# 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 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

# Contents

# List of Figures

# List of Tables

# Abbreviations

**ABC** Approximate Bayesian Computation.

**ATc** anhydrotetracycline.

**CS-LU** Lu classic switch.

**CS-MA** Mass action classic switch.

**DP-LU** Lu double positive switch.

**DP-MA** Mass action double positive switch.

**GPU** Graphical Processing Unit.

**MCMC** Markov Chain Monte Carlo.

**ODE** Ordinary Differential Equation.

**SMC** Sequential Monte Carlo.

**SP-LU** Lu single positive switch.

# Acknowledgements

# 1 Introduction

## 1.1 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:2009ez**; **Andrianantoandro:2006bi**). 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:2006ht**). A device processes inputs, performs functions and produces outputs (**Andrianantoandro:2006bi**). 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:2009bt**; **Heinemann:2006ht**). A hierarchy similar to computer science is used, with cells, pathways and biochemical reactions acting as computers, modules and gates respectively (**Andrianantoandro:2006bi**).

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:2009bt**). 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:2006bi**).

## 1.2 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) (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) (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:2007jl (Batt:2007jl)** 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) (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** (**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.

## 1.3 Introduction to Biochemical Modelling

### 1.3.1 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**

### 1.3.2 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 2006b). 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 2006b). 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 equations (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 2006b. 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 2006b). The algorithm, developed by Gillespie (1977) (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

### 1.3.3 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**).

## 1.4 Introduction to Bayesian statistics

# 2 Bayesian model selection

# 3 StabilityFinder

## 3.1 Background

Synthetic biology is now entering an age where simple synthetic circuits have been built, such as toggle switches (**Kramer:2004kq**; **Ham:2008hh**; **Deans:2007cya**; **Friedland:2009ce**; Gardner, Cantor, & Collins 2000; Isaacs et al. 2003), oscillators (**Stricker:2008jqa**; **Fung:2005jd**; **Tigges:2009jx**) and pulse generators (**Basu:2004gn**), 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:2009ez**), 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).

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:2009ez**), 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. 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.

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**).

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:2004kq**). Other synthetic applications of the toggle switch include the construction of a synthetic genetic clock (**Atkinson:2003tu**), of a predictable genetic timer (**Ellis:2009hka**), 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:2009ez**). For a review on current circuits see (**Khalil:2010hm**) and for possible future applications see (**Lu:2009ez**). This leap will be difficult to achieve before first being able to build robust and well characterised individual switches.

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 (de Jong 2002; Wilkinson 2006a). 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).

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 et al. 2007), 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:2007jo**; **Banaji:2010fh**; **Feliu:2013dz**). For example, **Feliu:2013dz** developed an approach using chemical reaction theory and generalised mass action modelling (**Feliu:2013dz**). No approach exists that can handle both deterministic and stochastic systems in an integrated manner.

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 (**Walczak:2005ds**; Gardner, Cantor, & Collins 2000; 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:2015vya** identified multiplicative noise as the source of bistability in the stochastic case (**Biancalani:2015vya**). 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 3.1. 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.

Here we present 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. Our framework can be used for comparing the conclusions drawn by various modelling approaches and thus provides a way to investigate appropriate abstractions. We have made this framework into a python package, called StabilityFinder. This methodology is used to investigate genetic toggle switches and uncover the design principles behind making a bistable switch, as well as those necessary to make a tristable switch. We find that degradation rates of transcription factors are important for bistability, and that the addition of positive autoregulation can create tristable behaviour and also significantly more robust bistability when feedback strength is well balanced. Modelling transcription and translation allows us to conclude that transcriptional bursting can inhibit bistability, and also that bistability can occur even when the assumptions of time scale separation in the repressor dynamics are not met. We also examine the design principles behind the design of bistable versus tristable switches and highlight the importance of including stochastic dynamics when modelling these systems. Finally we demonstrate the ability of the framework to examine more complex systems and examine the design principles of a three gene switch. These examples demonstrate that StabilityFinder will be a valuable tool in the future design and construction of novel gene networks.

**Table 3.1** Summary of stability for the toggle switch found via different modelling approaches

| | Simple | | | Double positive autoregulation | | |
|---|---|---|---|---|---|---|
| | *Stability* | *Reference* | *Notes* | *Stability* | *Reference* | *Notes* |
| **Deterministic** | Monostable | (Loinger et al. 2007) | no cooperativity, exclusive & general | Bistable | (Guantes & Poyatos 2008) | |
| | Bistable | (Gardner, Cantor, & Collins 2000) | copperativity >2, | Tristable | (Guantes & Poyatos 2008) | |
| | | (Loinger et al. 2007) | bound repressor degradation | 4 steady steates | (Guantes & Poyatos 2008) | |
| **Stochastic** | Monostable | (Loinger et al. 2007) | no cooperativity, weak repression | Tristable | (Lu, Onuchic, & Ben-Jacob 2014) | |
| | | (Lu, Onuchic, & Ben-Jacob 2014) | | | | |
| | Bistable | (**Biancalani:2015vya**) | exclusive, controlled by noise strength | | | |
| | | (Lipshtat et al. 2006) | no cooperativity | | | |
| | | (Loinger et al. 2007) | no cooperativity, exclusive & bound repression degradation | | | |
| | Tristable | (Loinger et al. 2007) | no cooperativity, strong repression | | | |
| | | (Ma et al. 2012) | | | | |

## 3.2 Methods

### 3.2.1 StabilityFinder

The framework is based on a statistical inference method which combines Approximate Bayesian Computation (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, $\theta$, 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 thorough 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, $\epsilon$, the sample is rejected, otherwise it is accepted.

**Algorithm 1** ABC rejection algorithm

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

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) (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, $\epsilon$ for the distance of the simulated data to the desired data is made

smaller at each iteration. When $\epsilon$ is sufficiently small, the result will approximate the posterior distribution (Toni et al. 2009).

To investigate the multistable behaviour of systems, a number of extensions to existing approaches are required. For a given set of parameter values, sample points are taken across initial conditions using latin hypercube sampling (XXX), and the ensemble system simulated in time until steady state. The distance function in ABC is replaced by a distance on the desired stability of the simulated model. To do this we cluster the steady state coordinates using k means clustering and use the gap statistic to determine the number of clusters (XXX). The algorithm is summarised below.

**Algorithm 2** StabilityFinder algorithm

1: Initialise $\epsilon$
2: population p $\leftarrow$ 1
3: **if** p = 1 **then**
4: Sample particles ($\theta$ ) 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: Sample initial conditions via Latin Hypercube Sampling.
10: Simulate each particle to obtain steady state values.
11: Cluster steady state
12: Reject particles if d $>$ $\epsilon$.
13: Calculate weight for each accepted $\theta$
14: Normalise weights
15: **repeat**steps 3 - 15
16: **until** $\epsilon \leq \epsilon_T$

This algorithm is available as a Python package, called StabilityFinder. The user provides an SBML model file (XXX) and an input file that contains all the necessary information to run the algorithm, including the desired stability and the final tolerance, $\epsilon$, for the distance from the desired behaviour necessary for the algorithm to terminate. The flow of execution is illustrated in Figure 3.1. Since the algorithm is computationally intensive, all deterministic and stochastic simulations are performed using algorithms implemented on Graphical Processing Unit (GPU)s (XXX).

### 3.2.2 Calculating robustness

We use the results from StabilityFinder to estimate system robustness by comparing the volume of the posterior to the volume of the prior within a given model. This is done using a Monte Carlo sampling rejection algorithm by taking random samples from the prior and accepting those found in the functional region $F$ (or posterior distribution). Robustness is then defined as the number of accepted samples divided by the total number of samples.

**Algorithm 3** Calculating robustness via Monte Carlo sampling rejection

1: Sample from priors
2: Get min and max boundaries of functional region of posterior ($F$)
3: **if** sample within $F$ **then**
4: $\quad accepted+ = 1$
5: **end if**
6: $acceptance\_rate = \frac{accepted}{number of samples}$
7: **return** $acceptance\_rate$

## 3.3 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 3.2A, and in the deterministic case is defined by the following ODEs.

$$\frac{du}{dt} = \frac{a_1}{1 + v^{\beta}} - u$$
$$\frac{dv}{dt} = \frac{a_2}{1 + u^{\gamma}} - v,$$

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, $\beta$ is the cooperativity of repression of promoter 1 and $\gamma$ of repressor 2. Gardner *et al* 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 $\beta$, $\gamma$ >1 (Gardner, Cantor, & Collins 2000). In order to test StabilityFinder we used it to find the posterior distribution for which this model exhibits bistable behaviour. We therefore set the desired behaviour to two (stable) steady states, and using a wide range of values as priors as shown in the Supplementary Information, we used StabilityFinder to

find the parameter values necessary for bistability to occur. The posterior distribution calculated by StabilityFinder for the Gardner deterministic case is shown in Figure 3.2B.

These results agree with the results reported by Gardner, Cantor, & Collins (2000) (Gardner, Cantor, & Collins 2000). For this switch to be bistable $a_1$ and $a_2$ must be balanced while $\beta$ and $\gamma$ must both be >1, as can be seen in the marginal distributions of $\beta$ and $\gamma$ in Figure 3.2B. The conditions set by the original paper for parameters $a_1$ and $a_2$ are met, as the joint distribution shown in Figure 3.2B matches the bifurcation lines calculated in the original paper How do you know, did you calculate these? I think you should try and plot them on the posterior. We next applied StabilityFinder 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 is shown in Figure 3.2C. 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), which is expected due to the nature of clustering deterministic steady states versus stochastic steady states. 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. The study on the effect of the clustering methods on robustness is shown in the Supplementary Information. These results demonstrate that StabilityFinder can be used to find the parameter values that can produce a desired stability and allow us to confidently apply the methodology to more complex models.

