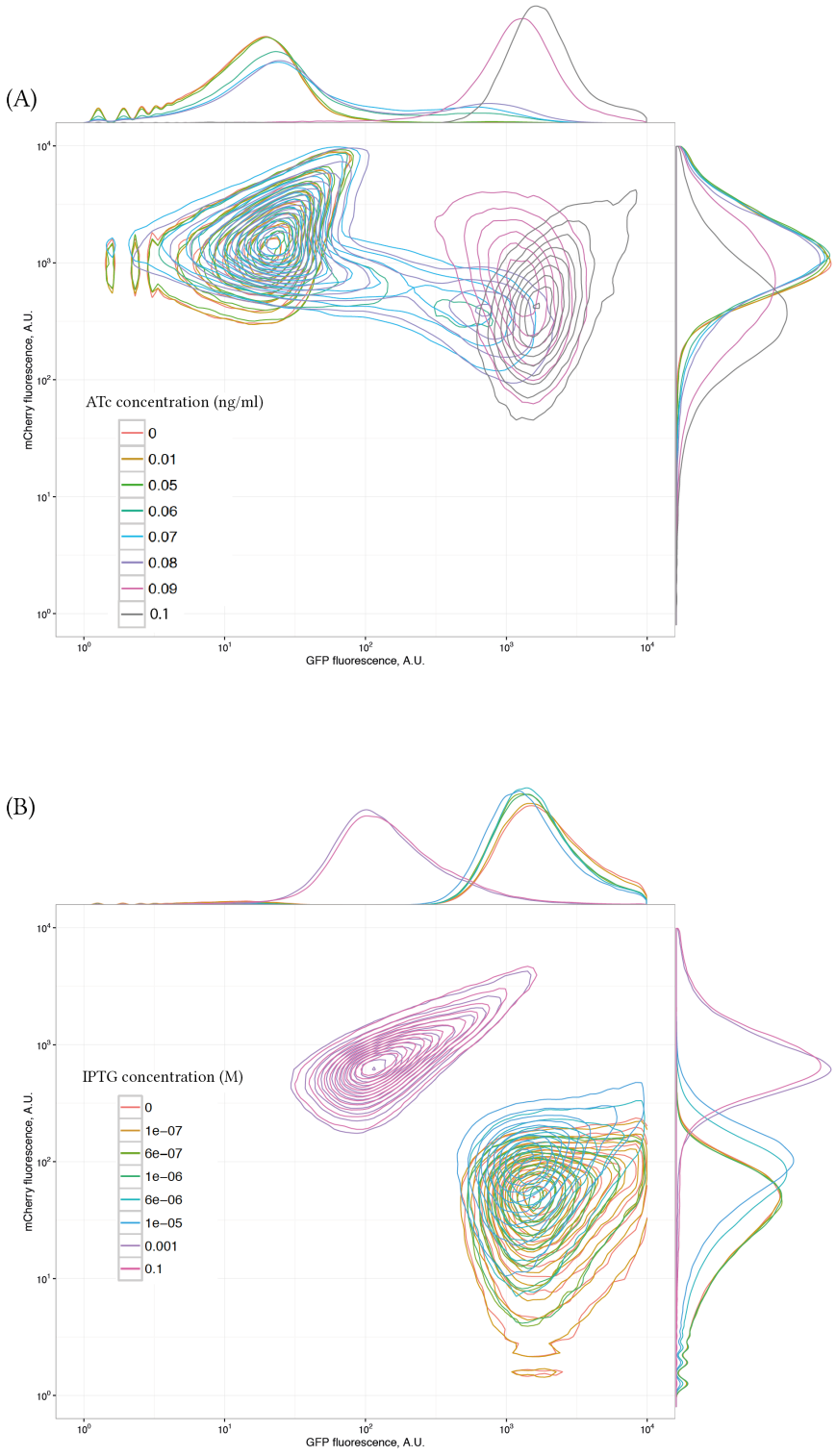

Figure 5.17 : (A) ATc induction at various concentrations (B) IPTG induction at various concentrations.

By taking into account the two induction curves of the switch turning to each high state, we can see the dynamic ranges of pKDL071 in *E.coli*. We can see in Figure 5.18 there is an approximately 100-fold change in fluorescent units during IPTG and ATc induction.

A Hill function was used to model the characterisation curves shown in Figure 5.18. The model used is the following:

$$F = Pmin + (Pmax - Pmin)\frac{\left(\frac{[I]}{Kd}\right)^n}{1+\left(\frac{[I]}{Kd}\right)^n}, \tag{5.8}$$

where F is the median fluorescent unit and [I] is the concentration of inducer. Pmin and Pmax are the minimum and maximum fluorescence respectively, and Kd and n are the dissociation constant, and Hill coefficient. I fit Hill function models using maximum likelihood estimation to the response curves. The values for parameters Pmin, Pmax, Kd, and n are 8, 1600, 0.1, 1.8 respectively for the ATc induction and 8, 700, 0.08, 2.5 for IPTG induction.

For the case of the ATc induction we observe a sharp switch between the GFP low to the GFP high state, as can be seen in the characterisation curve in Figure 5.18B. This is a clear indication of the bistability of this switch.

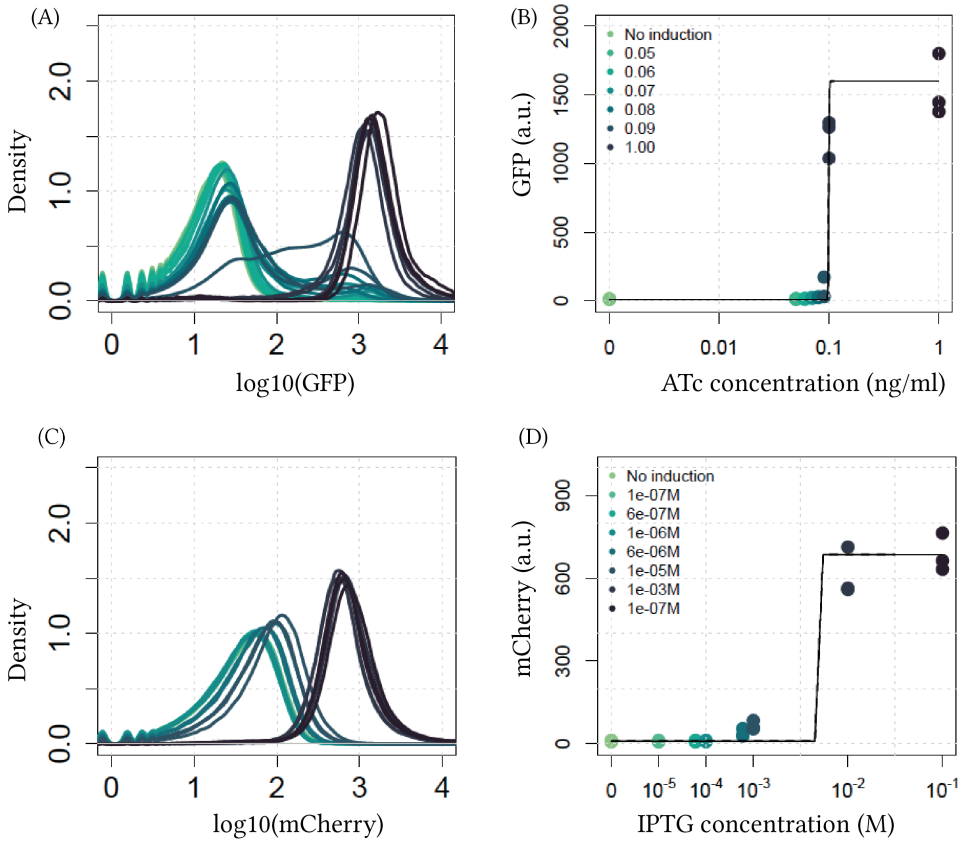

Figure 5.18 : (A, B) ATc induction of pKDL071. (C, D) IPTG induction of pKDL071.

5.6.3.5 Toggle switch time course assay

I further analysed the pKDL071 toggle switch by investigating the time it takes for it to switch from one high state to the other. To do that I used the method outlined in Section 5.6.2.16. I obtained separate time courses for the IPTG and ATc inductions.

As can be seen in Figure 5.19 pKDL071 ATc induction begins switching 1 hour after induction. Complete induction is seen at 6 hours. During the IPTG induction (Figure 5.20) we see a bimodal distribution at 4 hours, and induction is complete at 6 hours. We observe that during ATc induction there is an increase in GFP fluorescence and a decrease in mCherry fluorescence, in the case of IPTG induction the increase in mCherry fluorescence is not as prominent. A decrease in GFP fluorescence is observed during IPTG induction.

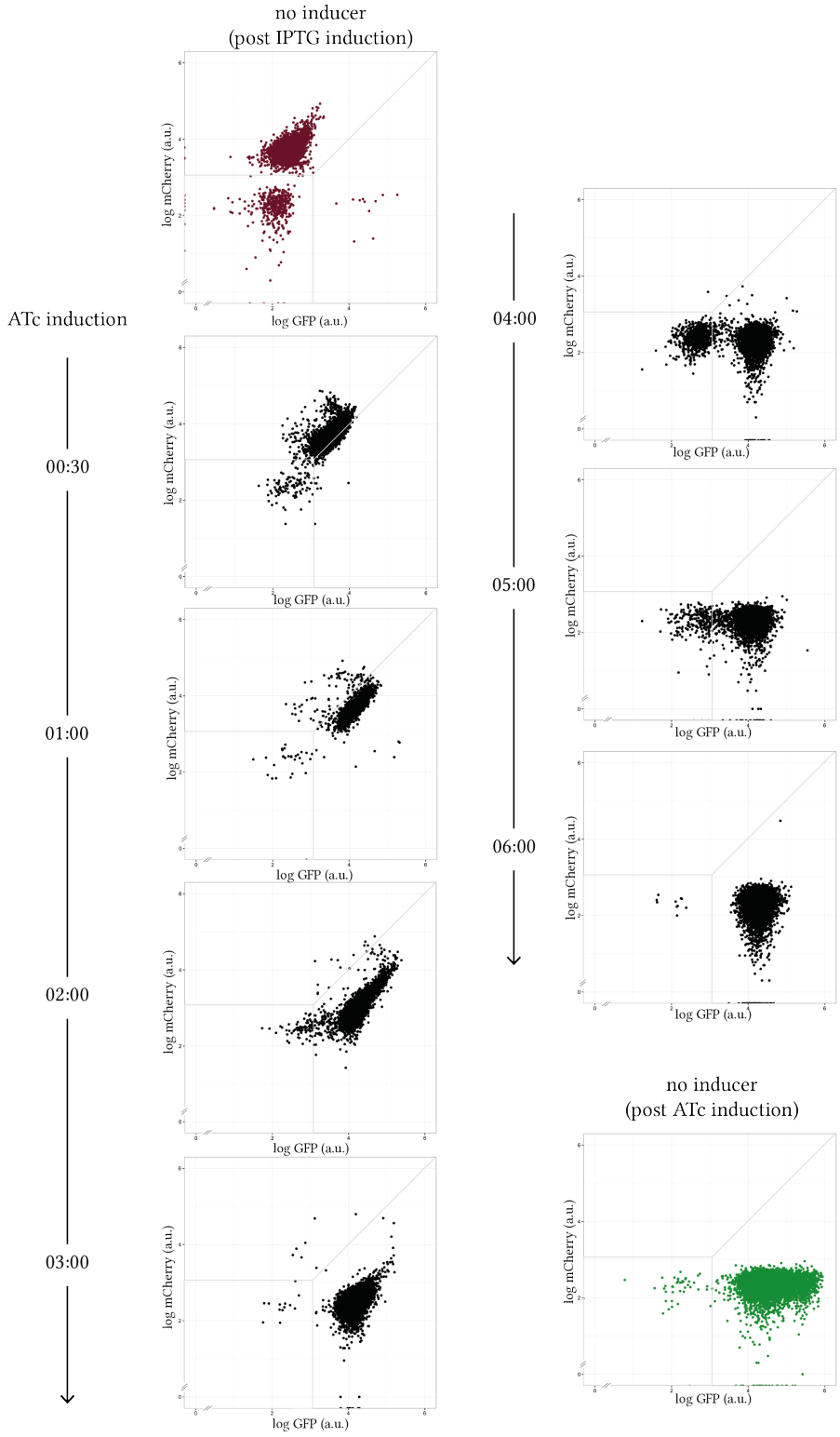

Figure 5.19 ATc induction of pKDL071 over time

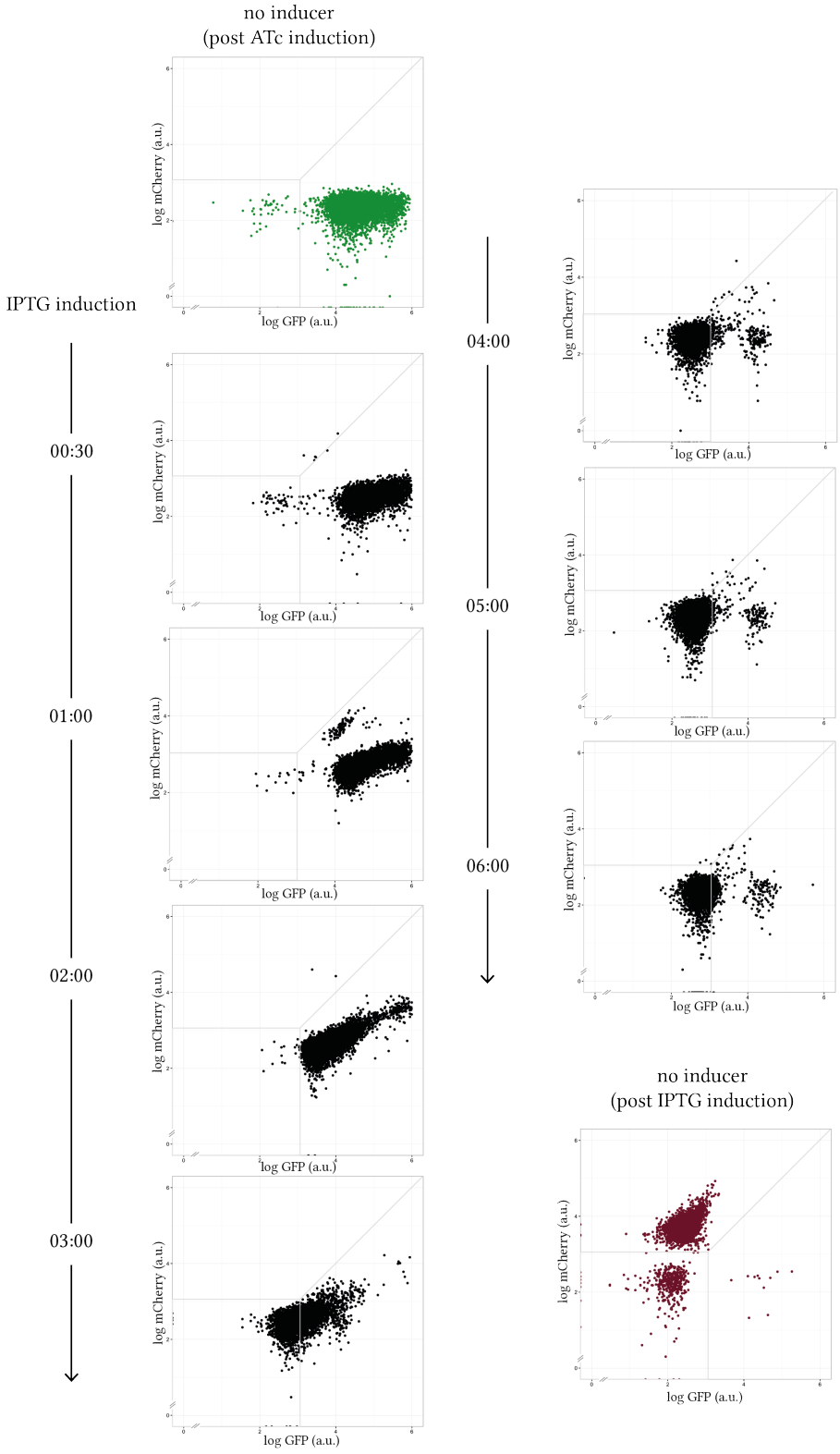

Figure 5.20 IPTG induction of pKDL071 over time

In the next section I use ABC-FLow to fit a computational model to the timecourse data obtained. Prior to fitting a model to it, I process the data by removing the unresponsive populations. This ensures that the model is fitted only to the data from cells that respond to the inducers. As seen in Figure 5.19, during the ATc induction there is an unresponsive population of cells where GFP and mChery fluorescence are both less that 10^3. This population is excluded from further analysis of the data. During the IPTG induction there is a population of cells that does not repsond to the addition of IPTG by switching from GFP high to mCherry high. This population of cells is also excluded from further analysis.

5.7 ABC-Flow used on experimental data

In this section I apply ABC-Flow to the experimental flow cytometry data collected in Section 5.6.3.5. The data set is comprised of time course data of the Litcofsky et al. (2012) toggle switch. The two states of the switch are represented by the levels of GFP and mCherry intensity in each bacterial cell. Using ATc inducer, each cell transitions from a mCherry high state to a GFP high state.

5.7.1 Toggle switch model used in ABC-Flow

I used the extension to the Gardner, Cantor, & Collins (2000) switch described in Equations 5.4-5.7. The model used is defined by the following hazards:

$$h_1 = KD_u \times (1 + KI_u) \times GFP \tag{5.9}$$

$$h_2 = 60 \times \frac{R_u \times KL_u}{1 + KL_u + KR_u \times mCherry^2} \tag{5.10}$$

$$h_3 = KD_v \times (1 + KI_v) \times mCherry \tag{5.11}$$

$$h_4 = 60 \times \frac{R_v \times KL_v}{1 + KL_v + KR_v \times GFP^2}, \tag{5.12}$$

where GFP and mCherry represent the two fluorescent proteins in the system. The cooperativity of the repressors is represented in the model. KI_u and KI_v KL_u KL_v KD_u KD_v KR_u KR_v, parameters KI_u and KI_v increase the degradation of one of the species, and simulates the addition of a repressor, IPTG or ATc respectively.

The two production hazards, $h2$ and $h4$ are multiplied by 60 to reflect the copy number of the toggle switch plasmid in each cell. The plasmid containing the toggle

switch used here, pKDL071, contains the ColE1 origin of replication, and thus 50-70 copies of the plasmid are present in each cell (Milo et al. 2010).

The priors used in ABC-Flow for this model are given in Table 5.9. All priors given assume a uniform ditribution. The values where chosen in agreement with (Lillacci & Khammash 2013) and in reference to http://bionumbers.hms.harvard.edu/, the database of useful biological numbers (Milo et al. 2010).

Table 5.9 The priors used for the 1D and 2D ABC-FLow model fitting to flow cytometry data

Parameters				
Description	Symbol	Units	ATc induction	IPTG induction
IPTG-induced mCherry degradation rate	KI_u	h^{-1} μM^{-1}		1 - 10
GFP transcription rate	R_u	molecules h^{-1}	1 - 50	1 - 50
GFP translation rate	KL_u	h^{-1}	1 - 50	1 - 50
GFP degradation	KD_u	h^{-1}	0.1 - 2	0.1 - 2
mCherry-induced GFP repression rate	KR_u	molecules^{-1} h^{-1}	0.016 - 1.2	0.016 - 1.2
mCherry transcription rate	R_v	molecules h^{-1}	1 - 50	1 - 50
mCherry translation rate	KL_v	h^{-1}	1 - 50	1 - 50
GFP-induced mCherry repression rate	KR_v	molecules^{-1} h^{-1}	0.016 - 1.2	0.016 - 1.2
mCherry degradation	KD_v	h^{-1}	0.1 - 2	0.1 - 2
ATC-induced GFP degradation rate	KI_v	h^{-1} μM^{-1}	1 - 10	
Species				
	GFP	mM	0 - 1	100 - 1000
	mCherry	mM	100 - 1000	0 - 1
Intensity paramters				
Mean of fluorescence of single GFP molecule	μ_{GFP}	AU	5 - 200	5 - 200
Mean of fluorescence of single mCherry molecule	$\mu_{mCherry}$	AU	5 - 200	5 - 200
Standard deviation of fluorescence of single GFP molecule	σ_{GFP}	AU	5 - 200	5 - 200
Standard deviation of fluorescence of single mCherry molecule	$\sigma_{mCherry}$	AU	5 - 200	5 - 200

5.7.2 ATc induction

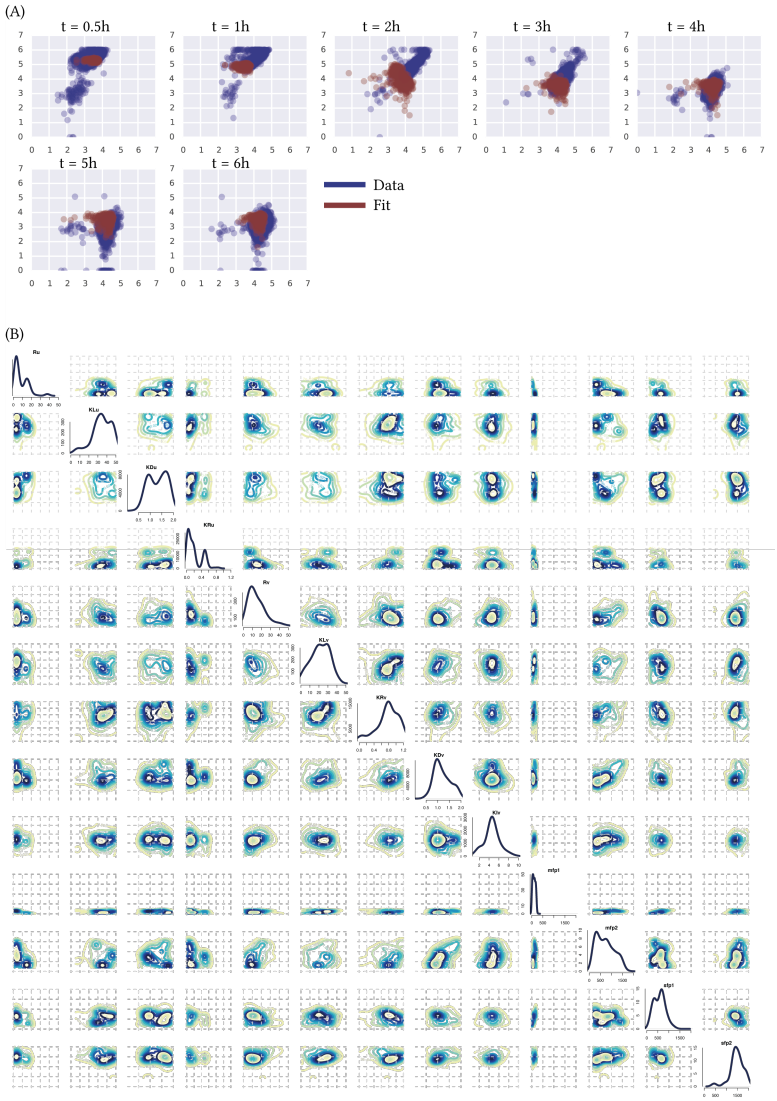

Figure 5.21 (A) The timecourse data collected in Section(XXX) (blue) and the resulting model fit from ABC-Flow (red). (B) The posterior distribution of the fitted morel.

5.7.3 IPTG induction

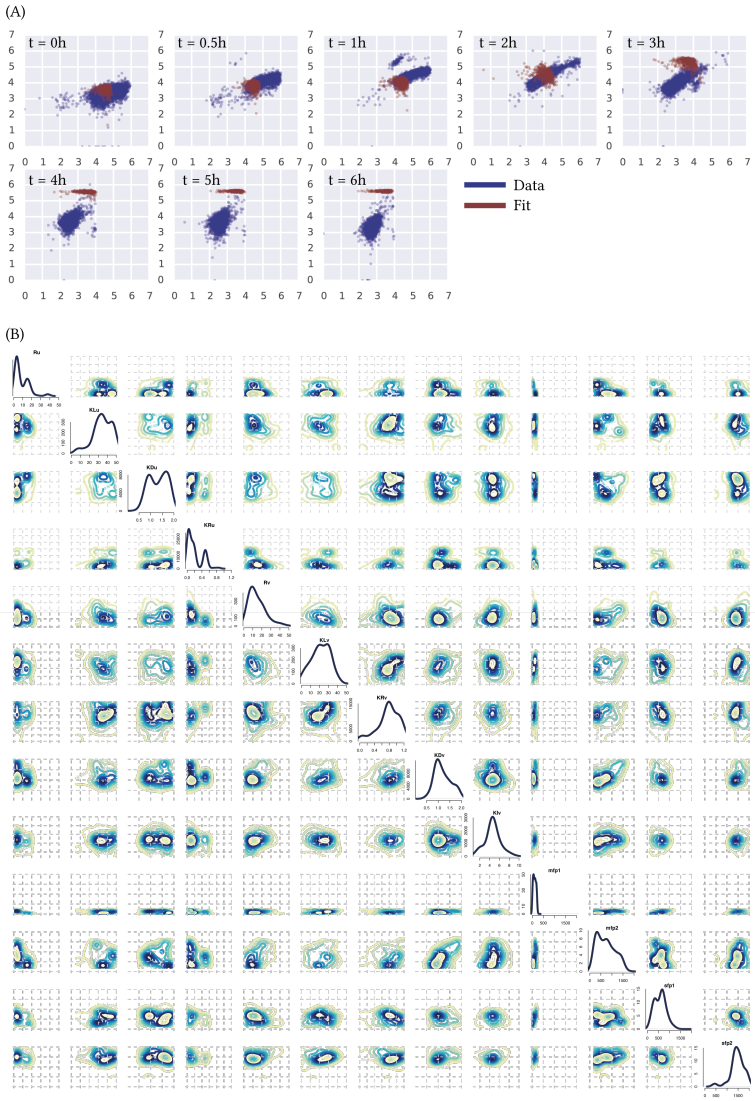

Figure 5.22

5.8 Discussion

Here I characterised the genetic toggle switch experimentally. First I study the effect of the two inducers ATc and IPTG on the growth rate of the selected chassis *E. coli* K-12 MG1655. I find that there is no detrimental effect to the bacterium by the inducers. I also find that which state the switch is on has no effect on the growth rate of the bacteria. In order for this toggle switch to be used in a synthetic biology application, it is important that both sides of the switch have an equivalent burden onto the chassis. If one of the steady states creates a larger burden and slows down the growth of the bacteria, this can create an imbalance in the population. If the toggle switch-bearing bacterial population exists in an environment with competing bacteria, for example the gut microbiome, and one of the two states creates a larger burden, this would cause the switch-bearing population to become less competitive compared to the non switch-bearing population. It is therefore crucial that the state of the switch does not affect the competitiveness of the chassis.

I further characterised the switch by determining the minimum inducer concentration necessary to change the state of the switch. I find that for ATc induction, a minimum of $0.09\,\mathrm{ng\,mL^{-1}}$ is required to cause the switch to go to a GFP high state. For IPTG induction I find that a minimum of 0.001 M is required to flip the switch to an mCherry high state. This information is critical for using this switch in other applications. Both sides of the switch are very sensitive to inducer concentrations, as the concentrations required to observe a change in fluorescence are very small.

Furthermore I find that this toggle switch, pKDL071, is faster to respond to a change in ATc concentration that to a change in IPTG concentration. For IPTG induction we observe a change in fluorescence after 3-4 hours of induction. For ATc induction we can see a difference within an hour of induction. This result is in agreement with Litcofsky et al. (2012). This difference in response times must be taken into account when using the pKDL071 switch for other applications. This difference could be due to maturation times of the fluorescent proteins. Macdonald, Chen, & Mueller (2012) found that mCherry half-maturation time is 150 mins, whereas the GFP variant used here, GFPmut3b has been especially mutated for fast action (Cormack, Valdivia, & Falkow 1996). Cormack, Valdivia, & Falkow (1996) found that whereas wild type GFP is detectable 1-2 hours after induction, GFPmut3b is detectable 8 minutes after induction. This difference could account for the different response times observed here, but further investigation is required.

ABC-Flow is similar to INSIGHT (Lillacci & Khammash 2013), as it used a Bayesian framework to fit the model to the data, and converts model output to fluorescence

intensity in order to compare the two.

5.9 Summary

In this chapter I summarised the experiments carried out for the analysis of the genetic toggle switch. I used the pKDL071 plasmid and characterised its switching behaviour over various inducer concentrations and over time. I found the concentration of each inducer necessary to flip the switch as well as the time it takes for the change to be observed. Furthermore, I investigated the effect of the inducers on the growth rate of the chassis and found that they have no effect. In the next chapter I use the data collected in the chapter to fit to the more realistic toggle switch models used in Chapter (XXX).

6 Designing new switches

6.1 Introduction

In the previous Chapters I studied the effect that adding positive feedback loops to the genetic toggle switch has on the robustness of the system. I found that adding two positive feedback loops to the simple toggle switch can increase its parametric robustness. The next step in this analysis would be to test these predictions experimentally. Therefore, in this Chapter I provide the experimental design for the construction of the genetic toggle switch with single and double positive autoregulation. The newly constructed switches could then be compared to the simple Litcofsky et al. (2012) toggle switch experimentally. Their robustness could be tested by varying the experimental conditions, like temperature and pH, and measuring the response of the switch.

Structurally, this Chapter is organised as follows: First I provide an overview of the cloning plan, by listing the relevant BioBrick parts used and their interactions. Then I outline the experimental design for producing these switches.

6.2 Cloning overview

The Litcofsky et al. (2012) toggle switch plasmid, pKDL071, used in Chapter (XXX) is modified to construct three new switches. Two switches will have single positive autoregulation, one on each gene, and one switch will have positive autoregulation on both genes. An overview of the cloning stages to be carried out is shown in Figure 6.1. The three stages required for the cloning plan to be completed are outlined in the sections below.

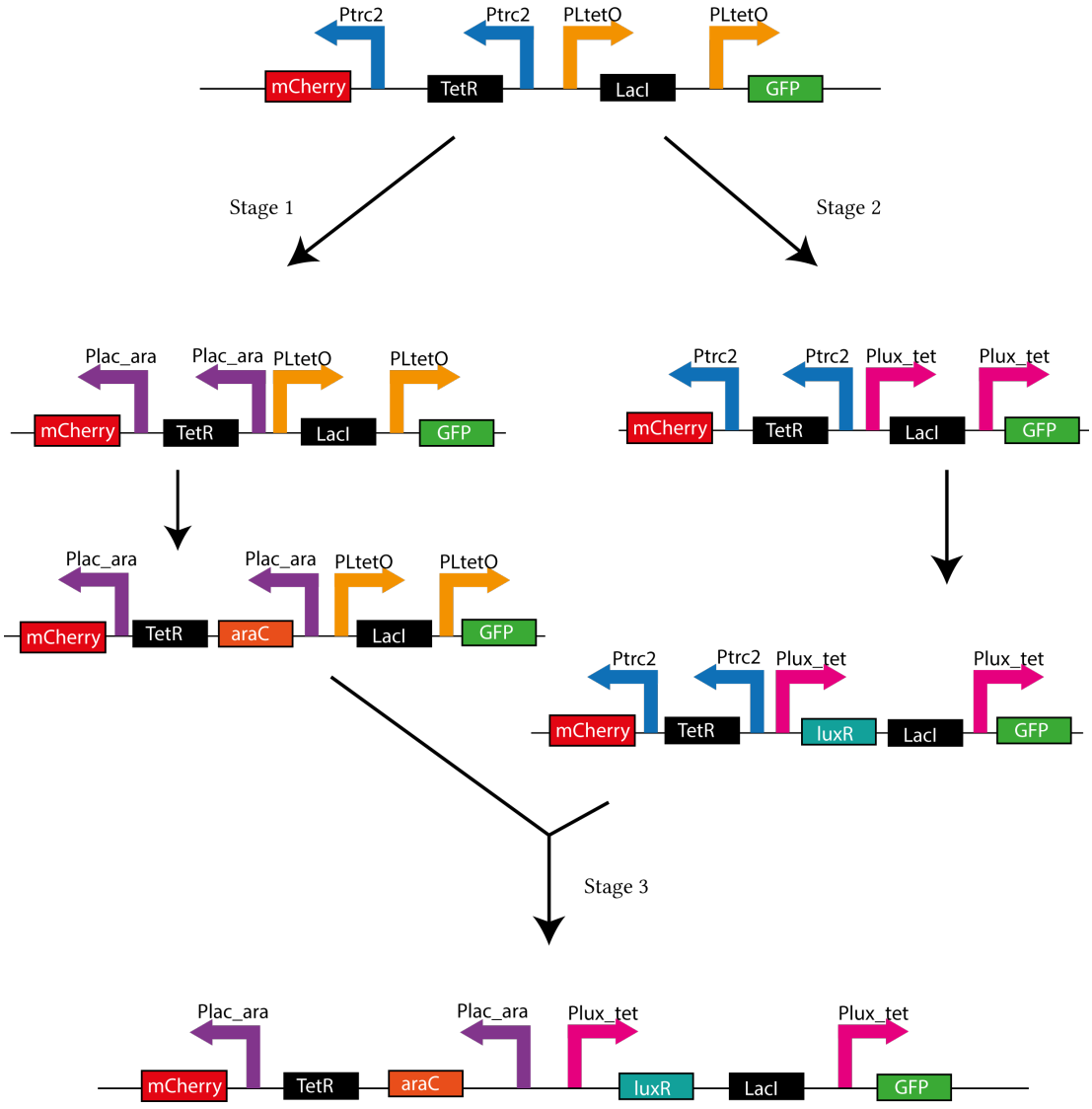

Figure 6.1 : An overview of the cloning plan to produce three new switches, two with single positive autoregulation and one with double positive autoregulation.