**Table 3.2** Gardner switch priors in the deterministic and stochastic cases

| Parameters | | | | Species | |
|---|---|---|---|---|---|
| $a_1$ | $\beta$ | $a_2$ | $\gamma$ | $s_1$ | $s_2$ |
| 0-60 | 0-5 | 0-60 | 0-5 | 0-100 | 0-100 |

## 3.4 Lu toggle switch models

We next analyzed an extension of the Gardner switch model developed by Lu, Onuchic, & Ben-Jacob (2014) (Lu, Onuchic, & Ben-Jacob 2014). They considered two types of switches, the classic switch consisting of two mutually repressing transcription

factors (model CS-LU), as well as a Lu double positive switch (DP-LU). The classical switch was found to be bistable given the set of parameters used, while the DP switch was found to be tristable (Lu, Onuchic, & Ben-Jacob 2014). The classical model used in their study is given by the following system of ODEs.

$$\dot{x} = g_x H^S_{xy}(y) - k_x x$$
$$\dot{y} = g_y H^S_{yx}(x) - k_y y,$$

where

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

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

For the parameter values used in the Lu study, the classical switch exhibits three steady states (Figure 3.3), two of which are stable and one is unstable. Using StabilityFinder with priors centred around the parameter values used in the original paper (see Supplementary Information), we can identify the most important parameters for achieving bistability. The posterior distribution of these models are shown in Figure 3.3A. 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 (**XXX**). Anything else regarding this classical model? with more parameters

It is known that that the addition of positive autoregulation to the classical toggle switch can induce tristability (**XXX**; Lu, Onuchic, & Ben-Jacob 2014). Here we investigate the interplay of positive autoregulation on the values of the other parameters in the model. We 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 advantage of using StabilityFinder 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.

The DP-LU model is given by

$$\dot{x} = f_x(x,y) = g_x H^S_{xy}(y) H^S_{xx}(x) - k_x x$$
$$\dot{y} = f_y(x,y) = g_y H^S_{yx}(x) H^S_{yy}(y) - k_y y,$$

whereas the CS-LU switch is modelled using the following ODE system

$$\dot{x} = g_x H^S_{xy}(y) H^S_{xx}(x) - k_x x$$
$$\dot{y} = g_y H^S_{yx}(x) - k_y y.$$

We find that the switch with single positive autoregulation is capable of bistable behaviour as seen in Figure 3.3B, but this is only possible when the strength of the promoter under positive autoregulation, $gx$, is small. 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 Can we say a bit more here?. 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. 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 ($\text{III}_H$) and one in which it is low ($\text{III}_L$) (Guantes & Poyatos 2008).

deterministic dynamics (Guantes & Poyatos 2008). The classical switch is also capable of both bistable and tristable behaviour when stochastic dynamics capture small number effects (Ma et al. 2012). It is therefore of great interest to understand the conditions under which these two behaviours occur in both stochastic and deterministic scenarios.

In order to do this using StabilityFinder, we first obtained posterior distributions for bistable and tristable behaviours in the deterministic case (DP-LU model) and then compared the individual parameter distributions (Figure 3.6 and Supplementary Information). From analysis of the one dimensional marginal distributions it appears that there is no difference in the parameter values that allow a switch to be bistable versus tristable (Figure 3.6B). However, upon examination of the two dimensional marginal distributions we find that the correlation structure of a small subset of the posterior parameter values is in fact different under the two behaviours (Figure 3.6C). Most notably, we find differences in the bivariate distribution of the two parameters for gene expression, $gx$ and $gy$, as highlighted in Figure 3.6C, box 1. In the tristable case the distribution is more constrained than in the bistable case, as

both parameters must be small for tristability to arise. Parameters $xYY$ and $gy$ are tightly constrained in the tristable case and both required to be small, but less so in the bistable case (Figure 3.6C, box 3). Another notable difference is between parameters $xXX$ and $lXX$ shown in Figure 3.6C, box 4, where they are constrained in the bistable case but not the tristable case. Interestingly, we also find parameter correlations conserved between the bistable and tristable case, as seen in Figure 3.6C, box 2, 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.

To further demonstrate the flexibility of our framework we investigated a system capable of higher stabilities. Multistability is found in differentiating pathways, like the myeloid differentiation pathway (**Ghaffarizadeh:2014bt**; **Cinquin:2005go**). We allow for these more complex dynamics by extending the Lu DP model by adding another gene, making it a three gene switch. This new system is depicted in Figure 3.8A. In StabilityFinder we look for six steady states, the output being in nodes $X$ and $Y$ and using the priors shown in Table 3.4. We successfully find that the system is capable of six steady states, as shown in Figure 3.8C and as predicted mathematically and shown in Figure 3.8B. Interestingly, and consistent with the results presented above, we find that the most constrained parameters for this behaviour are again the degradation rate of the proteins, $kx$. Additionally we find that the Hill coefficients for the repressors, $nxy$, are tightly contrained to be smaller than 1.5 as seen in Figure 3.8D. What else can we do here? Can we simulate the system switching or something? This example demonstrates that StabilityFinder 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.

## 3.5 Mass Action switches

Although the DP switch can achieve tristability, we investigated how the addition of positive autoregulation alters the robustness of the switch for bistable behaviour. Here we define a robust system as a device that can withstand fluctuations in parameter values and still produce the desired behaviour (parametric robustness). Feed-

**Table 3.3** Priors of the classical(CS-LU), single positive (SP-LU) and double positive (DP-LU) models.

| Parameter | Symbol | CS-LU | SP-LU | DP-LU |
|---|---|---|---|---|
| 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 |

**Table 3.4** Priors used in the three-node switch

| Parameter | Symbol | Range |
|---|---|---|
| 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 |

back loops are well known key regulatory motifs (Brandman et al. 2005). Although negative feedback loops are essential for homeostasis and buffering (Thomas, Thieffry, & Kaufman 1995) and can increase robustness to extrinsic noise sources, we focussed on the addition of positive feedback because this has been shown to increase robustness in other systems (**XXX**). We use StabilityFinder to compare the two models, and determine whether its worthwhile building a bistable circuit with added positive feedback.

In order to study the switch system in the most realistic way, we avoid using 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 Lu, Onuchic, & Ben-Jacob (2014) (Lu, Onuchic, & Ben-Jacob 2014) into a system of 18 equations and 7 parameters in the classical switch case. These are shown in the Supplementary Information.

Given the posterior distributions shown in Figure 3.4, we find that the parameter for transcription factor degradation (*deg*) has to be high, a result that reflects the findings for the Lu models above Do we really see that??. We also find that the addition of positive feedback loops greatly increases the system's robustness to parameter fluctuations as seen in Figure 3.4D. Adding positive feedback loops to the model allows it to be bistable over a greater range of parameter values, but only when the system is symmetric. When this constraint is removed we no longer see a difference in robustness between the CS-MA and DP-MA models. What does this mean?

We find that in the stochastic case, both the simple switch, S-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, we 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 3.7B, differences in the parameter values are observed between the bistable and tristable switches, in both CS-MA and DP-MA models. The design principles for both the CS-MA model and the DP-MA model are summarised in Table 3.5

**Table 3.5** 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 | - |

## 3.6 Conclusions

We have developed an algorithm that can identify the parameter regions necessary for a model to achieve a given number of stable steady states. The novelty in our framework over existing methodology is that complex models can be analyzed assuming both deterministic and stochastic dynamics. We have shown that the algorithm can be used to infer parameter ranges and compare robustness between models. We found that adding positive feedback increases the robustness of a toggle switch model to parameter fluctuations. We also found that stochastic modelling can enable a model to achieve stabilities not possible in the deterministic case. We uncovered the design principles that make a tristable switch and finally we found that a three-node switch is capable of hexastability.

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 StabilityFinder.

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. We also limited our 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.

More generally our results highlight that changing the level of abstraction included in the model can significantly alter the observed multistationarity. This is further exacerbated by a change in the dynamics from deterministic to stochastic. These results for the case of multistationarity, assuming they translate to systemic behaviour more generally, suggest that a programme of experimental work, com-

bined with systems modelling, is required to understand the rules of thumb for abstraction in model based design of synthetic biological systems.