6.2.1 Resulting switches

The three switches shown in Figure 6.2 will be constructed through this cloning process. The first switch, on plasmid pKDL071-plac/ara-araC is a toggle switch with positive autoregulation on the TetR/mCherry side of the switch. The second plasmid, pKDL071-pLuxTet-luxR consists of a toggle switch with positive autoregulation on the LacI/GFP side of the switch. Finally, the switch with positive autoregulation on both sides of the switch is on the pKLD0713a plasmid. The plasmid maps and a schematic of their components' interactions are shown in Figure 6.2.

6.3 Experimental design

The construction of the three switches shown in Figure 6.2 is broken down in three stages, one for the construction of each switch. In this section I will outline the necessary cloning steps that need to be carried out in order to construct each switch. The detailed methods that will have to be used for each cloning step are described in Section (XXX). All primer sequences have been designed and are given in Appendix (XXX). Following the construction of each plasmid outlined below, competent *E.coli* cells will be transformed following the method outlined in Section (XXX).

6.3.1 Stage 1 - Construction of pKDL071-plac/ara-araC

In order to construct plasmid pKDL071-plac/ara-araC with single positive autoregulation on the mCherry/TetR side, the P_{trc2} promoter will be swapped for the $P_{lac\_ara\text{-}1}$ and AraC added upstream of TetR. $P_{lac\_ara\text{-}1}$ is activated by arabinose (AraC) and repressed by LacI (Lutz & Bujard 1997), and is thus ideal. The $P_{lac\_ara\text{-}1}$ promoter is present in the pJS167 plasmid, which is provided by Jeff Hasty (Addgene plasmid # 48881) (Stricker et al. 2008). This promoter is also present in the BioBrick registry of standard biological parts as BBa_K1713000.

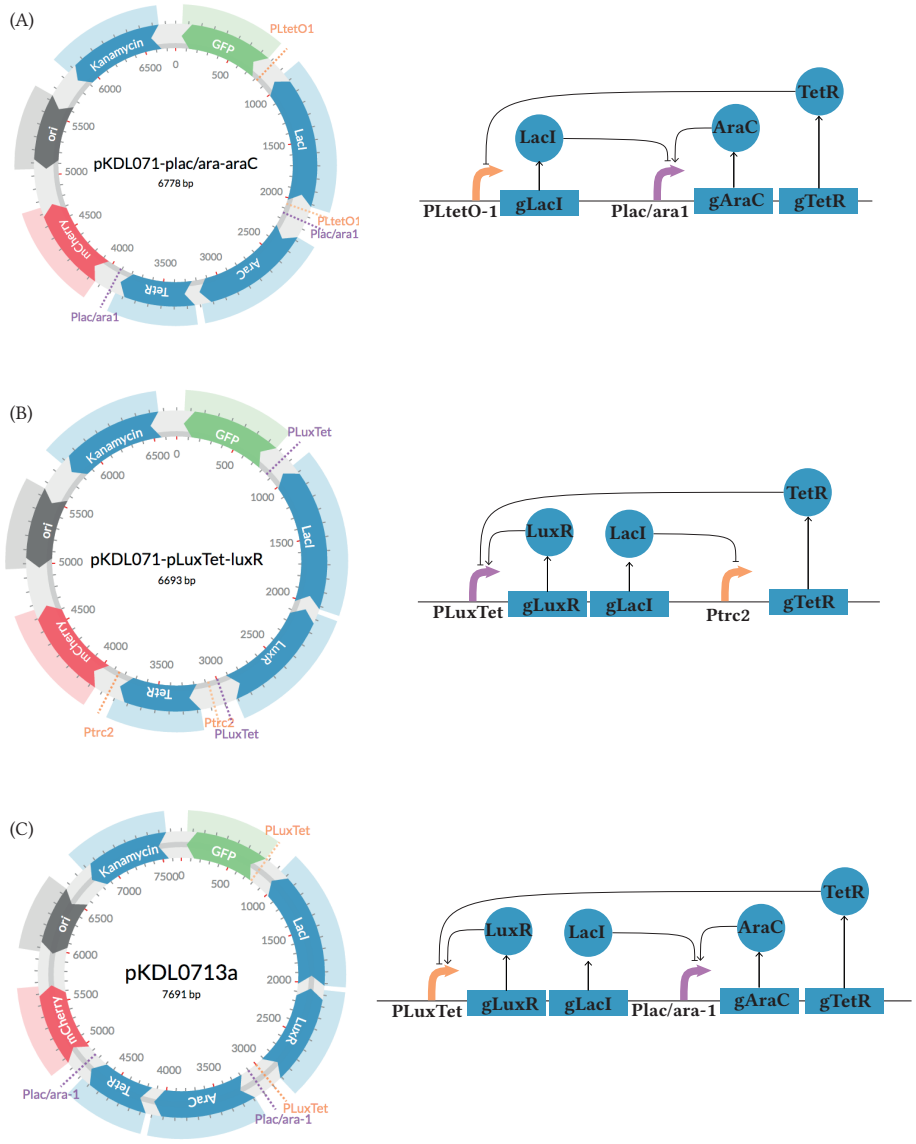

Figure 6.2 : The plasmid maps of three new switches to be constructed. The first two switches have a single positive autoregulation on each side of the switch respectively. The switch on the far right has positive autoregulation on both genes.

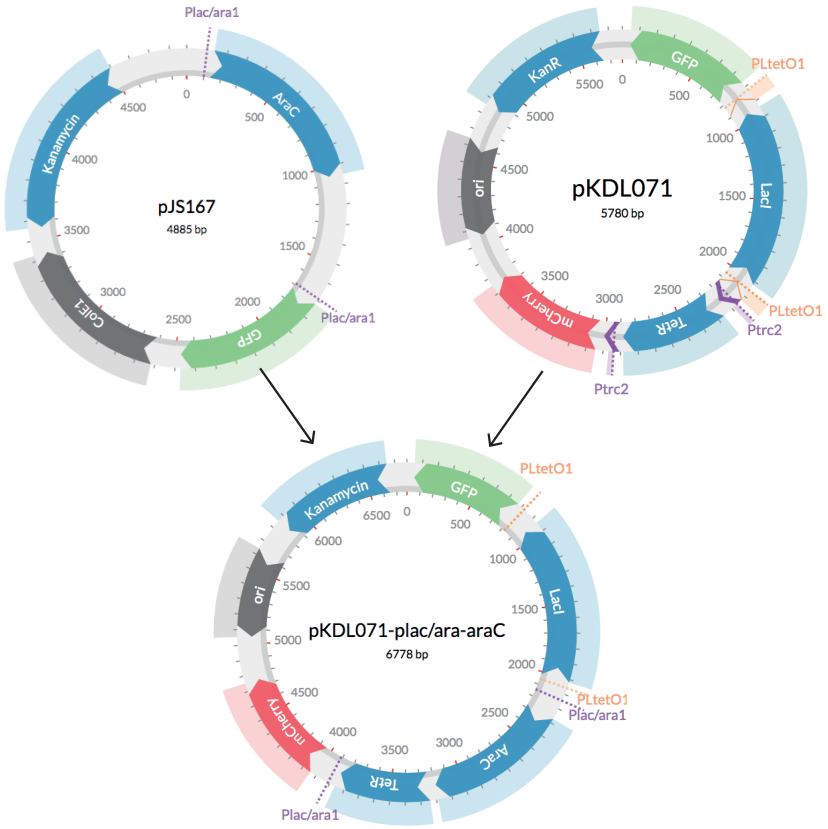

Figure 6.3 : Stage 1 cloning procedure. The pKDL071-plac/ara-araC plasmid is constructed via PCR cloning from the pJS167 and pKDL071 plasmids.

Using PCR cloning, the $P_{lac\_ara\text{-}1}$ promoter will be cloned from pJS167, with added XmaI and KasI restriction enzyme sequences on each end. Both pKDL071 and the PCR product will be digested with XmaI and KasI restriction enzymes. After gel electrophoresis and gel extraction, the two will be subsequently ligated. The resulting plasmid should have the $P_{lac\_ara\text{-}1}$ promoter instead of the P_{trc2} promoter upstream of mCherry.

Following that, PCR cloning will be used to clone $P_{lac\_ara\text{-}1}$ and AraC from the pJS167 plasmid. EagI and SalI flanking sequences will be added via PCR on the 5' and 3' ends. Plasmid pKDL071 and the PCR product will be digested using EagI and SalI. Following gel electrophoresis and gel extraction, the two products will be ligated in order to complete the pKDL071-plac/ara-araC plasmid. The detailed methods for each cloning technique mentioned here can be found in Section (XXX).

6.3.2 Stage 2 - Construction of pKDL071-pluxtet-luxR

In order to construct the plasmid pKDL071-pluxtet-luxR, the $P_{Lux/tet}$ promoter is necessary. The $P_{Lux/tet}$ promoter is present in the BioBrick registry of standard biological parts as BBa_K934024. $P_{Lux/tet}$ is a hybrid promoter activated by LuxR and repressed by TetR. This promoter will be added in exchange of $P_{LtetO\text{-}1}$ to the pKDL071 plasmid. The LuxR gene will also added upstream of LacI, in order to construct a switch with positive autoregulation on the LacI/GFP side.

First, the $P_{Lux/tet}$ promoter is synthesised. The reverse complement of BBa_K934024 with added flanking sequences of EcoO1091 and SphI on the 5' side and AclI and Eag1 at the 3' side. These are added to aid with further cloning steps. The sequence synthesised is given below:

5'- TTGGGACCTGCATGCTAATCTCTATCACTGATAGGGATAATACGAGTATCTC
TATCACTGATAGGGAGTAAACCTGTACGATCCTACAGGTAACGTTCGGCCG -3'

The pLux/tet and pKDL071 plasmids will subsequently be digested with SphI and AclI. Following gel extraction and ligation, the $P_{Lux/tet}$ promoter will be added upstream of GFP, replacing the $P_{LtetO\text{-}1}$ promoter. Then, the plux/tet and pKDL071 plasmids will be digested with EcoO109I and EagI restriction enzymes. Following gel extraction and digestion, the $P_{Lux/tet}$ promoter will be added upstream to LacI, replacing the $P_{LtetO\text{-}1}$ promoter.

The final stage of constructing the pKDL071-pluxtet-luxR plasmid consists of PCR cloning of the pTD103aiiA(Cm) plasmid with added BsGI flanking sequences

at both ends. The pTD103aiiA(Cm) is provided by Jeff Hasty (Addgene plasmid # 48886) (Prindle et al. 2012). The plasmid constructed in the previous step and the PCR product will be digested with BsGI restriction enzyme. Following gel extraction and ligation, the pKDL071-pluxtet-luxR should be complete. The ligated products will be transformed into thermocompetent *E.coli*.

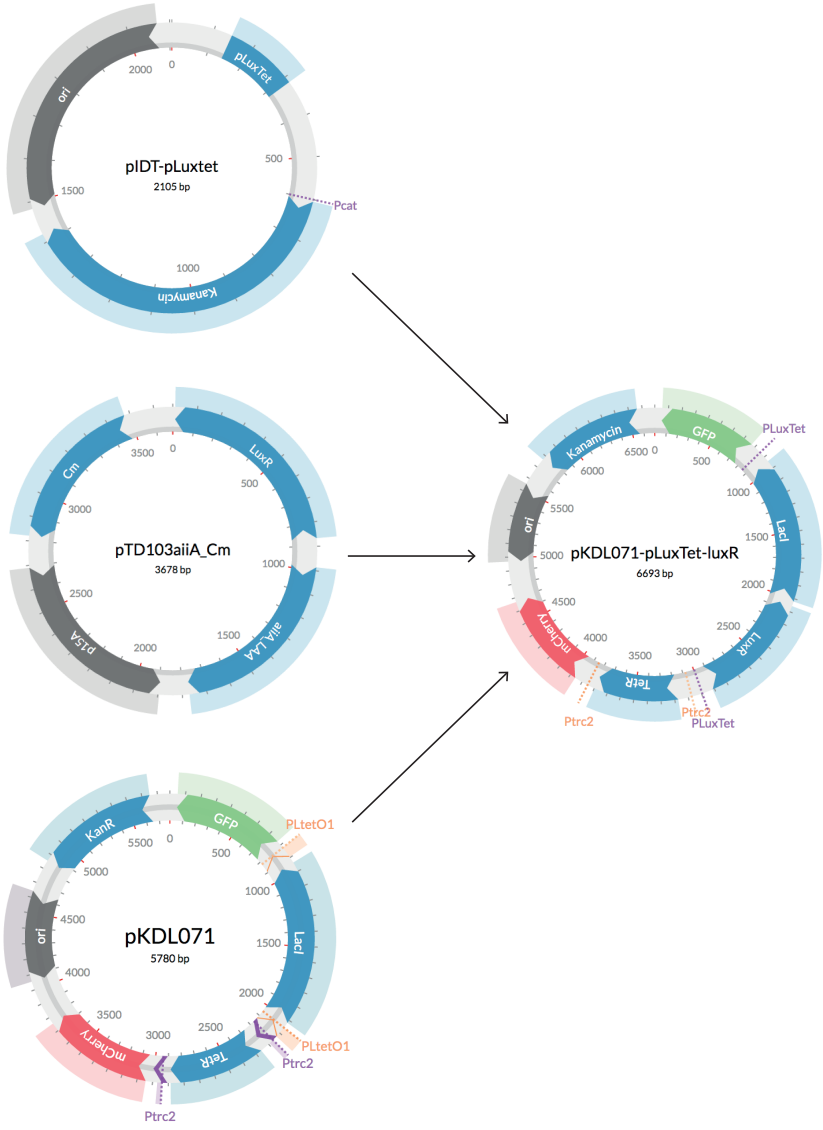

Figure 6.4 : Stage 2 cloning procedure. The pKDL071-pluxtet-luxR plasmid is constructed via PCR cloning from the synthesised $P_{Lux/tet}$ promoter, pKDL071 and pTD103aiiA(Cm) plasmids.

6.3.3 Stage 3 - Construction of pKDL0713a

The final construction stage requires the complete pKDL071-plac/ara-araC plasmid, as well as the synthesised $P_{Lux/tet}$ promoter and pTD103aiiA(Cm) plasmid used in Stage 2.

The plux/tet plasmid and pKDL071-plac/ara-araC will be digested with SphI and AclI restriction enzymes. This will be followed by gel extraction to isolate the fragments of interest. These will then be ligated to result in the modified pKDL071-plac/ara-araC plasmid, (pKDL071-plac/ara-araC-pluxtetA) with $P_{Lux/tet}$upstream of GFP instead of $P_{LtetO-1}$.

Then, the plux/tet plasmid and the plasmid created above (pKDL071-plac/ara-araC-pluxtetA) will be digested with EcoO109I and EagI. Following gel extraction and ligation, the $P_{Lux/tet}$ promoter will be added upstream of LacI instead of $P_{LtetO-1}$ to make a new plasmid, pKDL071-plac/ara-araC-pluxtet. Subsequently, the PCR product produced above of pTD103aiiA_Cm with BsGI flanking sequences and pKDL071-plac/ara-araC-pluxtet will be digested using BsGI. The fragments of interest will then be extracted following gel electrophoresis of the digested products and ligated. The ligates should be screened for the correct orientation of the insert.

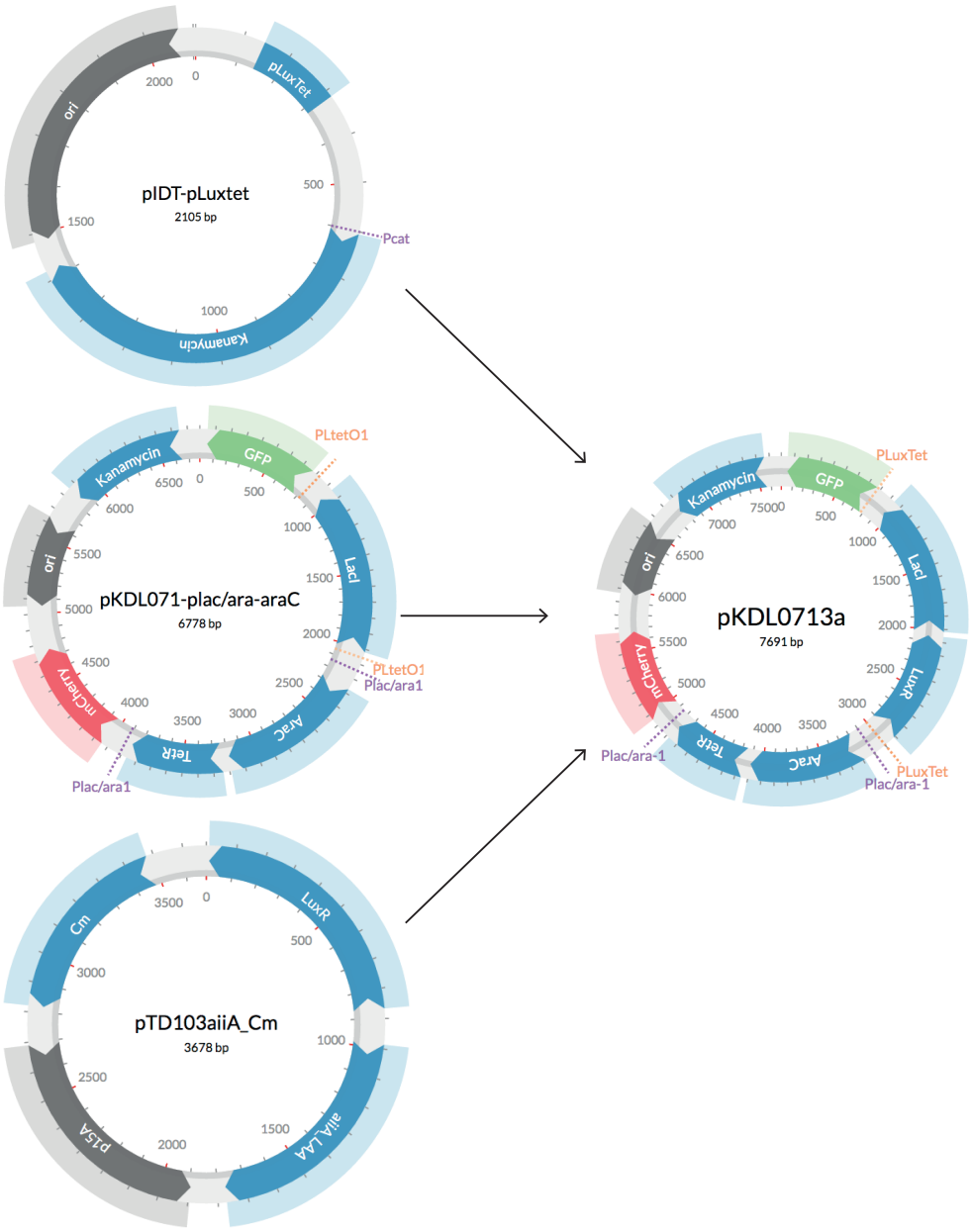

Figure 6.5 : Stage 3 cloning procedure. The pKDL0713a plasmid is constructed via PCR cloning from the synthesised $P_{Lux/tet}$ promoter, pKDL071-plac/ara-araC and pTD103aiiA(Cm) plasmids.

6.4 Discussion

I started implementing the experimental plan outlined above. In Stage 1 all the PCRs and digestions described were completed successfully (data not shown). Transformation of the ligated products into thermocompetent *E.coli* was carried out, but the transformation was not successful. In Stage 2 the $P_{Lux/tet}$ promoter was synthesised. Synthesis was carried out by Integrated DNA Technologies, Inc. (Leuven, Belgium, http://eu.idtdna.com/CodonOpt). *E.coli* Dh5α was transformed with the synthesised plasmid. The method is outlined in Section(XXX). All subsequent PCRs and digestions described in Section 6.3.2 were carried out successfully. Due to time constraints, the rest of the experimental plan was not carried out.

6.5 Summary

In this Chapter I designed the experimental protocol to be followed in order to construct three novel switches. These switches can be used in the future in synthetic biology applications. The execution of the experimental protocol described has not been completed to date but constitutes the future directions of this project.

7 Conclusions

7.1 Evaluation

7.2 Future work

Bibliography

Agapakis, C. M. & Silver, P. A. (2009). 'Synthetic biology: exploring and exploiting genetic modularity through the design of novel biological networks'. *Molecular BioSystems* **5**(7), 704.

Alon, U. (2007). *An Introduction To The Systems Biology*. Chapman & Hall/CRC.

Andrianantoandro, E., Basu, S., Karig, D. K., & Weiss, R. (2014). 'Synthetic biology: new engineering rules for an emerging discipline.' *Molecular systems biology* **2**(1), 2006.0028.

Attune NxT Acoustic Focusing Cytometer (2015). CO016625.

Babtie, A. C., Kirk, P., & Stumpf, M. P. H. (2014). 'Topological sensitivity analysis for systems biology.' *Proceedings of the National Academy of Sciences of the United States of America* **111**(52), 18507–18512.

Banaji, M. & Craciun, G. (2010). 'Graph-theoretic criteria for injectivity and unique equilibria in general chemical reaction systems'. *Advances in Applied Mathematics* **44**(2), 168–184.

Barkai, N. & Leibler, S. (1997). 'Robustness in simple biochemical networks.' *Nature* **387**(6636), 913–917.

Barnes, C. P., Silk, D., Sheng, X., & Stumpf, M. P. H. (2011). 'Bayesian design of synthetic biological systems.' *Proceedings of the National Academy of Sciences of the United States of America* **108**(37), 15190–15195.

Basu, S., Mehreja, R., Thiberge, S., Chen, M.-T., & Weiss, R. (2004). 'Spatiotemporal control of gene expression with pulse-generating networks.' *Proceedings of the National Academy of Sciences of the United States of America* **101**(17), 6355–6360.

Batt, G., Yordanov, B., Weiss, R., & Belta, C. (2007). 'Robustness analysis and tuning of synthetic gene networks.' *Bioinformatics (Oxford, England)* **23**(18), 2415–2422.

Becskei, A. & Serrano, L. (2000). 'Engineering stability in gene networks by autoregulation.' *Nature* **405**(6786), 590–593.

Biancalani, T. & Assaf, M. (2015). 'Genetic Toggle Switch in the Absence of Cooperative Binding: Exact Results'. *Physical review letters*.

Bower, A. G., McClintock, M. K., & Fong, S. S. (2010). 'Synthetic biology: a foundation for multi-scale molecular biology.' *Bioengineered Bugs* **1**(5), 309–312.

Brandman, O., Ferrell, J. E., Li, R., & Meyer, T. (2005). 'Interlinked fast and slow positive feedback loops drive reliable cell decisions.' *Science* **310**(5747), 496–498.

Canton, B., Labno, A., & Endy, D. (2008). 'Refinement and standardization of synthetic biological parts and devices.' *Nature Biotechnology* **26**(7), 787–793.

Chen, B.-S., Chang, C.-H., & Lee, H.-C. (2009). 'Robust synthetic biology design: stochastic game theory approach.' *Bioinformatics (Oxford, England)* **25**(14), 1822–1830.

Cherry, J. L. & Adler, F. R. (2000). 'How to make a biological switch.' *Journal of Theoretical Biology* **203**(2), 117–133.

Chickarmane, V., Enver, T., & Peterson, C. (2009). 'Computational modeling of the hematopoietic erythroid-myeloid switch reveals insights into cooperativity, priming, and irreversibility.' *PLoS Computational Biology* **5**(1), e1000268–e1000268.

Choi, S.-L., Rha, E., Lee, S. J., Kim, H., Kwon, K., Jeong, Y.-S., Rhee, Y. H., Song, J. J., Kim, H.-S., & Lee, S.-G. (2014). 'Toward a generalized and high-throughput enzyme screening system based on artificial genetic circuits.' *ACS Synthetic Biology* **3**(3), 163–171.

Cinquin, O. & Demongeot, J. (2005). 'High-dimensional switches and the modelling of cellular differentiation'. *Journal of Theoretical Biology* **233**(3), 391–411.

Clewley, R. (2012). 'Hybrid models and biological model reduction with PyDSTool.' *PLoS Computational Biology* **8**(8), e1002628–e1002628.

Conradi, C., Flockerzi, D., Raisch, J., & Stelling, J. (2007). 'Subnetwork analysis reveals dynamic features of complex (bio)chemical networks'. *PNAS* **104**(49), 19175–19180.

Cooling, M. T., Rouilly, V., Misirli, G., Lawson, J., Yu, T., Hallinan, J., & Wipat, A. (2010). 'Standard virtual biological parts: a repository of modular modeling components for synthetic biology.' **26**(7), 925–931.

Cormack, B. P., Valdivia, R. H., & Falkow, S. (1996). 'FACS-optimized mutants of the green fluorescent protein (GFP)'. *Gene* **173**(1), 33–38.

De Jong, H. (2002). 'Modeling and simulation of genetic regulatory systems: a literature review.' *Journal of Computational Biology* **9**(1), 67–103.

Díaz, M., Herrero, M., García, L. A., & Quirós, C. (2010). 'Application of flow cytometry to industrial microbial bioprocesses'. *Biochemical Engineering Journal* **48**(3), 385–407.

Doyle, J. & Csete, M. (2005). 'Motifs, Control, and Stability'. *PLoS Biology* **3**(11), e392.

Ellis, B., Gentleman, R., Hahne, F., Le Meur, N., Sarkar, D., & Jiang, M. (2016a). *flowViz: Visualization for flow cytometry*. R package version 1.36.2.

Ellis, B., Haaland, P., Hahne, F., Le Meur, N., Gopalakrishnan, N., Spidlen, J., & Jiang, M. (2016b). *flowCore: flowCore: Basic structures for flow cytometry data*. R package version 1.38.2.

Ellis, T., Wang, X., & Collins, J. J. (2009). 'Diversity-based, model-guided construction of synthetic gene networks with predicted functions.' *Nature Biotechnology* **27**(5), 465–471.

Fasano, G. & Franceschini, A. (1987). 'A multidimensional version of the Kolmogorov-Smirnov test'. *Monthly Notices of the Royal Astronomical Society* **225**(1), 155–170.

Fedorec, A. J. (2016). *autoGate*. https://github.com/ajfedorec/autoGate.git.

Feliu, E. & Wiuf, C. (2013). 'A computational method to preclude multistationarity in networks of interacting species.' *Bioinformatics (Oxford, England)* **29**(18), 2327–2334.

Ferrell Jr, J. E. (2002). 'Self-perpetuating states in signal transduction: positive feedback, double-negative feedback and bistability'. *Current opinion in cell biology* **14**(2), 140–148.

Finney, A., Hucka, M., & Le Novere, N. (2003). 'Systems Biology Markup Language (SBML) level 2: structures and facilities for model definitions'.

Friedland, A. E., Lu, T. K., Wang, X., Shi, D., Church, G., & Collins, J. J. (2009). 'Synthetic gene networks that count.' *Science* **324**(5931), 1199–1202.

Friedman, J. H. & Rafsky, L. C. (1979). 'Multivariate generalizations of the Wald-Wolfowitz and Smirnov two-sample tests'. *The Annals of Statistics*.

Fukunaga, K. (2013). *Introduction to Statistical Pattern Recognition*. Academic Press.

Fung, E., Wong, W. W., Suen, J. K., Bulter, T., Lee, S. G., & Liao, J. C. (2005). 'A synthetic gene–metabolic oscillator '. *Nature* **435**(7038), 118–122.

Gardner, T. S., Cantor, C. R., & Collins, J. J. (2000). 'Construction of a genetic toggle switch in Escherichia coli'. *Nature* **403**(6767), 339–342.

Ghaffarizadeh, A., Flann, N. S., & Podgorski, G. J. (2014). 'Multistable switches and their role in cellular differentiation networks.' *BMC Bioinformatics* **15 Suppl 7**, S7.

Gillespie, D. T. (1977). 'Exact Stochastic Simulation of Coupled Chemical-Reactions'. *Journal of Physical Chemistry* **81**(25), 2340–2361.

Guantes, R. & Poyatos, J. F. (2008). 'Multistable decision switches for flexible control of epigenetic differentiation.' *PLoS Computational Biology* **4**(11), e1000235.

Hafner, M., Koeppl, H., Hasler, M., & Wagner, A. (2009). ''Glocal' Robustness Analysis and Model Discrimination for Circadian Oscillators'. *PLoS Computational Biology* **5**(10), e1000534.

Ham, T. S., Lee, S. K., Keasling, J. D., & Arkin, A. P. (2008). 'Design and Construction of a Double Inversion Recombination Switch for Heritable Sequential Genetic Memory'. *PLoS ONE* **3**(7), e2815.

Heinemann, M. & Panke, S. (2006). 'Synthetic biology–putting engineering into biology'. *Bioinformatics (Oxford, England)* **22**(22), 2790–2799.

Holtz, W. J. & Keasling, J. D. (2010). 'Engineering Static and Dynamic Control of Synthetic Pathways'. *Cell* **140**(1), 19–23.

Hucka, M., Finney, A., Bornstein, B. J., Keating, S. M., Shapiro, B. E., Matthews, J., Kovitz, B. L., Schilstra, M. J., Funahashi, A., Doyle, J. C., & Kitano, H. (2004). 'Evolving a lingua franca and associated software infrastructure for computational systems biology: the Systems Biology Markup Language (SBML) project.' *Systems Biology, IEE Proceedings* **1**(1), 41–53.

Kelly, J. R., Rubin, A. J., Davis, J. H., Ajo-Franklin, C. M., Cumbers, J., Czar, M. J., de Mora, K., Glieberman, A. L., Monie, D. D., & Endy, D. (2009). 'Measuring the activity of BioBrick promoters using an in vivo reference standard.' *Journal of Biological Engineering* **3**(1), 4–4.

Kelwick, R., MacDonald, J. T., Webb, A. J., & Freemont, P. (2014). 'Developments in the tools and methodologies of synthetic biology.' *Frontiers in bioengineering and biotechnology* **2**, 60–60.

Khalil, A. S. & Collins, J. J. (2010). 'Synthetic biology: applications come of age'. *Nature Publishing Group* **11**(5), 367–379.

Kim, J., Bates, D. G., Postlewaite, I., Ma, L., & Iglesias, P. A. (2006). 'Robustness analysis of biochemical network models'. *Systems biology*.

Kirk, D. B. & Hwu, W.-m. W. (2010). *Programming Massively Parallel Processors*. A Hands-on Approach. Burlington: Morgan Kaufmann.

Kitano, H. (2007). 'Towards a theory of biological robustness'. *Molecular systems biology* **3**.