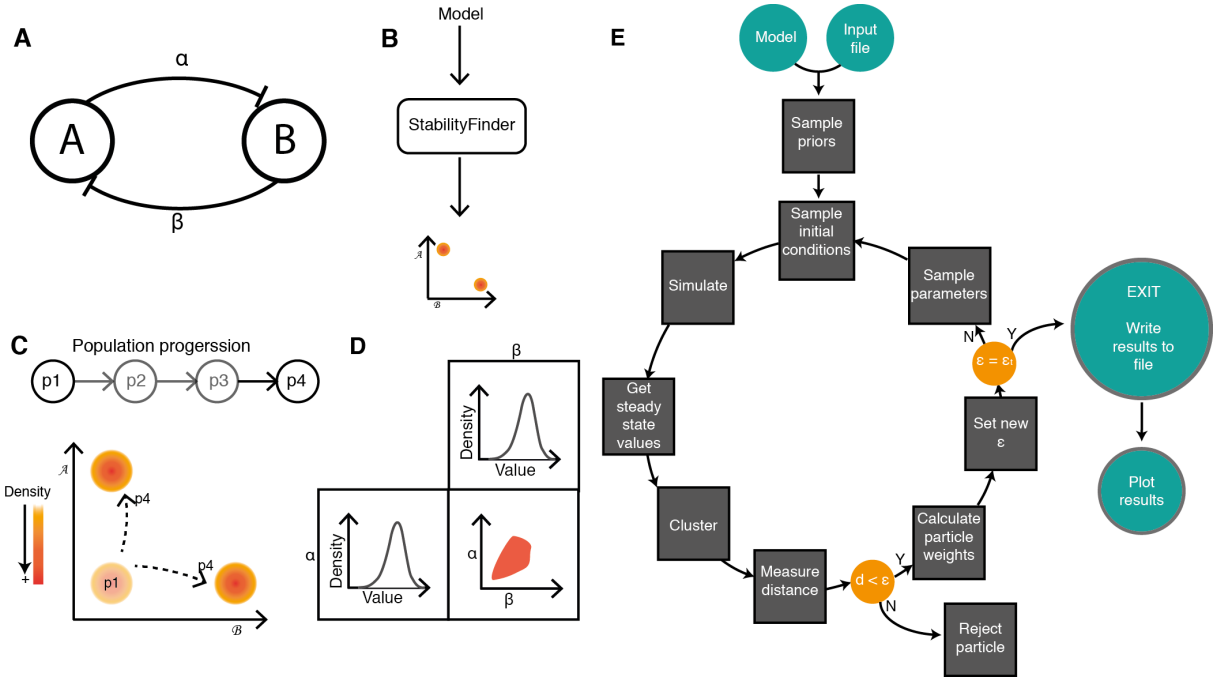


**Figure 3.1** : Using sequential Monte Carlo to examine system stability. The algorithm takes as input a model (A) and evolves it to the stability of choice (C) via intermediate populations. In this example model shown in A, There are two species and two parameters. For the model to be bistable, the 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 StabilityFinder. The overview of the algorithm is shown in (E).

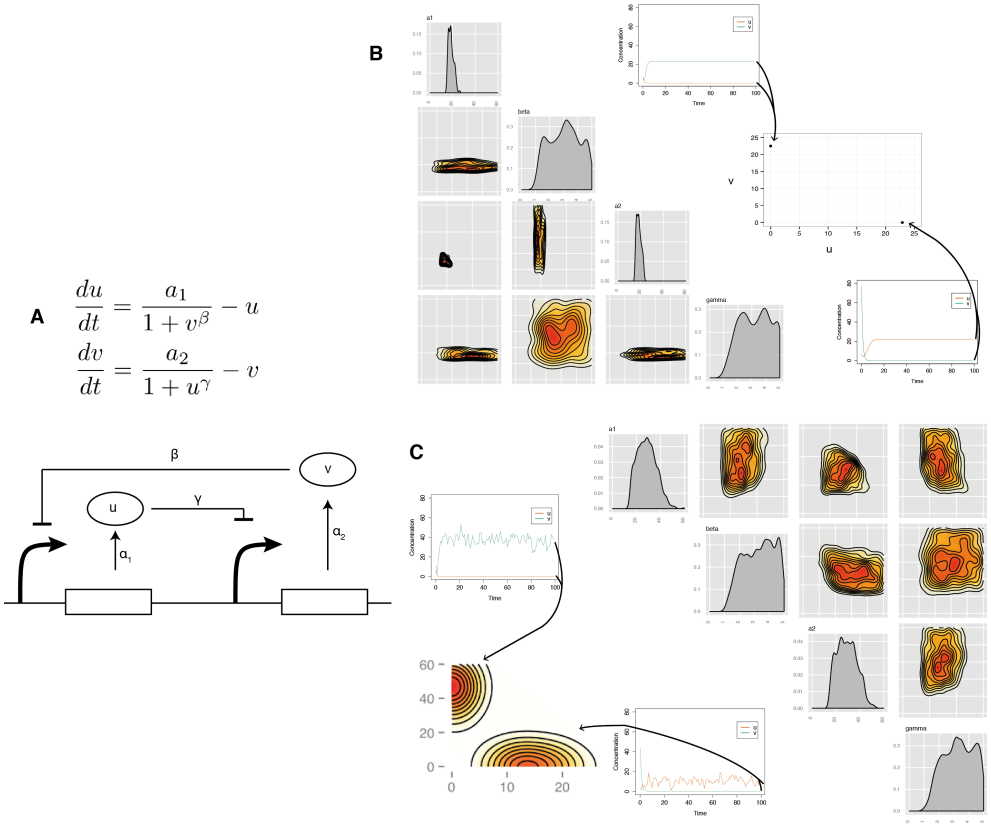


**Figure 3.2** : Elucidating the stability of the Gardner switch. The Gardner model (A) consists of two mutually repressing transcription factors. It can be reduced to a two-equation system, where $u$ and $v$ are the two transcription factors, $a_1$,$a_2$ are their effective rates of synthesis, $u$,$v$ are their concentrations and $\beta$, $\gamma$ represent the cooperativity of each promoter. There are four parameters in the model, for which we want to find the values for which this system is bistable. We use StabilityFinder to find the posterior distribution of the bistable Gardner switch, deterministically (B) and stochastically (C). The posterior distributions are shown as the density plots of each parameter as well as each one plotted against the other.

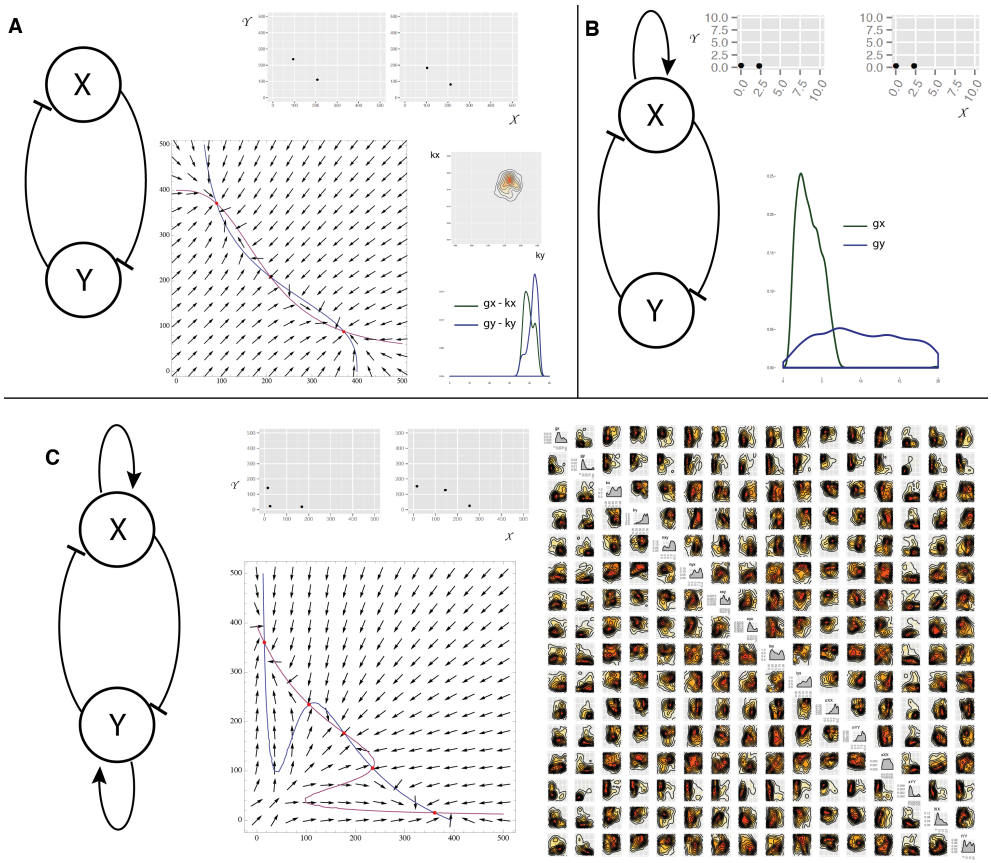


**Figure 3.3** : The three variants of the Lu models. (A)**CS (deterministic)**. The classic switch with no autoregulation is bistable. The most restricted parameters for this behaviour are $kx$ and $ky$ which both have to be high relative to the prior while the net protein production for $X$ and $Y$ myst be balanced. (B)**SP (deterministic)**.The extended Lu model with a single positive autoregulation on $X$. This model is bistable when $gx$ is small. (C) **DP (deterministic)**. The Lu model with double positive autoregulaiton is tristable, and its posterior distribution shown here. 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).

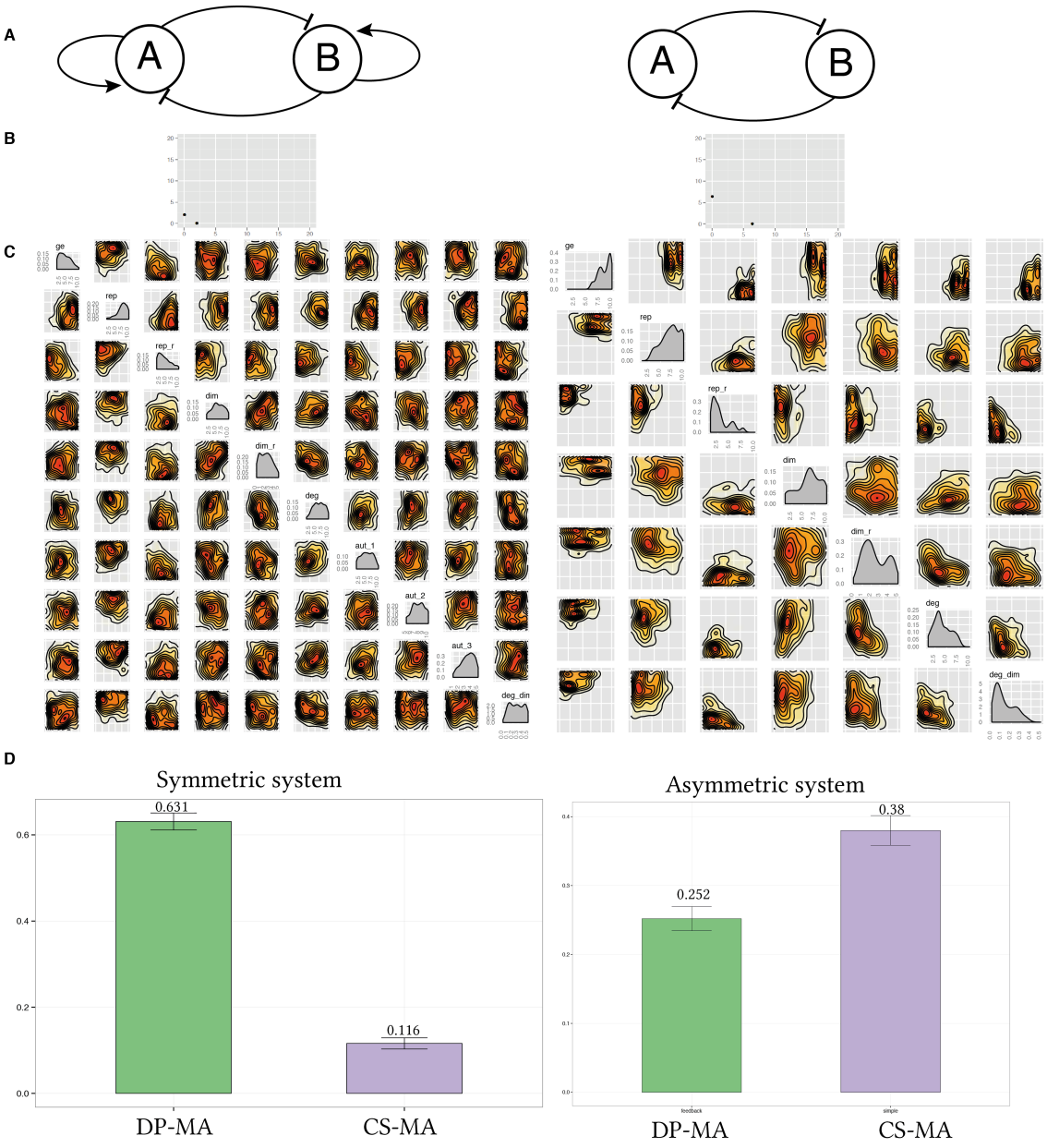


**Figure 3.4** : Adding positive feedback increases robustness to parameter fluctuations. We compare the posterior distributions of the simple mass action switch and the switch with added double positive autoregulation (A). The two models were both capable of bistable behaviour (B). The two posterior distributions are shown in (C). Comparing the functional region $F$ of the posterior distributions (D) we find that there is a five-fold increase in robustness when positive autoregulation is added. This is not the case for the asymmetric switch in which there is no significant difference in robustness between the CS-MA and DP-MA models.

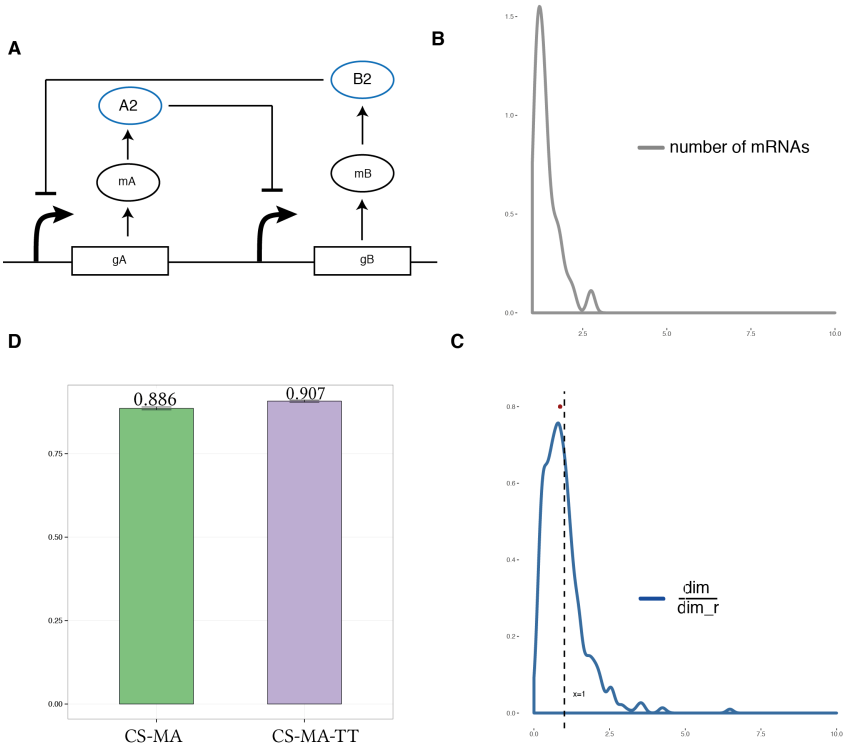


**Figure 3.5** : (A) Separating transcription and translation in the mass action model. (B) Transcriptional bursting does not allow for a bistable switch. (C) The Quassi Steady state approximation assumption that the dimerization reaction (*dim*) is much faster than its reverse (*dim_r*) cannot be justified 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 $\frac{dim}{dim\_r} < 1$ (D) No difference was found in the robustness to parameter fluctuations for the two models tested here.

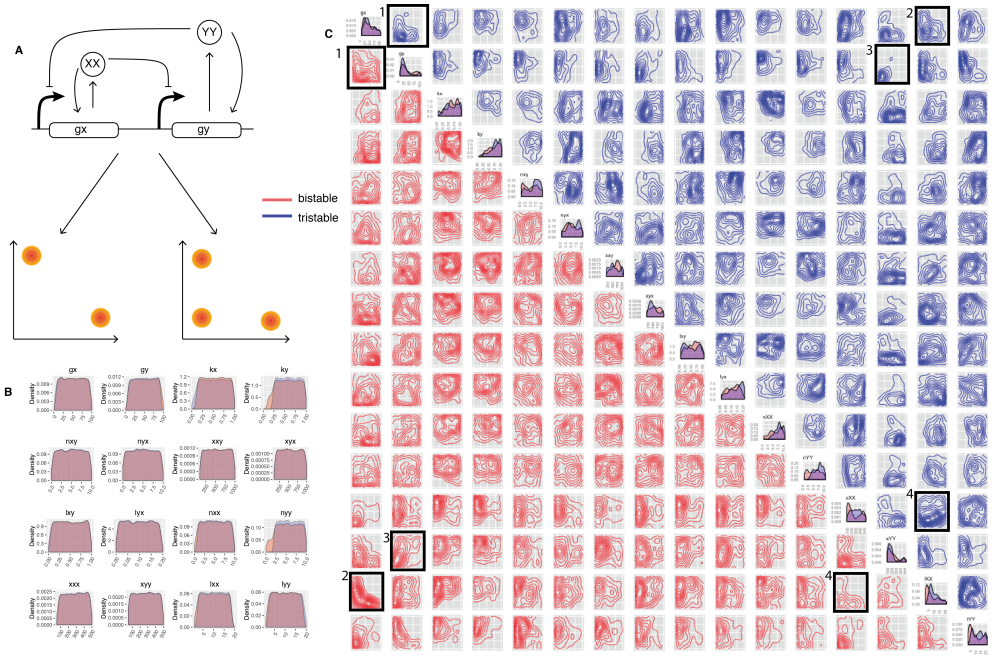


**Figure 3.6** : Design principles of a tristable switch. (A) Using the Lu model with added positive autoregulation we uncover the design principles dictating if a switch will be bistable or tristable. (B) By considering each parameter separately we cannot find a significant difference in the parameter values acceptable for a bistable versus a tristable switch. (C) 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. The posterior distribution of the bistable switch is shown in red and of the tristable switch in blue. The bivariate distributions for which a difference is observed between a bistable and a tristable switch are in black boxes.