Kobayashi, H., Kaern, M., Araki, M., Chung, K., Gardner, T. S., Cantor, C. R., & Collins, J. J. (2004). 'Programmable cells: interfacing natural and engineered

gene networks.' *Proceedings of the National Academy of Sciences of the United States of America* **101**(22), 8414–8419.

Kolmogorov, A. N. (1933). 'Sulla Determinazione Empirica di Una Legge di Distribuzione'. *Giornale dell'Istituto Italiano degli Attuari* **4**, 83–91.

Kramer, B. P., Viretta, A. U., Daoud-El-Baba, M., Aubel, D., Weber, W., & Fussenegger, M. (2004). 'An engineered epigenetic transgene switch in mammalian cells.' *Nature Biotechnology* **22**(7), 867–870.

Liepe, J., Barnes, C., Cule, E., Erguler, K., Kirk, P., Toni, T., & Stumpf, M. P. H. (2010). 'ABC-SysBio–approximate Bayesian computation in Python with GPU support.' *Bioinformatics (Oxford, England)* **26**(14), 1797–1799.

Liepe, J., Kirk, P., Filippi, S., Toni, T., Barnes, C. P., & Stumpf, M. P. H. (2014). 'A framework for parameter estimation and model selection from experimental data in systems biology using approximate Bayesian computation.' *Nature Protocols* **9**(2), 439–456.

Lillacci, G. & Khammash, M. (2013). 'The signal within the noise: efficient inference of stochastic gene regulation models using fluorescence histograms and stochastic simulations.' *Bioinformatics (Oxford, England)* **29**(18), 2311–2319.

Lipshtat, A., Loinger, A., Balaban, N. Q., & Biham, O. (2006). 'Genetic toggle switch without cooperative binding.' *Physical review letters* **96**(18), 188101.

Litcofsky, K. D., Afeyan, R. B., Krom, R. J., Khalil, A. S., & Collins, J. J. (2012). 'Iterative plug-and-play methodology for constructing and modifying synthetic gene networks.' *Nature Methods* **9**(11), 1077–1080.

Lloyd, S. P. (1982). 'Least squares quantization in PCM'. *Information Theory*.

Loinger, A. & Biham, O. (2009). 'Analysis of genetic toggle switch systems encoded on plasmids.' *Physical review letters* **103**(6), 068104.

Lopes, R. H. C., Reid, I., & Hobson, P. R. (2007). 'The two-dimensional Kolmogorov-Smirnov test'. In: *International Workshop on Advanced Computing and Analysis Techniques in Physics Research*. Amsterdam, 1–12.

Lu, M., Onuchic, J., & Ben-Jacob, E. (2014). 'Construction of an Effective Landscape for Multistate Genetic Switches'. *Physical review letters* **113**(7), 078102.

Lu, T. K., Khalil, A. S., & Collins, J. J. (2009). 'Next-generation synthetic gene networks'. *Nature Biotechnology* **27**(12), 1139–1150.

Lutz, R. & Bujard, H. (1997). 'Independent and tight regulation of transcriptional units in Escherichia coli via the LacR/O, the TetR/O and AraC/I1-I2 regulatory elements.' *Nucleic Acids Research* **25**(6), 1203–1210.

Ma, R., Wang, J., Hou, Z., & Liu, H. (2012). 'Small-number effects: a third stable state in a genetic bistable toggle switch'. *Physical review letters* **109**(24), 248107.

Macdonald, P. J., Chen, Y., & Mueller, J. D. (2012). 'Chromophore maturation and fluorescence fluctuation spectroscopy of fluorescent proteins in a cell-free expression system.' *Analytical Biochemistry* **421**(1), 291–298.

Major, S. (2016). *ndtest*. https://github.com/syrte/ndtest.git.

Marjoram, P., Molitor, J., Plagnol, V., & Tavare, S. (2003). 'Markov chain Monte Carlo without likelihoods.' *Proceedings of the National Academy of Sciences of the United States of America* **100**(26), 15324–15328.

Mathematica (2016). *version 10.3*. Wolfram Research, Inc.

McKay, M. D., Beckman, R. J., & Conover, W. J. (2000). 'A comparison of three methods for selecting values of input variables in the analysis of output from a computer code'. *Technometrics* **42**(1), 55–61.

Milo, R., Jorgensen, P., Moran, U., Weber, G., & Springer, M. (2010). 'BioNumbers–the database of key numbers in molecular and cell biology.' *Nucleic Acids Research* **38**(Database issue), D750–D753.

Monaco, J. V. (2014). 'Classification and Authentication of One-dimensional Behavioral Biometrics'. In: *International Journal of Cognitive Biometrics*, 1–8.

Niwa, H., Toyooka, Y., Shimosato, D., Strumpf, D., Takahashi, K., Yagi, R., & Rossant, J. (2005). 'Interaction between Oct3/4 and Cdx2 Determines Trophectoderm Differentiation'. *Cell* **123**(5), 13–13.

Pedersen, M. G., Bersani, A. M., & Bersani, E. (2007). 'Quasi steady-state approximations in complex intracellular signal transduction networks – a word of caution'. *Journal of Mathematical Chemistry* **43**(4), 1318–1344.

Prill, R. J., Iglesias, P. A., & Levchenko, A. (2005). 'Dynamic properties of network motifs contribute to biological network organization.' *PLoS Biology* **3**(11), e343–e343.

Prindle, A., Samayoa, P., Razinkov, I., Danino, T., Tsimring, L. S., & Hasty, J. (2012). 'A sensing array of radically coupled genetic 'biopixels'.' *Nature* **481**(7379), 39–44.

Pritchard, J. K., Seielstad, M. T., Perez-Lezaun, A., & Feldman, M. W. (1999). 'Population growth of human Y chromosomes: A study of Y chromosome microsatellites'. *Molecular Biology and Evolution* **16**(12), 1791–1798.

Salis, H. M., Mirsky, E. A., & Voigt, C. A. (2009). 'Automated design of synthetic ribosome binding sites to control protein expression.' *Nature Biotechnology* **27**(10), 946–950.

Shaner, N. C., Campbell, R. E., Steinbach, P. A., Giepmans, B. N. G., Palmer, A. E., & Tsien, R. Y. (2004). 'Improved monomeric red, orange and yellow fluorescent proteins derived from Discosoma sp. red fluorescent protein.' *Nature Biotechnology* **22**(12), 1567–1572.

Shapiro, H. M. (1941). *Practical flow cytometry*. Wiley-Liss.

Shimomura, O., Johnson, F. H., & Saiga, Y. (1962). 'Extraction, purification and properties of aequorin, a bioluminescent protein from the luminous hydromedusan, Aequorea.' *Journal of Cellular and Comparative Physiology* **59**, 223–239.

Shinar, G. & Feinberg, M. (2010). 'Structural Sources of Robustness in Biochemical Reaction Networks'. *Science* **327**(5971), 1389–1391.

Stelling, J., Sauer, U., Szallasi, Z., Doyle, F. J., & DOYLE, J. (2004). 'Robustness of cellular functions'. *Cell* **118**(6), 675–685.

Strasser, M., Theis, F. J., & Marr, C. (2012). 'Stability and multiattractor dynamics of a toggle switch based on a two-stage model of stochastic gene expression.' *Biophysical Journal* **102**(1), 19–29.

Stricker, J., Cookson, S., Bennett, M. R., Mather, W. H., Tsimring, L. S., & Hasty, J. (2008). 'A fast, robust and tunable synthetic gene oscillator'. *Nature* **456**(7221), 516–519.

Thomas, R., Thieffry, D., & Kaufman, M. (1995). 'Dynamical behaviour of biological regulatory networks–I. Biological role of feedback loops and practical use of the concept of the loop-characteristic state.' *Bulletin of mathematical biology* **57**(2), 247–276.

Tibshirani, R., Walther, G., & Hastie, T. (2001). 'Estimating the number of clusters in a data set via the gap statistic'. *Journal of the Royal ….*

Tigges, M., Marquez-Lago, T. T., Stelling, J., & Fussenegger, M. (2009). 'A tunable synthetic mammalian oscillator.' *Nature* **457**(7227), 309–312.

Toni, T., Jovanovic, G., Huvet, M., Buck, M., & Stumpf, M. P. H. (2011). 'From qualitative data to quantitative models: analysis of the phage shock protein stress response in Escherichia coli.' *BMC systems biology* **5**, 69–69.

Toni, T., Welch, D., Strelkowa, N., Ipsen, A., & Stumpf, M. P. H. (2009). 'Approximate Bayesian computation scheme for parameter inference and model selection in dynamical systems.' *Journal of the Royal Society, Interface / the Royal Society* **6**(31), 187–202.

Walczak, A. M., Onuchic, J. N., & Wolynes, P. G. (2005). 'Absolute rate theories of epigenetic stability'. *Proceedings of the National Academy of Sciences of the United States of America* **102**(52), 18926–18931.

Wald, A. & Wolfowitz, J. (1940). 'On a test whether two samples are from the same population'. *The Annals of Mathematical Statistics*.

Warren, P. B. & ten Wolde, P. R. (2004). 'Enhancement of the Stability of Genetic Switches by Overlapping Upstream Regulatory Domains'. *Physical review letters* **92**(12), 128101.

Warren, P. B. & ten Wolde, P. R. (2005). 'Chemical models of genetic toggle switches'. *The Journal of Physical Chemistry B* **109**(14), 6812–6823.

Wickham, H. (2009). *ggplot2: Elegant Graphics for Data Analysis*. Springer-Verlag New York. ISBN: 978-0-387-98140-6. URL: http://ggplot2.org.

Wilk, M. B. & Gnanadesikan, R. (1968). 'Probability plotting methods for the analysis of data.' *Biometrika* **55**(1), 1–17.

Woods, M. L., Leon, M., Perez-Carrasco, R., & Barnes, C. P. (2016). 'A Statistical Approach Reveals Designs for the Most Robust Stochastic Gene Oscillators.' *ACS Synthetic Biology* **5**(6), 459–470.

Wu, C.-H., Lee, H.-C., & Chen, B.-S. (2011). 'Robust synthetic gene network design via library-based search method'. *Bioinformatics (Oxford, England)* **27**(19), 2700–2706.

Zamora-Sillero, E., Hafner, M., Ibig, A., Stelling, J., & Wagner, A. (2011). 'Efficient characterization of high-dimensional parameter spaces for systems biology.' *BMC systems biology* **5**, 142.

Zhou, Y., Liepe, J., Sheng, X., Stumpf, M. P. H., & Barnes, C. (2011). 'GPU accelerated biochemical network simulation.' *Bioinformatics (Oxford, England)* **27**(6), 874–876.

*

A Biochemical kinetic models

A.1 Ordinary differential equations

A.1.1 Standard toggle switch with inducers

$$\begin{aligned}
\frac{\mathrm{d}([\mathrm{A}]\cdot V_{\mathrm{cell}})}{\mathrm{d}t} &= +V_{\mathrm{cell}}\cdot(\mathrm{ge}\cdot[\mathrm{gA}]) - V_{\mathrm{cell}}\cdot(\mathrm{deg}\cdot[\mathrm{A}]) - 2\cdot V_{\mathrm{cell}}\cdot(\mathrm{dim}\cdot[\mathrm{A}]\cdot[\mathrm{A}]) \\
&\quad + 2\cdot V_{\mathrm{cell}}\cdot(\mathrm{dim_r}\cdot[\mathrm{A2}]) \\
\frac{\mathrm{d}([\mathrm{gA}]\cdot V_{\mathrm{cell}})}{\mathrm{d}t} &= -V_{\mathrm{cell}}\cdot(\mathrm{rep}\cdot[\mathrm{gA}]\cdot[\mathrm{B2}]) + V_{\mathrm{cell}}\cdot(\mathrm{rep_r}\cdot[\mathrm{B2gA}]) \\
\frac{\mathrm{d}([\mathrm{B}]\cdot V_{\mathrm{cell}})}{\mathrm{d}t} &= +V_{\mathrm{cell}}\cdot(\mathrm{ge}\cdot[\mathrm{gB}]) - V_{\mathrm{cell}}\cdot(\mathrm{deg}\cdot[\mathrm{B}]) - 2\cdot V_{\mathrm{cell}}\cdot(\mathrm{dim}\cdot[\mathrm{B}]\cdot[\mathrm{B}]) \\
&\quad + 2\cdot V_{\mathrm{cell}}\cdot(\mathrm{dim_r}\cdot[\mathrm{B2}]) \\
\frac{\mathrm{d}([\mathrm{gB}]\cdot V_{\mathrm{cell}})}{\mathrm{d}t} &= -V_{\mathrm{cell}}\cdot(\mathrm{rep}\cdot[\mathrm{gB}]\cdot[\mathrm{A2}]) + V_{\mathrm{cell}}\cdot(\mathrm{rep_r}\cdot[\mathrm{A2gB}]) \\
\frac{\mathrm{d}([\mathrm{A2}]\cdot V_{\mathrm{cell}})}{\mathrm{d}t} &= -V_{\mathrm{cell}}\cdot(\mathrm{rep_dim}\cdot[\mathrm{S}]\cdot[\mathrm{A2}]) + V_{\mathrm{cell}}\cdot(\mathrm{rep_dim_r}\cdot[\mathrm{SA2}]) + V_{\mathrm{cell}}\cdot(\mathrm{dim}\cdot[\mathrm{A}]\cdot[\mathrm{A}]) \\
&\quad - V_{\mathrm{cell}}\cdot(\mathrm{dim_r}\cdot[\mathrm{A2}]) - V_{\mathrm{cell}}\cdot(\mathrm{rep}\cdot[\mathrm{gB}]\cdot[\mathrm{A2}]) + V_{\mathrm{cell}}\cdot(\mathrm{rep_r}\cdot[\mathrm{A2gB}]) \\
\frac{\mathrm{d}([\mathrm{B2}]\cdot V_{\mathrm{cell}})}{\mathrm{d}t} &= -V_{\mathrm{cell}}\cdot(\mathrm{rep_dim}\cdot[\mathrm{R}]\cdot[\mathrm{B2}]) + V_{\mathrm{cell}}\cdot(\mathrm{rep_dim_r}\cdot[\mathrm{RB2}]) + V_{\mathrm{cell}}\cdot(\mathrm{dim}\cdot[\mathrm{B}]\cdot[\mathrm{B}]) \\
&\quad - V_{\mathrm{cell}}\cdot(\mathrm{dim_r}\cdot[\mathrm{B2}]) - V_{\mathrm{cell}}\cdot(\mathrm{rep}\cdot[\mathrm{gA}]\cdot[\mathrm{B2}]) + V_{\mathrm{cell}}\cdot(\mathrm{rep_r}\cdot[\mathrm{B2gA}]) \\
\frac{\mathrm{d}([\mathrm{A2gB}]\cdot V_{\mathrm{cell}})}{\mathrm{d}t} &= +V_{\mathrm{cell}}\cdot(\mathrm{rep}\cdot[\mathrm{gB}]\cdot[\mathrm{A2}]) - V_{\mathrm{cell}}\cdot(\mathrm{rep_r}\cdot[\mathrm{A2gB}]) \\
\frac{\mathrm{d}([\mathrm{B2gA}]\cdot V_{\mathrm{cell}})}{\mathrm{d}t} &= +V_{\mathrm{cell}}\cdot(\mathrm{rep}\cdot[\mathrm{gA}]\cdot[\mathrm{B2}]) - V_{\mathrm{cell}}\cdot(\mathrm{rep_r}\cdot[\mathrm{B2gA}])
\end{aligned}$$

$$\frac{\mathrm{d}([\mathrm{S}]\cdot V_{\mathrm{cell}})}{\mathrm{d}t} = -V_{\mathrm{cell}}\cdot\left(\mathrm{rep\_dim}\cdot[\mathrm{S}]\cdot[\mathrm{A2}]\right)+V_{\mathrm{cell}}\cdot\left(\mathrm{rep\_dim\_r}\cdot[\mathrm{SA2}]\right)-V_{\mathrm{cell}}\cdot\left(\mathrm{deg\_sr}\cdot[\mathrm{S}]\right)$$

$$\frac{\mathrm{d}([\mathrm{SA2}]\cdot V_{\mathrm{cell}})}{\mathrm{d}t} = +V_{\mathrm{cell}}\cdot\left(\mathrm{rep\_dim}\cdot[\mathrm{S}]\cdot[\mathrm{A2}]\right)-V_{\mathrm{cell}}\cdot\left(\mathrm{rep\_dim\_r}\cdot[\mathrm{SA2}]\right)$$

$$\frac{\mathrm{d}([\mathrm{R}]\cdot V_{\mathrm{cell}})}{\mathrm{d}t} = -V_{\mathrm{cell}}\cdot\left(\mathrm{rep\_dim}\cdot[\mathrm{R}]\cdot[\mathrm{B2}]\right)+V_{\mathrm{cell}}\cdot\left(\mathrm{rep\_dim\_r}\cdot[\mathrm{RB2}]\right)-V_{\mathrm{cell}}\cdot\left(\mathrm{deg\_sr}\cdot[\mathrm{R}]\right)$$

$$\frac{\mathrm{d}([\mathrm{RB2}]\cdot V_{\mathrm{cell}})}{\mathrm{d}t} = +V_{\mathrm{cell}}\cdot\left(\mathrm{rep\_dim}\cdot[\mathrm{R}]\cdot[\mathrm{B2}]\right)-V_{\mathrm{cell}}\cdot\left(\mathrm{rep\_dim\_r}\cdot[\mathrm{RB2}]\right)$$