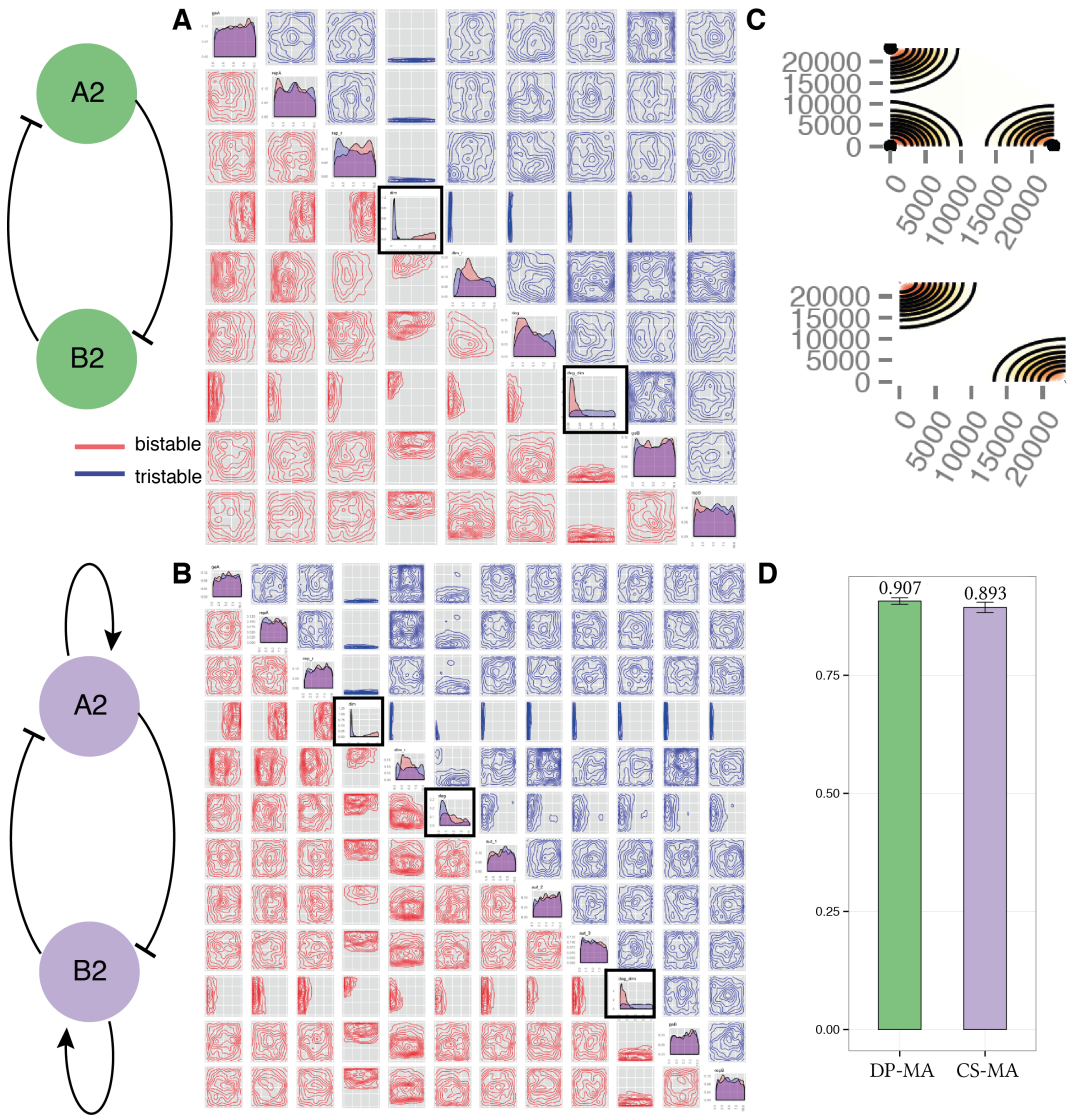


**Figure 3.7** : Tristability is possible in the mass action toggle switch models only when simulated stochastically. (A) the simple toggle switch with no autoregulation can be both bistable and tristable. The two posteriors are shown, where the posterior distribution of the bistable switch is shown 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). A sample phase plot of a stochastic tristable and bistable mass action switch is shown in (C). Comparing the robustness of the switch with and without positive autoregulation, we don't find a significant difference (D).

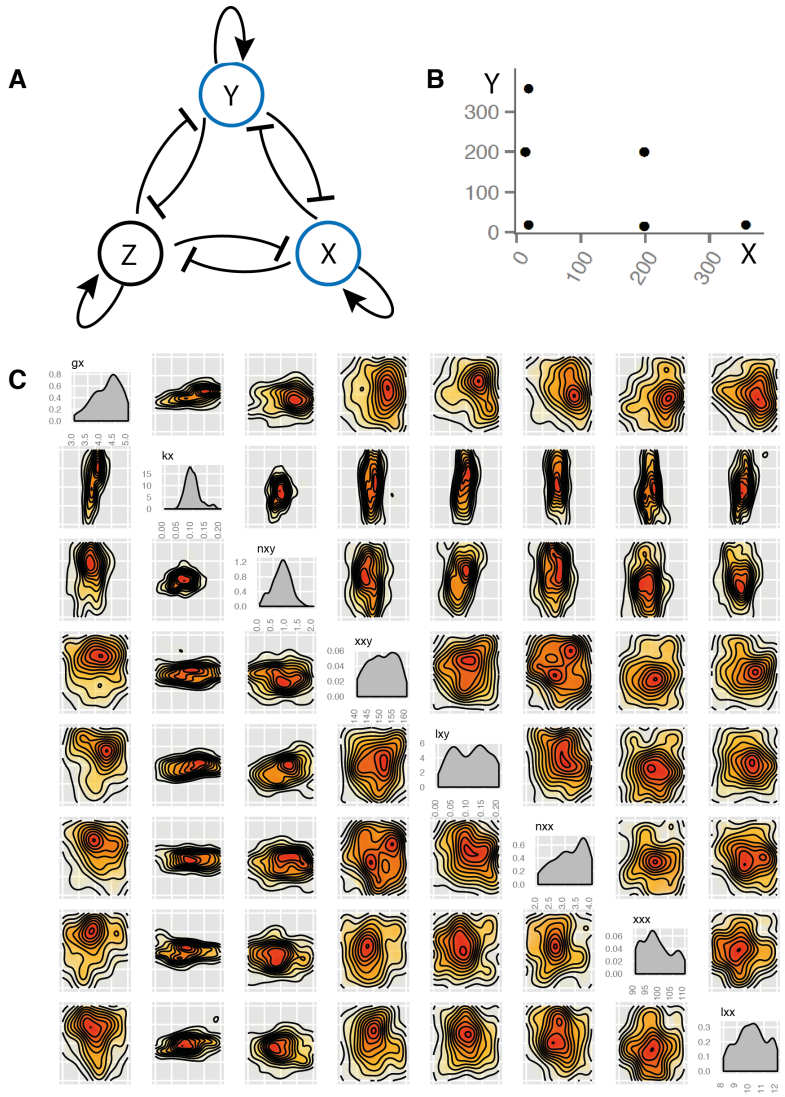


**Figure 3.8** : The three-node mutual repression model, with added positive auto-regulation on each node. (A) The model. The model is studied in two dimensions using StabilityFinder, $X$ and $Y$. (B) The phase plot of the resulting final population of StabilityFinder. There are 6 steady states. (C) The posterior distribution of the 6-steady state three-node system. Parameters $kx$ and $nxy$ are the most constrained.

# 4 Characterising the genetic toggle switch

# 5 ABC-Flow

## 5.1 Introduction

## 5.2 Methods

**Algorithm 4** ABC-Flow

1: Read input file
2: **if** ABC-Rejection **then**
3: Sample from priors
4: Simulate model
5: Convert signal to intensity
6: Measure distance to data
7: Reject particles if d $> \epsilon$.
8: **if** number of accepted particles == number of particles **then**
9: Exit
10: **else**
11: Return to step 3.
12: **end if**
13: **end if**

14: **if** ABC-SMC **then**
15: Initialise $\epsilon$
16: population p $\leftarrow 1$
17: **if** p $= 1$ **then**
18: Sample particles ($\theta$) from priors
19: **else**
20: Sample particles from previous population
21: Perturb each particle by $\pm$ half the range of the previous population (j) to obtain new perturbed population (i).
22: **end if**
23: Simulate model
24: Convert signal to intensity
25: Measure distance to data
26: Reject particles if d $> \epsilon$.
27: Calculate weight for each accepted $\theta$
28: $$w_t^{(i)} = \begin{cases} 1, & \text{if } p = 0 \\ \frac{\pi(\theta_t^{(i)})}{\sum_{j=1}^{N} w_{t-1}^{(j)} K_t(\theta_{t-1}^{(j)}, \theta_t^{(i)})}, & \text{if } p \geq 0. \end{cases}$$
29: Normalise weights
30: **repeat** steps 17 - 29
31: **until** $\epsilon \leq \epsilon_T$
32: **end if**

## 5.3 Results

### 5.3.1 Distance Calculations

**Algorithm 5** 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: $d = \sum((fD - fS)^2)$

### 5.3.2 Comparing 1D and 2D distances

In order to compare the 1D and 2D fitting to the data in ABC-Flow, we must first find out how comparable the distance measures are. Here we simulate two normal distributions, with identical mu and sigma, and calculate the distance between the two using the distance measure used in ABC-Flow. Doing this 1000 times, we then plot the distribution of epsilon. By doing that we can calculate the variance of the epsilon distribution, and find out the error that can be expected when measuring the distance in ABC-Flow. By doing that in 1D and in 2D we can compare the epsilon variances.

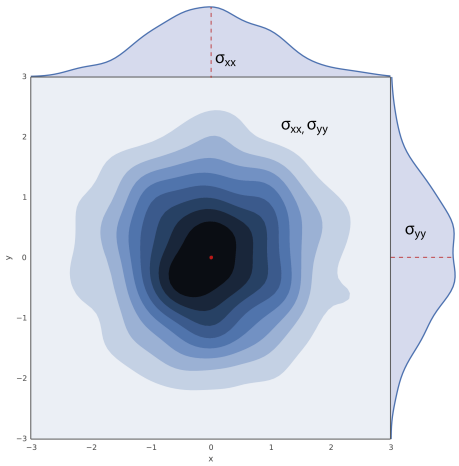


**Figure 5.1** Comparing 1D and 2D distributions.

#### 5.3.2.1 Normal distribution

Here, we simulate two normal distributions, with $\mu = 0$ and $\sigma = 1$ 1000 times and measure the distance. In the 2D case, we simulate two multivariate normal distributions, with $\mu = 0$ and covariance=[0.5 0, 0 0.5]. In the 1D case the median of the epsilon distribution was 0.00196 and the variance 0.0016. In the 2D case the median was 3.41e-14 and the variance 1.97e-07.