Positive autoregulation on B with inducers

$$\frac{\mathrm{d}([\mathrm{A}]\cdot V_{\mathrm{cell}})}{\mathrm{d}t} = +V_{\mathrm{cell}}\cdot(\mathrm{ge}\cdot[\mathrm{gA}]) - 2\cdot V_{\mathrm{cell}}\cdot(\mathrm{dim}\cdot[\mathrm{A}]\cdot[\mathrm{A}]) + 2\cdot V_{\mathrm{cell}}\cdot(\mathrm{dim\_r}\cdot[\mathrm{A2}]) - V_{\mathrm{cell}}\cdot(\mathrm{deg}\cdot[\mathrm{A}])$$

$$\frac{\mathrm{d}([\mathrm{gA}]\cdot V_{\mathrm{cell}})}{\mathrm{d}t} = -V_{\mathrm{cell}}\cdot(\mathrm{rep}\cdot[\mathrm{gA}]\cdot[\mathrm{B2}]) + V_{\mathrm{cell}}\cdot(\mathrm{rep\_r}\cdot[\mathrm{B2gA}])$$

$$\frac{\mathrm{d}([\mathrm{B}]\cdot V_{\mathrm{cell}})}{\mathrm{d}t} = +V_{\mathrm{cell}}\cdot(\mathrm{ge}\cdot[\mathrm{gB}]) - 2\cdot V_{\mathrm{cell}}\cdot(\mathrm{dim}\cdot[\mathrm{B}]\cdot[\mathrm{B}]) + 2\cdot V_{\mathrm{cell}}\cdot(\mathrm{dim\_r}\cdot[\mathrm{B2}]) - V_{\mathrm{cell}}\cdot(\mathrm{deg}\cdot[\mathrm{B}]) + V_{\mathrm{cell}}\cdot(\mathrm{aut\_2}\cdot[\mathrm{B2gB}])$$

$$\frac{\mathrm{d}([\mathrm{gB}]\cdot V_{\mathrm{cell}})}{\mathrm{d}t} = -V_{\mathrm{cell}}\cdot(\mathrm{rep}\cdot[\mathrm{gB}]\cdot[\mathrm{A2}]) + V_{\mathrm{cell}}\cdot(\mathrm{rep\_r}\cdot[\mathrm{A2gB}]) - V_{\mathrm{cell}}\cdot(\mathrm{aut\_1}\cdot[\mathrm{B2}]\cdot[\mathrm{gB}]) + V_{\mathrm{cell}}\cdot(\mathrm{aut\_3}\cdot[\mathrm{B2gB}])$$

$$\frac{\mathrm{d}([\mathrm{A2}]\cdot V_{\mathrm{cell}})}{\mathrm{d}t} = +V_{\mathrm{cell}}\cdot(\mathrm{dim}\cdot[\mathrm{A}]\cdot[\mathrm{A}]) - V_{\mathrm{cell}}\cdot(\mathrm{dim\_r}\cdot[\mathrm{A2}]) - V_{\mathrm{cell}}\cdot(\mathrm{rep}\cdot[\mathrm{gB}]\cdot[\mathrm{A2}]) + V_{\mathrm{cell}}\cdot(\mathrm{rep\_r}\cdot[\mathrm{A2gB}]) - V_{\mathrm{cell}}\cdot(\mathrm{rep\_dim}\cdot[\mathrm{S}]\cdot[\mathrm{A2}]) + V_{\mathrm{cell}}\cdot(\mathrm{rep\_dim\_r}\cdot[\mathrm{SA2}])$$

$$\frac{\mathrm{d}([\mathrm{B2}]\cdot V_{\mathrm{cell}})}{\mathrm{d}t} = +V_{\mathrm{cell}}\cdot(\mathrm{dim}\cdot[\mathrm{B}]\cdot[\mathrm{B}]) - V_{\mathrm{cell}}\cdot(\mathrm{dim\_r}\cdot[\mathrm{B2}]) - V_{\mathrm{cell}}\cdot(\mathrm{rep}\cdot[\mathrm{gA}]\cdot[\mathrm{B2}]) + V_{\mathrm{cell}}\cdot(\mathrm{rep\_r}\cdot[\mathrm{B2gA}]) - V_{\mathrm{cell}}\cdot(\mathrm{aut\_1}\cdot[\mathrm{B2}]\cdot[\mathrm{gB}]) + V_{\mathrm{cell}}\cdot(\mathrm{aut\_3}\cdot[\mathrm{B2gB}]) - V_{\mathrm{cell}}\cdot(\mathrm{rep\_dim}\cdot[\mathrm{R}]\cdot[\mathrm{B2}]) + V_{\mathrm{cell}}\cdot(\mathrm{rep\_dim\_r}\cdot[\mathrm{RB2}])$$

$$\frac{\mathrm{d}([\mathrm{B2gA}]\cdot V_{\mathrm{cell}})}{\mathrm{d}t} = +V_{\mathrm{cell}}\cdot(\mathrm{rep}\cdot[\mathrm{gA}]\cdot[\mathrm{B2}]) - V_{\mathrm{cell}}\cdot(\mathrm{rep\_r}\cdot[\mathrm{B2gA}])$$

$$\frac{\mathrm{d}([\mathrm{A2gB}]\cdot V_{\mathrm{cell}})}{\mathrm{d}t} = +V_{\mathrm{cell}}\cdot(\mathrm{rep}\cdot[\mathrm{gB}]\cdot[\mathrm{A2}]) - V_{\mathrm{cell}}\cdot(\mathrm{rep\_r}\cdot[\mathrm{A2gB}])$$

$$\frac{\mathrm{d}([\mathrm{B2gB}]\cdot V_{\mathrm{cell}})}{\mathrm{d}t} = +V_{\mathrm{cell}}\cdot(\mathrm{aut\_1}\cdot[\mathrm{B2}]\cdot[\mathrm{gB}]) - V_{\mathrm{cell}}\cdot(\mathrm{aut\_3}\cdot[\mathrm{B2gB}])$$

$$\frac{\mathrm{d}([\mathrm{S}]\cdot V_{\mathrm{cell}})}{\mathrm{d}t} = -V_{\mathrm{cell}}\cdot\left(\mathrm{rep\_dim}\cdot[\mathrm{S}]\cdot[\mathrm{A2}]\right)+V_{\mathrm{cell}}\cdot\left(\mathrm{rep\_dim\_r}\cdot[\mathrm{SA2}]\right)-V_{\mathrm{cell}}\cdot\left(\mathrm{deg\_sr}\cdot[\mathrm{S}]\right)$$

$$\frac{\mathrm{d}([\mathrm{SA2}]\cdot V_{\mathrm{cell}})}{\mathrm{d}t} = +V_{\mathrm{cell}}\cdot\left(\mathrm{rep\_dim}\cdot[\mathrm{S}]\cdot[\mathrm{A2}]\right)-V_{\mathrm{cell}}\cdot\left(\mathrm{rep\_dim\_r}\cdot[\mathrm{SA2}]\right)$$

$$\frac{\mathrm{d}([\mathrm{R}]\cdot V_{\mathrm{cell}})}{\mathrm{d}t} = -V_{\mathrm{cell}}\cdot\left(\mathrm{rep\_dim}\cdot[\mathrm{R}]\cdot[\mathrm{B2}]\right)+V_{\mathrm{cell}}\cdot\left(\mathrm{rep\_dim\_r}\cdot[\mathrm{RB2}]\right)-V_{\mathrm{cell}}\cdot\left(\mathrm{deg\_sr}\cdot[\mathrm{R}]\right)$$

$$\frac{\mathrm{d}([\mathrm{RB2}]\cdot V_{\mathrm{cell}})}{\mathrm{d}t} = +V_{\mathrm{cell}}\cdot\left(\mathrm{rep\_dim}\cdot[\mathrm{R}]\cdot[\mathrm{B2}]\right)-V_{\mathrm{cell}}\cdot\left(\mathrm{rep\_dim\_r}\cdot[\mathrm{RB2}]\right)$$

Positive autoregulation on A with inducers

$$\frac{\mathrm{d}([\mathrm{A}]\cdot V_{\mathrm{cell}})}{\mathrm{d}t} = +V_{\mathrm{cell}}\cdot(\mathrm{aut\_2}\cdot[\mathrm{A2gA}]) - V_{\mathrm{cell}}\cdot(\mathrm{deg}\cdot[\mathrm{A}]) + 2\cdot V_{\mathrm{cell}}\cdot(\mathrm{dim\_r}\cdot[\mathrm{A2}]) - 2\cdot V_{\mathrm{cell}}\cdot(\mathrm{dim}\cdot[\mathrm{A}]\cdot[\mathrm{A}]) + V_{\mathrm{cell}}\cdot(\mathrm{ge}\cdot[\mathrm{gA}])$$

$$\frac{\mathrm{d}([\mathrm{gA}]\cdot V_{\mathrm{cell}})}{\mathrm{d}t} = +V_{\mathrm{cell}}\cdot(\mathrm{aut\_3}\cdot[\mathrm{A2gA}]) - V_{\mathrm{cell}}\cdot(\mathrm{aut\_1}\cdot[\mathrm{A2}]\cdot[\mathrm{gA}]) + V_{\mathrm{cell}}\cdot(\mathrm{rep\_r}\cdot[\mathrm{B2gA}]) - V_{\mathrm{cell}}\cdot(\mathrm{rep}\cdot[\mathrm{gA}]\cdot[\mathrm{B2}])$$

$$\frac{\mathrm{d}([\mathrm{B}]\cdot V_{\mathrm{cell}})}{\mathrm{d}t} = -V_{\mathrm{cell}}\cdot(\mathrm{deg}\cdot[\mathrm{B}]) + 2\cdot V_{\mathrm{cell}}\cdot(\mathrm{dim\_r}\cdot[\mathrm{B2}]) - 2\cdot V_{\mathrm{cell}}\cdot(\mathrm{dim}\cdot[\mathrm{B}]\cdot[\mathrm{B}]) + V_{\mathrm{cell}}\cdot(\mathrm{ge}\cdot[\mathrm{gB}])$$

$$\frac{\mathrm{d}([\mathrm{gB}]\cdot V_{\mathrm{cell}})}{\mathrm{d}t} = +V_{\mathrm{cell}}\cdot(\mathrm{rep\_r}\cdot[\mathrm{A2gB}]) - V_{\mathrm{cell}}\cdot(\mathrm{rep}\cdot[\mathrm{gB}]\cdot[\mathrm{A2}])$$

$$\frac{\mathrm{d}([\mathrm{A2}]\cdot V_{\mathrm{cell}})}{\mathrm{d}t} = +V_{\mathrm{cell}}\cdot(\mathrm{rep\_dim\_r}\cdot[\mathrm{SA2}]) - V_{\mathrm{cell}}\cdot(\mathrm{rep\_dim}\cdot[\mathrm{S}]\cdot[\mathrm{A2}]) + V_{\mathrm{cell}}\cdot(\mathrm{aut\_3}\cdot[\mathrm{A2gA}]) - V_{\mathrm{cell}}\cdot(\mathrm{aut\_1}\cdot[\mathrm{A2}]\cdot[\mathrm{gA}]) + V_{\mathrm{cell}}\cdot(\mathrm{rep\_r}\cdot[\mathrm{A2gB}]) - V_{\mathrm{cell}}\cdot(\mathrm{rep}\cdot[\mathrm{gB}]\cdot[\mathrm{A2}]) - V_{\mathrm{cell}}\cdot(\mathrm{dim\_r}\cdot[\mathrm{A2}]) + V_{\mathrm{cell}}\cdot(\mathrm{dim}\cdot[\mathrm{A}]\cdot[\mathrm{A}])$$

$$\frac{\mathrm{d}([\mathrm{B2}]\cdot V_{\mathrm{cell}})}{\mathrm{d}t} = +V_{\mathrm{cell}}\cdot(\mathrm{rep\_dim\_r}\cdot[\mathrm{RB2}]) - V_{\mathrm{cell}}\cdot(\mathrm{rep\_dim}\cdot[\mathrm{R}]\cdot[\mathrm{B2}]) + V_{\mathrm{cell}}\cdot(\mathrm{rep\_r}\cdot[\mathrm{B2gA}]) - V_{\mathrm{cell}}\cdot(\mathrm{rep}\cdot[\mathrm{gA}]\cdot[\mathrm{B2}]) - V_{\mathrm{cell}}\cdot(\mathrm{dim\_r}\cdot[\mathrm{B2}]) + V_{\mathrm{cell}}\cdot(\mathrm{dim}\cdot[\mathrm{B}]\cdot[\mathrm{B}])$$

$$\frac{\mathrm{d}([\mathrm{B2gA}]\cdot V_{\mathrm{cell}})}{\mathrm{d}t} = -V_{\mathrm{cell}}\cdot(\mathrm{rep\_r}\cdot[\mathrm{B2gA}]) + V_{\mathrm{cell}}\cdot(\mathrm{rep}\cdot[\mathrm{gA}]\cdot[\mathrm{B2}])$$

$$\frac{\mathrm{d}([\mathrm{A2gB}]\cdot V_{\mathrm{cell}})}{\mathrm{d}t} = -V_{\mathrm{cell}}\cdot(\mathrm{rep\_r}\cdot[\mathrm{A2gB}]) + V_{\mathrm{cell}}\cdot(\mathrm{rep}\cdot[\mathrm{gB}]\cdot[\mathrm{A2}])$$

$$\frac{\mathrm{d}([\mathrm{A2gA}]\cdot V_{\mathrm{cell}})}{\mathrm{d}t} = -V_{\mathrm{cell}}\cdot(\mathrm{aut\_3}\cdot[\mathrm{A2gA}]) + V_{\mathrm{cell}}\cdot(\mathrm{aut\_1}\cdot[\mathrm{A2}]\cdot[\mathrm{gA}])$$

$$\frac{\mathrm{d}([\mathrm{S}]\cdot V_{\mathrm{cell}})}{\mathrm{d}t} = -V_{\mathrm{cell}}\cdot(\mathrm{deg\_sr}\cdot[\mathrm{S}]) + V_{\mathrm{cell}}\cdot(\mathrm{rep\_dim\_r}\cdot[\mathrm{SA2}]) - V_{\mathrm{cell}}\cdot(\mathrm{rep\_dim}\cdot[\mathrm{S}]\cdot[\mathrm{A2}])$$

$$\frac{\mathrm{d}([\mathrm{SA2}]\cdot V_{\mathrm{cell}})}{\mathrm{d}t} = -V_{\mathrm{cell}}\cdot(\mathrm{rep\_dim\_r}\cdot[\mathrm{SA2}]) + V_{\mathrm{cell}}\cdot(\mathrm{rep\_dim}\cdot[\mathrm{S}]\cdot[\mathrm{A2}])$$

$$\frac{\mathrm{d}([\mathrm{R}]\cdot V_{\mathrm{cell}})}{\mathrm{d}t} = -V_{\mathrm{cell}}\cdot(\mathrm{deg\_sr}\cdot[\mathrm{R}]) + V_{\mathrm{cell}}\cdot(\mathrm{rep\_dim\_r}\cdot[\mathrm{RB2}]) - V_{\mathrm{cell}}\cdot(\mathrm{rep\_dim}\cdot[\mathrm{R}]\cdot[\mathrm{B2}])$$

$$\frac{\mathrm{d}([\mathrm{RB2}]\cdot V_{\mathrm{cell}})}{\mathrm{d}t} = -V_{\mathrm{cell}}\cdot(\mathrm{rep\_dim\_r}\cdot[\mathrm{RB2}]) + V_{\mathrm{cell}}\cdot(\mathrm{rep\_dim}\cdot[\mathrm{R}]\cdot[\mathrm{B2}])$$