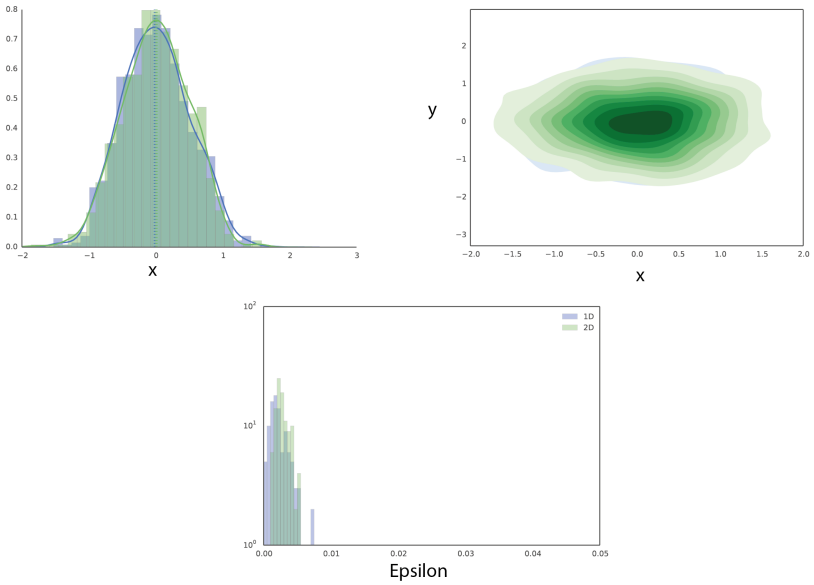


**Figure 5.2** Epsilon distribution for 1D (blue) and 2D (green) distances.

Next we study the effect that the number of data points have on the variance of the epsilons.

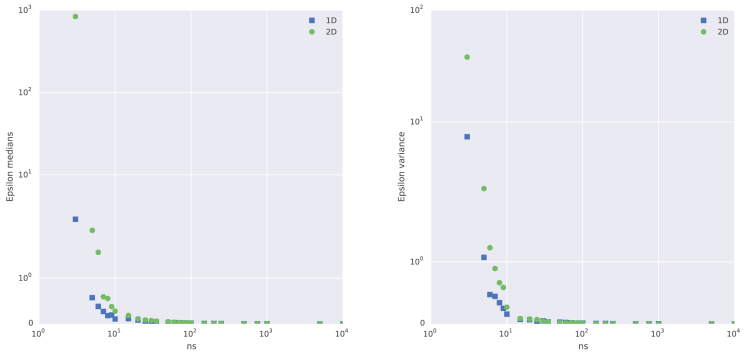


**Figure 5.3** Epsilon distribution medians and variance over number of data points.

As the number of data points increases, the distance calculation becomes more precise and more accurate. The median distance approaches zero, and the variance of the epsilon distribution decreases to zero. From Figure 5.3 we conclude that the minimum number of data points that need to be used to calculate the distance between the distributions in ABC-Flow is 100.

The next parameter to be optimised is the bin size used in the distance calculation. In both 1D and 2D, the space is divided into bins, and the distance between corresponding bins in the data and the simulated data is calculated. The overall distance between to distributions is the sum of the distances between corresponding bins.

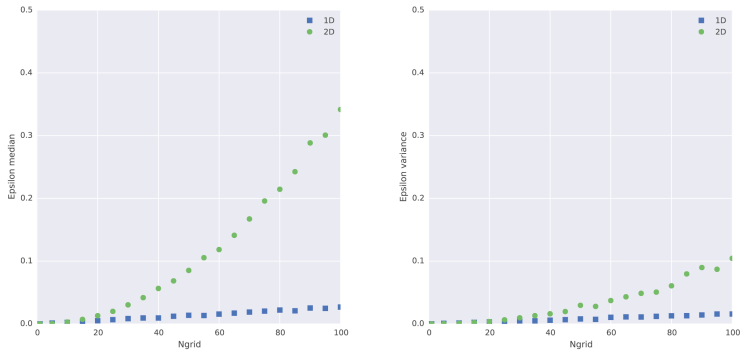


**Figure 5.4** Epsilon distribution medians and variance vary with the size of bins used.

We find that the 2D distance is more sensitive to the bin size than the 1D distance. From Figure 5.4 we conclude that for a sample size of 100 data points, the optimal bin size for 1D and 2D distance calculation is 10. The above optimizations have been made by using standard normal distributions, of $\mu = 0$ and $\sigma = 1$. Here we investigate whether the distance calculation depends on the $\mu$ and *sigma* of the distributions.

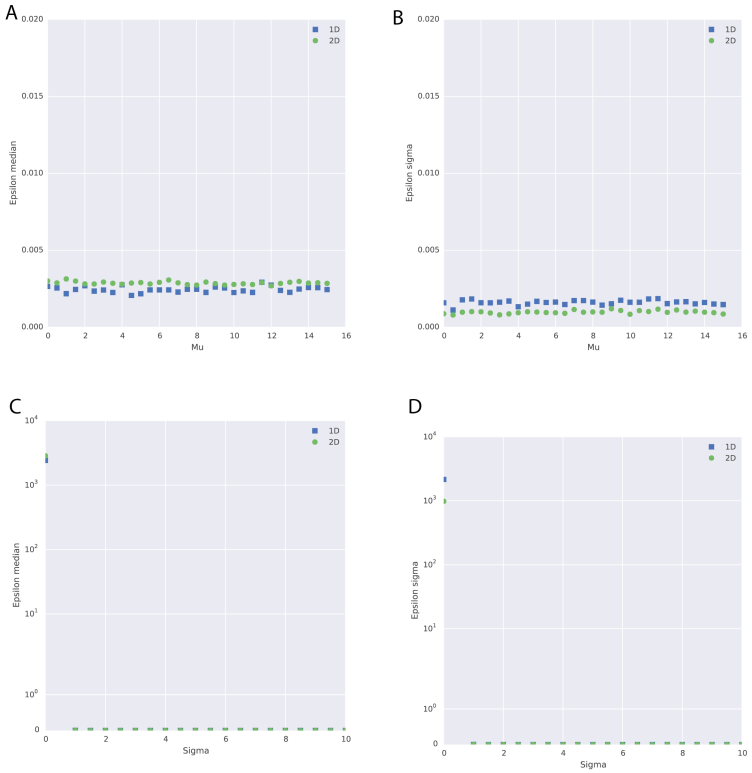


**Figure 5.5** The distance calculations do not depend on the $\mu$ and $\sigma$ of the distributions. The median (A) and variance (B) of the epsilons remains constant with increasing $\mu$ (C,D) as well as with increasing $\sigma$.

Next, by varying the amount of $\mu$ and $\sigma$ by which the two normals differ, we can find out the dynamical range of the epsilons. Whether two distributions are identical or vary by a large amount, we can get an estimate of how much the epsilon will vary from one to the other, both in 1D and 2D.

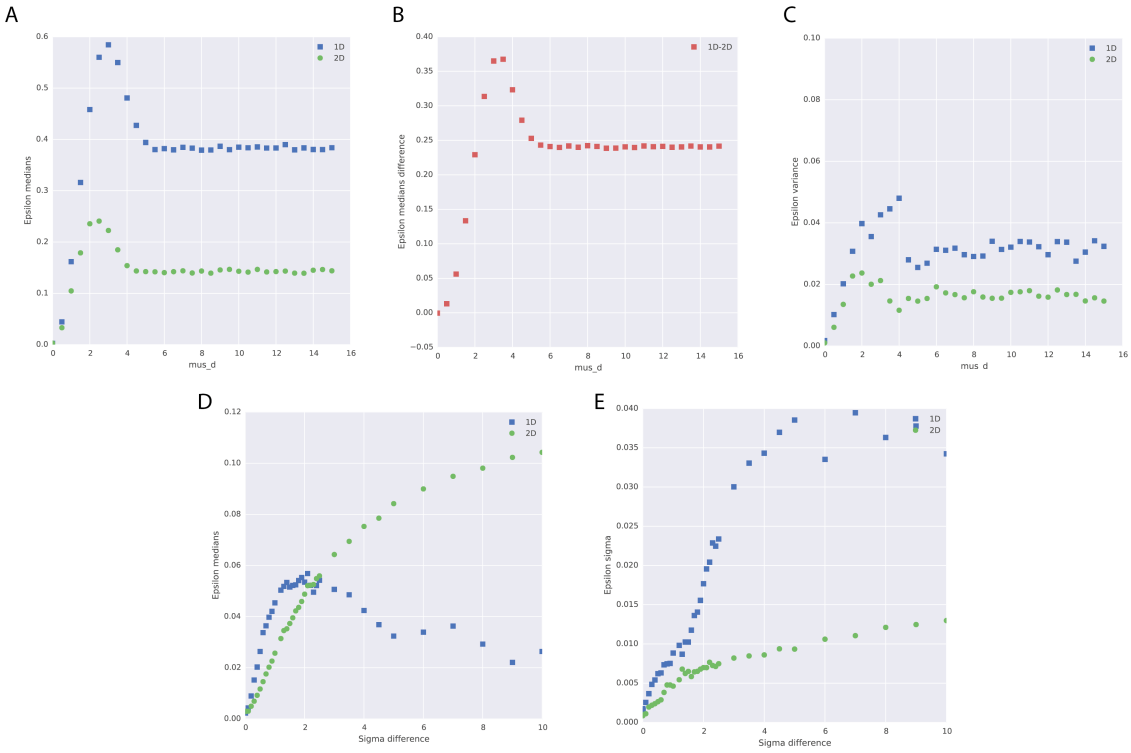


**Figure 5.6** (A)The range by which epsilon varies as the difference between the $\mu$ of the distributions increases. (B) The difference between the epsilon distribution medians in the 1D and 2D case is not constant with increasing $\mu$ difference between the data sets. (C)The variance of the epsilon distributions remains relatively constant with increasing $\mu$. (D)The median of the epsilon distributions varies by a small amount with increasing difference in the $\sigma$ of the distributions, but the variance (E) remains relatively constant.

From Figure 5.6 we find that as the difference in $\mu$ increases the epsilon medians reach a plateau. We find that beyond a difference of 4 in $\mu$, the distance calculation cannot further separate the distances. This can be caused by the fact that when first dividing the space into bins, the range of the data is used to define the grid. If all the simulated data is located outside that grid, the algorithm can no longer distinguish between them, and will only allocate them as outside the range. The variance of the epsilon distributions does not vary significantly with increasing difference in $\mu$. As the difference in $\sigma$ increases between the distributions, we find that the median in the 2D distance calculation increases but not in the 1D. Note, that the range of the difference in the epsilon medians is small and thus we conclude that differences in the $\mu$ of a distribution are much better detected than the differences in the $\sigma$. The variance of the epsilon distribution when $\sigma$ is varied does not change significantly with increasing $\sigma$ difference.

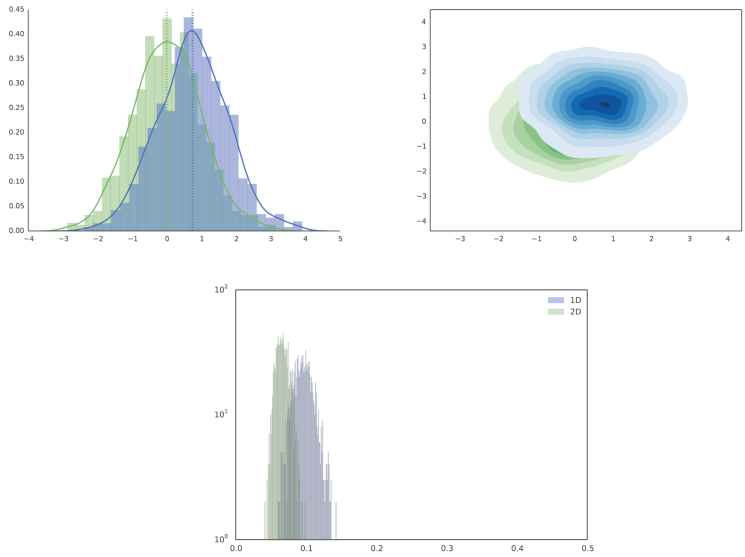


**Figure 5.7** The difference in distributions when epsilon median is smaller than 0.1 in 1D and 2D

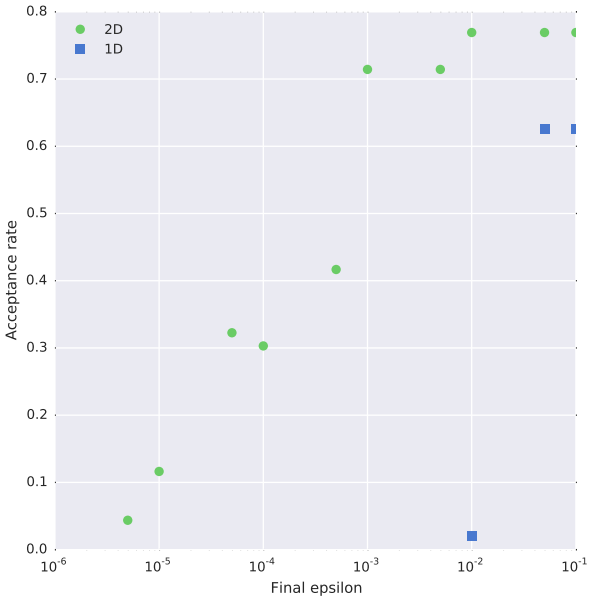


**Figure 5.8** Acceptance rate drops rapidly in the 1D case

#### 5.3.2.2 Uniform distribution

We study the epsilon distribution variance in the uniform distribution, [0, 1].

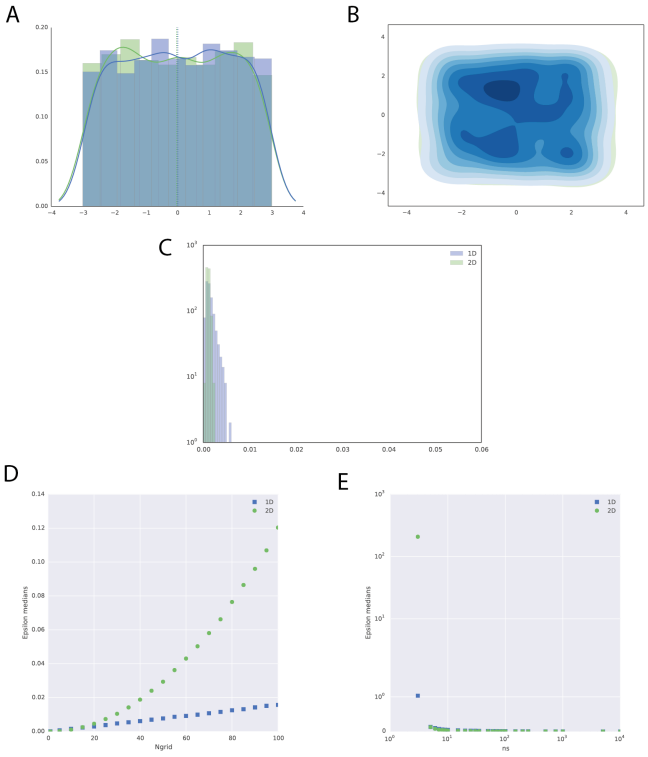


**Figure 5.9** Comparing the 1D and 2D distances between uniform distributions. (A) and (B) show samples of the uniform distributions compared in 1D and 2D respectively. (C) The epsilon distributions in the 1D and 2D distances are equivalent (D,E) Epsilon medians and variance varies with how dense the grid is in the 2D case.

#### 5.3.2.3 Comparing uniform and normal distributions

When comparing a normally distributed data set to a uniformly distributed data set, we don't see a great difference between the 1D and 2D epsilons.

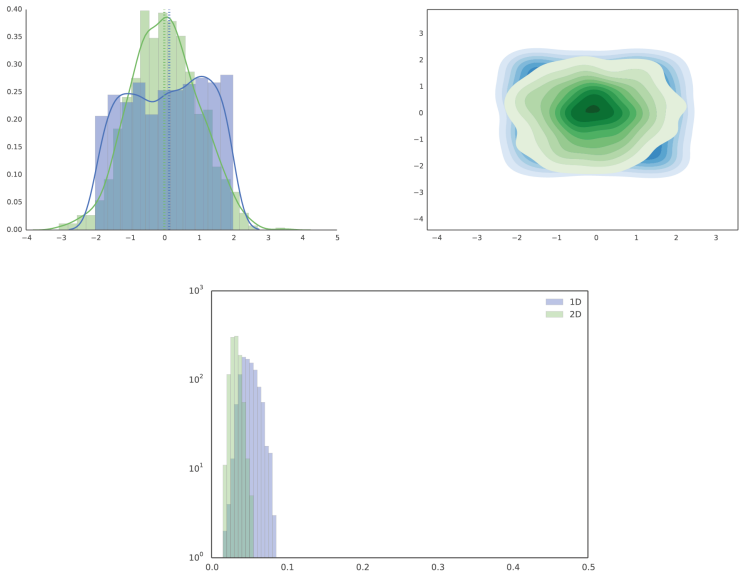


**Figure 5.10** Comparing normally distributed data to uniformly distributed simulations.

#### 5.3.2.4 Bimodal distributions

Another type of distribution that is commonly found in Flow cytometry experiments, is a bimodal distribution. Here we test whether the 1D and 2D distances are equivalent when measuring distances in this type of distributions.

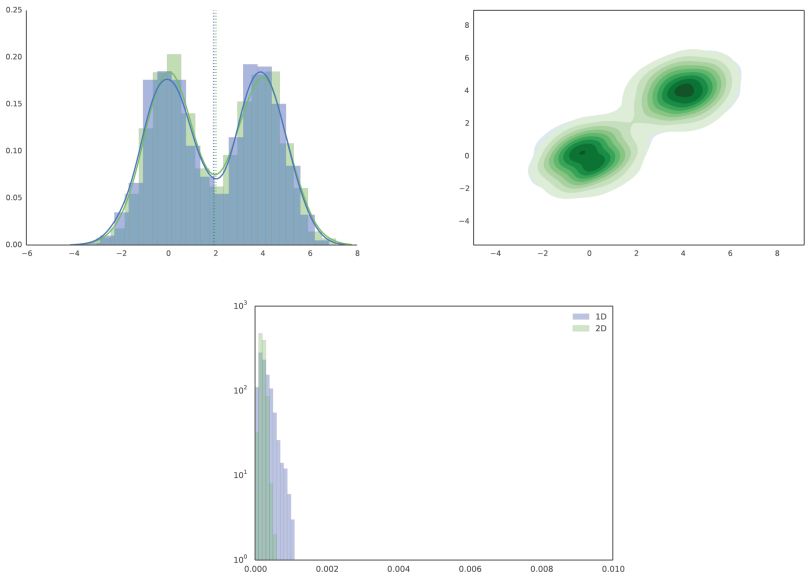


**Figure 5.11** Comparing the 1D and 2D distances between bimodal distributions.

We find that the two distances are comparable when dealing with this kind of distibution. Next, we examine how the distance between bimodal distributions with different $\mu$ varies in the 1D and 2D cases. We find a similar behaviour like the one observed when comparing normal distributions with increasinglt different $\mu$ (Figure 5.6).