A.1.2 Positive autoregulation on A and B with inducers

$$
\begin{aligned}
\frac{\mathrm{d}([\mathrm{A}]\cdot V_{\mathrm{cell}})}{\mathrm{d}t} &= +2\cdot V_{\mathrm{cell}}\cdot(\mathrm{dim\_r}\cdot[\mathrm{A2}]) - 2\cdot V_{\mathrm{cell}}\cdot(\mathrm{dim}\cdot[\mathrm{A}]\cdot[\mathrm{A}]) + V_{\mathrm{cell}}\cdot(\mathrm{ge}\cdot[\mathrm{gA}]) \\
&\quad + V_{\mathrm{cell}}\cdot(\mathrm{aut\_2}\cdot[\mathrm{A2gA}]) - V_{\mathrm{cell}}\cdot(\mathrm{deg}\cdot[\mathrm{A}]) \\
\frac{\mathrm{d}([\mathrm{gA}]\cdot V_{\mathrm{cell}})}{\mathrm{d}t} &= +V_{\mathrm{cell}}\cdot(\mathrm{aut\_3}\cdot[\mathrm{A2gA}]) + V_{\mathrm{cell}}\cdot(\mathrm{rep\_r}\cdot[\mathrm{B2gA}]) - V_{\mathrm{cell}}\cdot(\mathrm{rep}\cdot[\mathrm{gA}]\cdot[\mathrm{B2}]) \\
&\quad - V_{\mathrm{cell}}\cdot(\mathrm{aut\_1}\cdot[\mathrm{A2}]\cdot[\mathrm{gA}]) \\
\frac{\mathrm{d}([\mathrm{B}]\cdot V_{\mathrm{cell}})}{\mathrm{d}t} &= +2\cdot V_{\mathrm{cell}}\cdot(\mathrm{dim\_r}\cdot[\mathrm{B2}]) - 2\cdot V_{\mathrm{cell}}\cdot(\mathrm{dim}\cdot[\mathrm{B}]\cdot[\mathrm{B}]) + V_{\mathrm{cell}}\cdot(\mathrm{ge}\cdot[\mathrm{gB}]) \\
&\quad + V_{\mathrm{cell}}\cdot(\mathrm{aut\_2}\cdot[\mathrm{B2gB}]) - V_{\mathrm{cell}}\cdot(\mathrm{deg}\cdot[\mathrm{B}]) \\
\frac{\mathrm{d}([\mathrm{gB}]\cdot V_{\mathrm{cell}})}{\mathrm{d}t} &= -V_{\mathrm{cell}}\cdot(\mathrm{rep}\cdot[\mathrm{gB}]\cdot[\mathrm{A2}]) + V_{\mathrm{cell}}\cdot(\mathrm{aut\_3}\cdot[\mathrm{B2gB}]) - V_{\mathrm{cell}}\cdot(\mathrm{aut\_1}\cdot[\mathrm{B2}]\cdot[\mathrm{gB}]) \\
&\quad + V_{\mathrm{cell}}\cdot(\mathrm{rep\_r}\cdot[\mathrm{A2gB}]) \\
\frac{\mathrm{d}([\mathrm{A2}]\cdot V_{\mathrm{cell}})}{\mathrm{d}t} &= +V_{\mathrm{cell}}\cdot(\mathrm{aut\_3}\cdot[\mathrm{A2gA}]) - V_{\mathrm{cell}}\cdot(\mathrm{rep}\cdot[\mathrm{gB}]\cdot[\mathrm{A2}]) - V_{\mathrm{cell}}\cdot(\mathrm{dim\_r}\cdot[\mathrm{A2}]) \\
&\quad + V_{\mathrm{cell}}\cdot(\mathrm{dim}\cdot[\mathrm{A}]\cdot[\mathrm{A}]) - V_{\mathrm{cell}}\cdot(\mathrm{aut\_1}\cdot[\mathrm{A2}]\cdot[\mathrm{gA}]) - V_{\mathrm{cell}}\cdot(\mathrm{rep\_dim}\cdot[\mathrm{S}]\cdot[\mathrm{A2}]) \\
&\quad + V_{\mathrm{cell}}\cdot(\mathrm{rep\_dim\_r}\cdot[\mathrm{SA2}]) + V_{\mathrm{cell}}\cdot(\mathrm{rep\_r}\cdot[\mathrm{A2gB}]) \\
\frac{\mathrm{d}([\mathrm{B2}]\cdot V_{\mathrm{cell}})}{\mathrm{d}t} &= +V_{\mathrm{cell}}\cdot(\mathrm{rep\_r}\cdot[\mathrm{B2gA}]) - V_{\mathrm{cell}}\cdot(\mathrm{rep}\cdot[\mathrm{gA}]\cdot[\mathrm{B2}]) - V_{\mathrm{cell}}\cdot(\mathrm{dim\_r}\cdot[\mathrm{B2}]) \\
&\quad + V_{\mathrm{cell}}\cdot(\mathrm{dim}\cdot[\mathrm{B}]\cdot[\mathrm{B}]) + V_{\mathrm{cell}}\cdot(\mathrm{aut\_3}\cdot[\mathrm{B2gB}]) - V_{\mathrm{cell}}\cdot(\mathrm{rep\_dim}\cdot[\mathrm{R}]\cdot[\mathrm{B2}]) \\
&\quad + V_{\mathrm{cell}}\cdot(\mathrm{rep\_dim\_r}\cdot[\mathrm{RB2}]) - V_{\mathrm{cell}}\cdot(\mathrm{aut\_1}\cdot[\mathrm{B2}]\cdot[\mathrm{gB}]) \\
\frac{\mathrm{d}([\mathrm{B2gA}]\cdot V_{\mathrm{cell}})}{\mathrm{d}t} &= -V_{\mathrm{cell}}\cdot(\mathrm{rep\_r}\cdot[\mathrm{B2gA}]) + V_{\mathrm{cell}}\cdot(\mathrm{rep}\cdot[\mathrm{gA}]\cdot[\mathrm{B2}])
\end{aligned}
$$

$$\frac{\mathrm{d}([\mathrm{A2gB}]\cdot V_{\mathrm{cell}})}{\mathrm{d}t} = +V_{\mathrm{cell}}\cdot(\mathrm{rep}\cdot[\mathrm{gB}]\cdot[\mathrm{A2}]) - V_{\mathrm{cell}}\cdot(\mathrm{rep\_r}\cdot[\mathrm{A2gB}])$$

$$\frac{\mathrm{d}([\mathrm{B2gB}]\cdot V_{\mathrm{cell}})}{\mathrm{d}t} = -V_{\mathrm{cell}}\cdot(\mathrm{aut\_3}\cdot[\mathrm{B2gB}]) + V_{\mathrm{cell}}\cdot(\mathrm{aut\_1}\cdot[\mathrm{B2}]\cdot[\mathrm{gB}])$$

$$\frac{\mathrm{d}([\mathrm{A2gA}]\cdot V_{\mathrm{cell}})}{\mathrm{d}t} = -V_{\mathrm{cell}}\cdot(\mathrm{aut\_3}\cdot[\mathrm{A2gA}]) + V_{\mathrm{cell}}\cdot(\mathrm{aut\_1}\cdot[\mathrm{A2}]\cdot[\mathrm{gA}])$$

$$\frac{\mathrm{d}([\mathrm{S}]\cdot V_{\mathrm{cell}})}{\mathrm{d}t} = -V_{\mathrm{cell}}\cdot(\mathrm{rep\_dim}\cdot[\mathrm{S}]\cdot[\mathrm{A2}]) + V_{\mathrm{cell}}\cdot(\mathrm{rep\_dim\_r}\cdot[\mathrm{SA2}]) - V_{\mathrm{cell}}\cdot(\mathrm{deg\_sr}\cdot[\mathrm{S}])$$

$$\frac{\mathrm{d}([\mathrm{SA2}]\cdot V_{\mathrm{cell}})}{\mathrm{d}t} = +V_{\mathrm{cell}}\cdot(\mathrm{rep\_dim}\cdot[\mathrm{S}]\cdot[\mathrm{A2}]) - V_{\mathrm{cell}}\cdot(\mathrm{rep\_dim\_r}\cdot[\mathrm{SA2}])$$

$$\frac{\mathrm{d}([\mathrm{R}]\cdot V_{\mathrm{cell}})}{\mathrm{d}t} = -V_{\mathrm{cell}}\cdot(\mathrm{rep\_dim}\cdot[\mathrm{R}]\cdot[\mathrm{B2}]) + V_{\mathrm{cell}}\cdot(\mathrm{rep\_dim\_r}\cdot[\mathrm{RB2}]) - V_{\mathrm{cell}}\cdot(\mathrm{deg\_sr}\cdot[\mathrm{R}])$$

$$\frac{\mathrm{d}([\mathrm{RB2}]\cdot V_{\mathrm{cell}})}{\mathrm{d}t} = +V_{\mathrm{cell}}\cdot(\mathrm{rep\_dim}\cdot[\mathrm{R}]\cdot[\mathrm{B2}]) - V_{\mathrm{cell}}\cdot(\mathrm{rep\_dim\_r}\cdot[\mathrm{RB2}])$$

A.1.3 CS-MA

$$\frac{\mathrm{d}([\mathrm{A}]\cdot V_{\mathrm{cell}})}{\mathrm{d}t} = +2\cdot V_{\mathrm{cell}}\cdot(\mathrm{dim\_r}\cdot[\mathrm{A2}]) - 2\cdot V_{\mathrm{cell}}\cdot(\mathrm{dim}\cdot[\mathrm{A}]\cdot[\mathrm{A}]) + V_{\mathrm{cell}}\cdot(\mathrm{geA}\cdot[\mathrm{gA}]) - V_{\mathrm{cell}}\cdot(\mathrm{de}$$

$$\frac{\mathrm{d}([\mathrm{gA}]\cdot V_{\mathrm{cell}})}{\mathrm{d}t} = +V_{\mathrm{cell}}\cdot(\mathrm{rep\_r}\cdot[\mathrm{B2gA}]) - V_{\mathrm{cell}}\cdot(\mathrm{repA}\cdot[\mathrm{gA}]\cdot[\mathrm{B2}])$$

$$\frac{\mathrm{d}([\mathrm{B}]\cdot V_{\mathrm{cell}})}{\mathrm{d}t} = +2\cdot V_{\mathrm{cell}}\cdot(\mathrm{dim\_r}\cdot[\mathrm{B2}]) - 2\cdot V_{\mathrm{cell}}\cdot(\mathrm{dim}\cdot[\mathrm{B}]\cdot[\mathrm{B}]) + V_{\mathrm{cell}}\cdot(\mathrm{geB}\cdot[\mathrm{gB}]) - V_{\mathrm{cell}}\cdot(\mathrm{deg}$$

$$\frac{\mathrm{d}([\mathrm{gB}]\cdot V_{\mathrm{cell}})}{\mathrm{d}t} = +V_{\mathrm{cell}}\cdot(\mathrm{rep\_r}\cdot[\mathrm{A2gB}]) - V_{\mathrm{cell}}\cdot(\mathrm{repB}\cdot[\mathrm{gB}]\cdot[\mathrm{A2}])$$

$$\frac{\mathrm{d}([\mathrm{A2}]\cdot V_{\mathrm{cell}})}{\mathrm{d}t} = -V_{\mathrm{cell}}\cdot(\mathrm{dim\_r}\cdot[\mathrm{A2}]) + V_{\mathrm{cell}}\cdot(\mathrm{dim}\cdot[\mathrm{A}]\cdot[\mathrm{A}]) - V_{\mathrm{cell}}\cdot(\mathrm{deg\_dim}\cdot[\mathrm{A2}]) + V_{\mathrm{cell}}\cdot(\mathrm{rep\_}$$
$$- V_{\mathrm{cell}}\cdot(\mathrm{repB}\cdot[\mathrm{gB}]\cdot[\mathrm{A2}])$$

$$\frac{\mathrm{d}([\mathrm{B2}]\cdot V_{\mathrm{cell}})}{\mathrm{d}t} = -V_{\mathrm{cell}}\cdot(\mathrm{dim\_r}\cdot[\mathrm{B2}]) + V_{\mathrm{cell}}\cdot(\mathrm{dim}\cdot[\mathrm{B}]\cdot[\mathrm{B}]) - V_{\mathrm{cell}}\cdot(\mathrm{deg\_dim}\cdot[\mathrm{B2}]) + V_{\mathrm{cell}}\cdot(\mathrm{rep\_r}$$
$$- V_{\mathrm{cell}}\cdot(\mathrm{repA}\cdot[\mathrm{gA}]\cdot[\mathrm{B2}])$$

$$\frac{\mathrm{d}([\mathrm{A2gB}]\cdot V_{\mathrm{cell}})}{\mathrm{d}t} = -V_{\mathrm{cell}}\cdot(\mathrm{rep\_r}\cdot[\mathrm{A2gB}]) + V_{\mathrm{cell}}\cdot(\mathrm{repB}\cdot[\mathrm{gB}]\cdot[\mathrm{A2}])$$

$$\frac{\mathrm{d}([\mathrm{B2gA}]\cdot V_{\mathrm{cell}})}{\mathrm{d}t} = -V_{\mathrm{cell}}\cdot(\mathrm{rep\_r}\cdot[\mathrm{B2gA}]) + V_{\mathrm{cell}}\cdot(\mathrm{repA}\cdot[\mathrm{gA}]\cdot[\mathrm{B2}])$$

The CS-MA switch was simulated using stochastic dynamics. The stoichimetry matrix and hazards defining the model are shown below:

Table A.1 CS-MA stoichiometry matrix

1.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
0.0	0.0	1.0	0.0	0.0	0.0	0.0	0.0
-2.0	0.0	0.0	0.0	1.0	0.0	0.0	0.0
0.0	0.0	-2.0	0.0	0.0	1.0	0.0	0.0
0.0	0.0	2.0	0.0	0.0	-1.0	0.0	0.0
2.0	0.0	0.0	0.0	-1.0	0.0	0.0	0.0
0.0	-1.0	0.0	0.0	0.0	-1.0	0.0	1.0
0.0	1.0	0.0	0.0	0.0	1.0	0.0	-1.0
0.0	0.0	0.0	-1.0	-1.0	0.0	1.0	0.0
0.0	0.0	0.0	1.0	1.0	0.0	-1.0	0.0
-1.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
0.0	0.0	-1.0	0.0	0.0	0.0	0.0	0.0
0.0	0.0	0.0	0.0	-1.0	0.0	0.0	0.0
0.0	0.0	0.0	0.0	0.0	-1.0	0.0	0.0

$$h[1] = geA \times gA$$
$$h[2] = geB \times gB$$
$$h[3] = dim \times A^2$$
$$h[4] = dim \times B^2$$
$$h[5] = dim\_r \times B2$$
$$h[6] = dim\_r \times A2$$
$$h[7] = geA \times gA \times B2$$
$$h[8] = rep\_r \times B2gA$$
$$h[9] = repB \times gB \times A2$$
$$h[10] = rep\_r \times A2gB$$
$$h[11] = deg \times A$$
$$h[12] = deg \times B$$
$$h[13] = deg\_dim \times A2$$
$$h[14] = deg\_dim \times B2$$

A.1.4 DP-MA

$$
\begin{aligned}
\frac{\mathrm{d}([\mathrm{A}]\cdot V_{\mathrm{cell}})}{\mathrm{d}t} &= -V_{\mathrm{cell}}\cdot(\mathrm{deg}\cdot[\mathrm{A}]) + 2\cdot V_{\mathrm{cell}}\cdot(\mathrm{dim\_r}\cdot[\mathrm{A2}]) - 2\cdot V_{\mathrm{cell}}\cdot(\mathrm{dim}\cdot[\mathrm{A}]\cdot[\mathrm{A}]) + V_{\mathrm{cell}}\cdot(\mathrm{geA} \\
&\quad + V_{\mathrm{cell}}\cdot(\mathrm{aut\_2}\cdot[\mathrm{A2gA}]) \\
\frac{\mathrm{d}([\mathrm{gA}]\cdot V_{\mathrm{cell}})}{\mathrm{d}t} &= -V_{\mathrm{cell}}\cdot(\mathrm{aut\_1}\cdot[\mathrm{A2}]\cdot[\mathrm{gA}]) + V_{\mathrm{cell}}\cdot(\mathrm{rep\_r}\cdot[\mathrm{B2gA}]) - V_{\mathrm{cell}}\cdot(\mathrm{repA}\cdot[\mathrm{gA}]\cdot[\mathrm{B2}]) \\
&\quad + V_{\mathrm{cell}}\cdot(\mathrm{aut\_3}\cdot[\mathrm{A2gA}]) \\
\frac{\mathrm{d}([\mathrm{B}]\cdot V_{\mathrm{cell}})}{\mathrm{d}t} &= +V_{\mathrm{cell}}\cdot(\mathrm{aut\_2}\cdot[\mathrm{B2gB}]) - V_{\mathrm{cell}}\cdot(\mathrm{deg}\cdot[\mathrm{B}]) + 2\cdot V_{\mathrm{cell}}\cdot(\mathrm{dim\_r}\cdot[\mathrm{B2}]) - 2\cdot V_{\mathrm{cell}}\cdot(\mathrm{dim} \\
&\quad + V_{\mathrm{cell}}\cdot(\mathrm{geB}\cdot[\mathrm{gB}]) \\
\frac{\mathrm{d}([\mathrm{gB}]\cdot V_{\mathrm{cell}})}{\mathrm{d}t} &= +V_{\mathrm{cell}}\cdot(\mathrm{aut\_3}\cdot[\mathrm{B2gB}]) - V_{\mathrm{cell}}\cdot(\mathrm{aut\_1}\cdot[\mathrm{B2}]\cdot[\mathrm{gB}]) + V_{\mathrm{cell}}\cdot(\mathrm{rep\_r}\cdot[\mathrm{A2gB}]) \\
&\quad - V_{\mathrm{cell}}\cdot(\mathrm{repB}\cdot[\mathrm{gB}]\cdot[\mathrm{A2}]) \\
\frac{\mathrm{d}([\mathrm{A2}]\cdot V_{\mathrm{cell}})}{\mathrm{d}t} &= -V_{\mathrm{cell}}\cdot(\mathrm{aut\_1}\cdot[\mathrm{A2}]\cdot[\mathrm{gA}]) + V_{\mathrm{cell}}\cdot(\mathrm{rep\_r}\cdot[\mathrm{A2gB}]) - V_{\mathrm{cell}}\cdot(\mathrm{repB}\cdot[\mathrm{gB}]\cdot[\mathrm{A2}]) - V_{\mathrm{ce}} \\
&\quad + V_{\mathrm{cell}}\cdot(\mathrm{dim}\cdot[\mathrm{A}]\cdot[\mathrm{A}]) + V_{\mathrm{cell}}\cdot(\mathrm{aut\_3}\cdot[\mathrm{A2gA}]) \\
&\quad - V_{\mathrm{cell}}\cdot(\mathrm{deg\_dim}\cdot[\mathrm{A2}]) \\
\frac{\mathrm{d}([\mathrm{B2}]\cdot V_{\mathrm{cell}})}{\mathrm{d}t} &= +V_{\mathrm{cell}}\cdot(\mathrm{aut\_3}\cdot[\mathrm{B2gB}]) - V_{\mathrm{cell}}\cdot(\mathrm{aut\_1}\cdot[\mathrm{B2}]\cdot[\mathrm{gB}]) + V_{\mathrm{cell}}\cdot(\mathrm{rep\_r}\cdot[\mathrm{B2gA}]) \\
&\quad - V_{\mathrm{cell}}\cdot(\mathrm{repA}\cdot[\mathrm{gA}]\cdot[\mathrm{B2}]) - V_{\mathrm{cell}}\cdot(\mathrm{dim\_r}\cdot[\mathrm{B2}]) + V_{\mathrm{cell}}\cdot(\mathrm{dim}\cdot[\mathrm{B}]\cdot[\mathrm{B}]) - V_{\mathrm{cell}}\cdot(\mathrm{de} \\
\frac{\mathrm{d}([\mathrm{B2gA}]\cdot V_{\mathrm{cell}})}{\mathrm{d}t} &= -V_{\mathrm{cell}}\cdot(\mathrm{rep\_r}\cdot[\mathrm{B2gA}]) + V_{\mathrm{cell}}\cdot(\mathrm{repA}\cdot[\mathrm{gA}]\cdot[\mathrm{B2}]) \\
\frac{\mathrm{d}([\mathrm{A2gB}]\cdot V_{\mathrm{cell}})}{\mathrm{d}t} &= -V_{\mathrm{cell}}\cdot(\mathrm{rep\_r}\cdot[\mathrm{A2gB}]) + V_{\mathrm{cell}}\cdot(\mathrm{repB}\cdot[\mathrm{gB}]\cdot[\mathrm{A2}]) \\
\frac{\mathrm{d}([\mathrm{B2gB}]\cdot V_{\mathrm{cell}})}{\mathrm{d}t} &= -V_{\mathrm{cell}}\cdot(\mathrm{aut\_3}\cdot[\mathrm{B2gB}]) + V_{\mathrm{cell}}\cdot(\mathrm{aut\_1}\cdot[\mathrm{B2}]\cdot[\mathrm{gB}]) \\
\frac{\mathrm{d}([\mathrm{A2gA}]\cdot V_{\mathrm{cell}})}{\mathrm{d}t} &= +V_{\mathrm{cell}}\cdot(\mathrm{aut\_1}\cdot[\mathrm{A2}]\cdot[\mathrm{gA}]) - V_{\mathrm{cell}}\cdot(\mathrm{aut\_3}\cdot[\mathrm{A2gA}])
\end{aligned}
$$

The stoichiometry matrix and hazards defining the DP-MA model are given be-

low.

Table A.2 DP-MA stoichiometry matrix

1.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
0.0	0.0	1.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
-2.0	0.0	0.0	0.0	1.0	0.0	0.0	0.0	0.0	0.0
0.0	0.0	-2.0	0.0	0.0	1.0	0.0	0.0	0.0	0.0
0.0	0.0	2.0	0.0	0.0	-1.0	0.0	0.0	0.0	0.0
2.0	0.0	0.0	0.0	-1.0	0.0	0.0	0.0	0.0	0.0
0.0	-1.0	0.0	0.0	0.0	-1.0	1.0	0.0	0.0	0.0
0.0	1.0	0.0	0.0	0.0	1.0	-1.0	0.0	0.0	0.0
0.0	0.0	0.0	-1.0	-1.0	0.0	0.0	1.0	0.0	0.0
0.0	0.0	0.0	1.0	1.0	0.0	0.0	-1.0	0.0	0.0
-1.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
0.0	0.0	-1.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
0.0	0.0	0.0	-1.0	0.0	-1.0	0.0	0.0	1.0	0.0
0.0	0.0	1.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
0.0	0.0	0.0	1.0	0.0	1.0	0.0	0.0	-1.0	0.0
0.0	-1.0	0.0	0.0	-1.0	0.0	0.0	0.0	0.0	1.0
1.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
0.0	1.0	0.0	0.0	1.0	0.0	0.0	0.0	0.0	-1.0
0.0	0.0	0.0	0.0	-1.0	0.0	0.0	0.0	0.0	0.0
0.0	0.0	0.0	0.0	0.0	-1.0	0.0	0.0	0.0	0.0

$$h[1] = repA \times gA$$

$$h[2] = repB \times gB$$

$$h[3] = dim \times A^2$$

$$h[4] = dim\_r \times B^2$$

$$h[5] = dim\_r \times B2$$

$$h[6] = deg \times A2$$

$$h[7] = rep\_r \times gA \times B2$$

$$h[8] = rep\_r \times B2gA$$

$$h[9] = repB \times gB \times A2$$

$$h[10] = dim \times A2gB$$

$$h[11] = deg \times A$$

$$h[12] = deg \times B$$

$$h[13] = aut\_1 \times B2 \times gB$$

$$h[14] = aut\_2 \times B2gB$$

$$h[15] = aut\_3 \times B2gB$$

$$h[16] = aut\_1 \times A2 \times gA$$

$$h[17] = aut\_2 \times A2gA$$

$$h[18] = aut\_3 \times A2gA$$

$$h[19] = deg\_dim \times A2$$

$$h[20] = deg\_dim \times B2$$

B Primers

B.1 Primers used during PCR and sequencing

Table B.1 List of primers used for PCR amplification

Direction	Amplifies	Added site	Sequence
Forward	Plac/ara-1	NcoI	TAAGCACCATGGCTCGAGCATAGCATTTTTATCCAT
Reverse	Plac/ara-1	SalI	AAGCAGGTCGACTTCTGTGTGAAATTGTTATCCGC
Forward	Plac/ara-1	XmaI	TAAGCACCCGGGCTCGAGCATAGCATTTTTATCCAT
Reverse	Plac/ara-1	KasI	AAGCAGGGCGCCCTTTCTCCTCTTTAATGAATTCTGTGT
Forward	Plac/ara-1, AraC	EagI	TAAGCACGGCCGCTCGAGCATAGCATTTTTATC
Reverse	Plac/ara-1, AraC	SalI	AAGCAGGTCGACCTAATTAAGCTTTCACGCTG
Forward	LuxR	BsGI	TAAGCATGTACAAGGCCCTTTCGTCTTCAC
Reverse	LuxR	BsGI	AAGCAGTGTACAAGCGATACAATAGTGTGACAA
Forward	mCherry	XmaI	CTCCATATGCTCGTTCCCGGGC
Reverse	mCherry	PstI	CGCTGTCTGCAGCTGCGTTATCCCCTGATTCTGTGGATA
Forward	YFP	EcoRV	ATAGGGAGGCGCCGATGCGTAAAGGGAG
Reverse	YFP	KasI	GCCATAGATATCTTATTATTTGTATAGTTCATCC

C Algorithms

C.1 Clustering algorithms

C.1.1 Deterministic case

Algorithm 8 Clustering the steady state deterministic simulation results

```
1:  for each data point do
2:      if first point then
3:          Make first cluster
4:          cluster counter = 1
5:      else
6:          for each cluster do
7:              if cluster within cluster means ± delta then
8:                  Add to existing cluster
9:                  Update means of clusters
10:             end if
11:             if reached_end and not assigned to cluster then
12:                 cluster counter += 1
13:                 Add new cluster
14:             end if
15:         end for
16:     end if
17: end for
```

C.1.2 Stochastic case

Gap statistic

Algorithm 9 Choosing the optimal number of clusters

function Wk(clusters, cluster_centres)
    for each cluster **do**
        for each point in cluster **do**
            a = matrix norm (cluster_centre − point)
        end for
        $dk = \sum((a)^2) \times (2\times$ number of points in cluster)
    end for
    $wk = \frac{\sum(dk)}{2\times(numberofpointsincluster)}$
    return wk
end function

function Gap_statistic(data, cutoff)
    ks = [1,2,3,4]
    for k in ks **do**
        cluster_centres, clusters = Kmeans(data, k, cutoff)
        $Wk = \log($Wk(clusters, cluster_centres)$)$
        Create references datasets
        for each references dataset **do**
            cluster_centres, clusters = Kmeans(data, k, cutoff)
            $BWk = \log($Wk(clusters, cluster_centres)$)$
        end for
        $Wkb = \frac{\sum(BWk)}{10}$
        $sk = \sqrt{\sum(\frac{(BWk-Wkb)^2}{10})}$
    end for
    $sk = sk \times \sqrt{1+\frac{1}{B}}$
    return ks, Wk, Wkb, sk, data_centres, clusters
end function

function Distance(data, cutoff)
    ks, logWks, logWkbs, sk, clusters_means, clusts = gap_statistic(data, cutoff)
    gaps = logWks − logWkbs
    optimum number of clusters = $gaps[i] \geq (gaps[i+1] - sk[i+1])$
    return cluster_counter, clusters_means
end function

C.2 K-means clustering

Algorithm 10 Clustering stochastic case

```
1:  function KMEANS CLUSTERING(data, k, cutoff)

2:      function UPDATE_CENTRES(old_centres, values)
3:          centre_coords = mean for each dimension
4:          shift = GETDISTANCE(centre_coords, old_centres)
5:          return shift, centre_coords
6:      end function

7:      function GETDISTANCE(a, b)
8:          dist = sqrt((a[x] − b[x])^2 + (a[y] − b[y])^2)
9:          return dist
10:     end function

11:     while True do
12:         for each point in data do
13:             for each cluster do
14:                 dist =GETDISTANCE(point, cluster centre)
15:             end for
16:             Find cluster with minimum distance
17:             Repopulate clusters
18:         end for
19:         biggest_shift ← 0
20:         for as many times as there are clusters do
21:             shift, cluster centres = UPDATE_CENTRES(old_centres, clusters)
22:             biggest_shift = max between shift, biggest_shift
23:         end for
24:         if biggest_shift ≤ cutoff then
25:             break
26:         end if
27:     end while
28:     return cluster_centres, clusters
29: end function
```

C.3 Two sample Kolmogorov-Smirnov test in 2D

The following code is adapted from (Major 2016).

```
from __future__ import division
```

```
import numpy as np
from numpy import random
from scipy.spatial.distance import pdist, cdist
from scipy.stats import kstwobign, pearsonr
from scipy.stats import genextreme

__all__ = ['ks2d2s', 'estat']

def ks2d2s(x1, y1, x2, y2, nboot=None, extra=False):


    '''Two-dimensional Kolmogorov-Smirnov test on two samples.
    Parameters
    ----------
    x1, y1 : ndarray, shape (n1, )
        Data of sample 1.
    x2, y2 : ndarray, shape (n2, )
        Data of sample 2. Size of two samples can be different.
    extra: bool, optional
        If True, KS statistic is also returned. Default is False.
    Returns
    -------
    D : float, optional
        KS statistic. Returned if keyword `extra` is True.
        """


    assert (len(x1) == len(y1)) and (len(x2) == len(y2))
    n1, n2 = len(x1), len(x2)
    sqen = np.sqrt(n1*n2/(n1+n2))

    D1 = maxdist(x1, y1, x2, y2)
    D2 = maxdist(x2, y2, x1, y1)
    D = (D1 + D2)/2

    r1 = pearsonr(x1, y1)[0]
    r2 = pearsonr(x2, y2)[0]
```

```
    if np.isfinite(r2) == False:
        r2 = 0
    r = np.sqrt(1 - (r1**2 + r2**2)/2)

    if nboot is None:
        D = D * sqen/(1 + r*(0.25 - 0.75/sqen))
        #p = kstwobign.sf(D)
        p = 0
    else:
        raise NotImplementedError
    if extra:
        return p, D
    else:
        return p


def maxdist(x1, y1, x2, y2):
    n1 = len(x1)
    D1 = np.zeros((n1, 4))
    for i in range(n1):
        a1, b1, c1, d1 = quadct(x1[i], y1[i], x1, y1)
        a2, b2, c2, d2 = quadct(x1[i], y1[i], x2, y2)
        D1[i] = [a1-a2, b1-b2, c1-c2, d1-d2]
    D1[:,0] -= 1/n1          # re-assign the point to maximize difference,
    #D1[D1 >= 0] += 1/n1      # discrepancy is significant for N < ~50
    #D1 = np.abs(D1).max()
    ix = np.unravel_index(np.argmax(np.abs(D1)), D1.shape)
    D1 = D1[ix]
    if D1 >=0:
        D1 += 1/n1
    return D1


def quadct(x, y, xx, yy):
    n = len(xx)
```

```
    ix1, ix2 = xx <= x, yy <= y
    a = np.sum( ix1 &  ix2)/n
    b = np.sum( ix1 & ~ix2)/n
    c = np.sum(~ix1 &  ix2)/n
    d = 1 - a - b -c
    return a, b, c, d
```