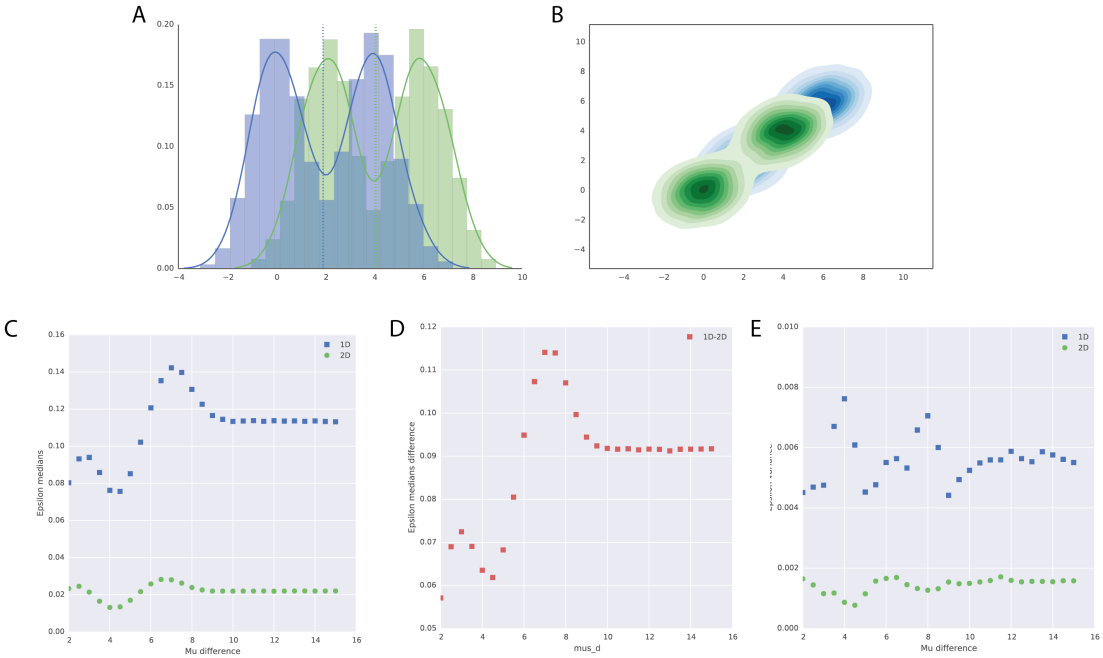


**Figure 5.12** Comparing the 1D and 2D distances between bimodal distributions. (A) and (B) show samples of the bimodal distributions compared in 1D and 2D respectively with a $\mu$ difference of 4 between simulations and data. (C,D) The range by which epsilon median and variance varies as the difference between the $\mu$ of the distributions increases.

#### 5.3.2.5 Comparing bimodal and normal distributions

Finally, we study how the distances perform when comparing a bimodal with a normal distribution. We test the distances by using a bimodal distribution and a series of normal distributions with increasing $\mu$, in 1D and 2D. We find that epsilon is the lowest when the $\mu$ of the normal distribution corresponds to the $\mu$ of one of the two peaks in the bimodal distribution and the highest when there is no overlap between the distributions.

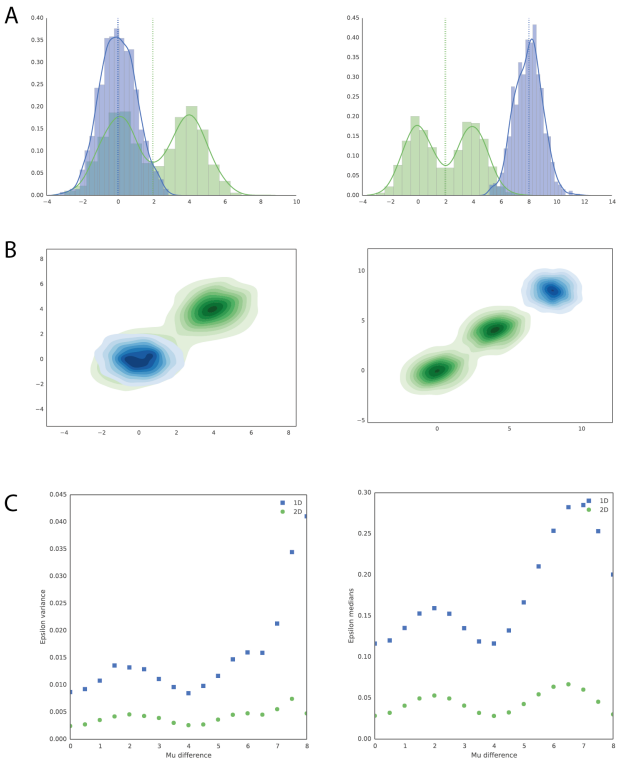


**Figure 5.13** Comparing a multimodal to a normal distribution, in 1D and 2D. (A, B) We vary the *mu* of the normal distribution from equal to the $\mu$ of the first peak of the bimodal distribution to beyond the range of the bimodal distribution. (C) We find that epsilon median and variance are at the lowest when the $\mu$ of the normal distribution is equal to the $\mu$ of one of the peaks of the bimodal distribution.

### 5.3.3 Applying ABC-Flow to simulated Gardner data

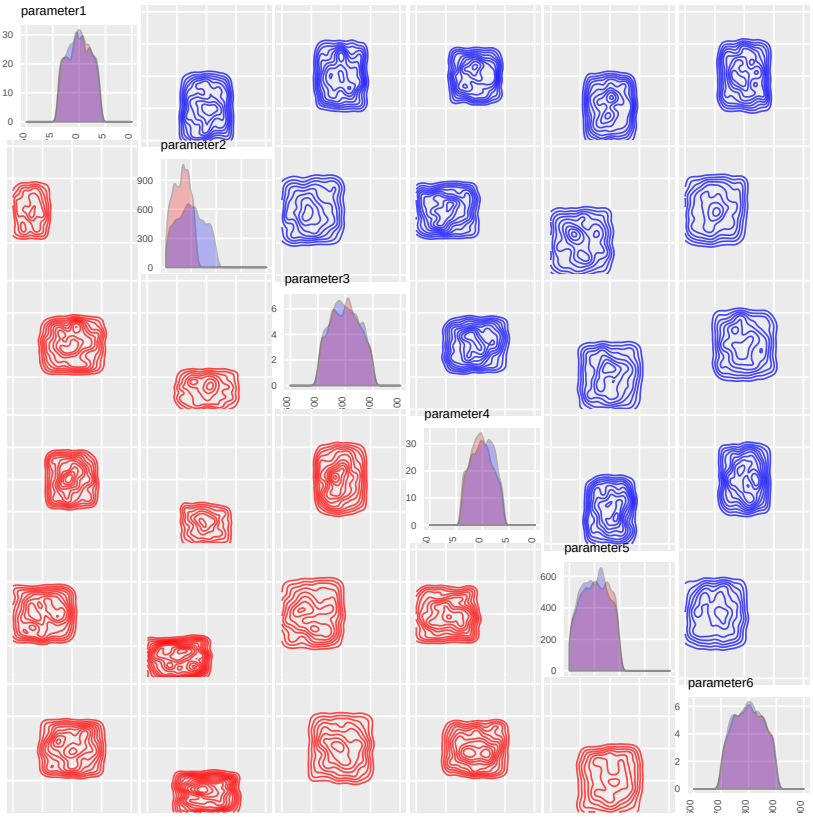


**Figure 5.14** Comparing the fit in 1D (red) Vs 2D (blue) in ABC-Flow, when $\epsilon$=0.1.

In order to improve the identifiability of the parameters in the 2D case, we lower the final $\epsilon$: We find increased identifiability for parameters 2 and 5. Next we will

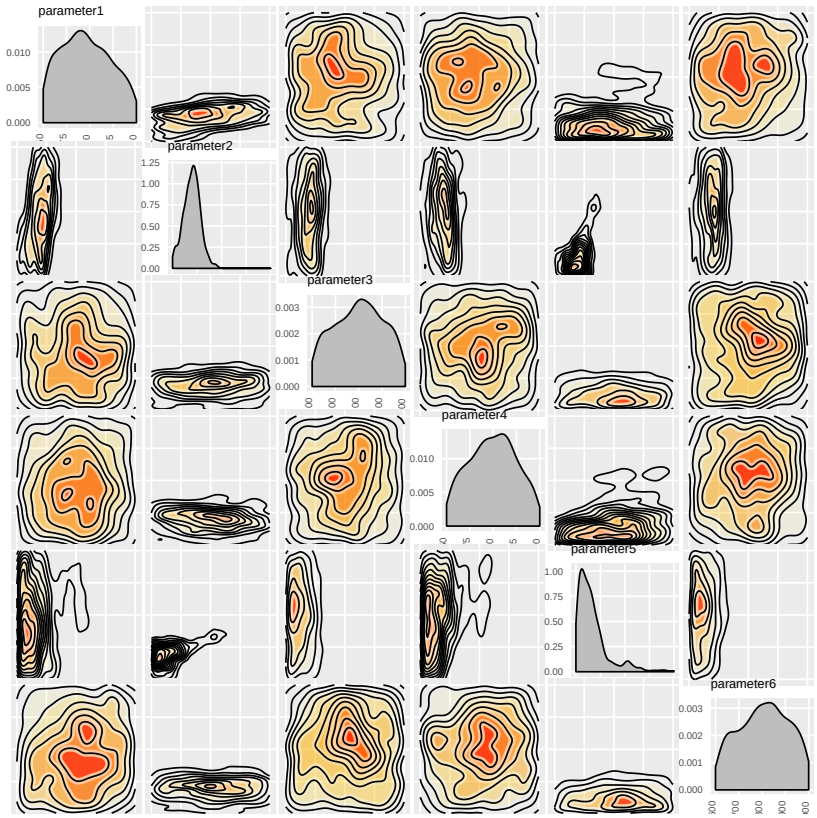


**Figure 5.15** $\epsilon$=0.00001.

use this epsilon in the 1D case.

### 5.3.4 Applying ABC-Flow to experimental toggle switch data

#### 5.3.4.1 ATc induction

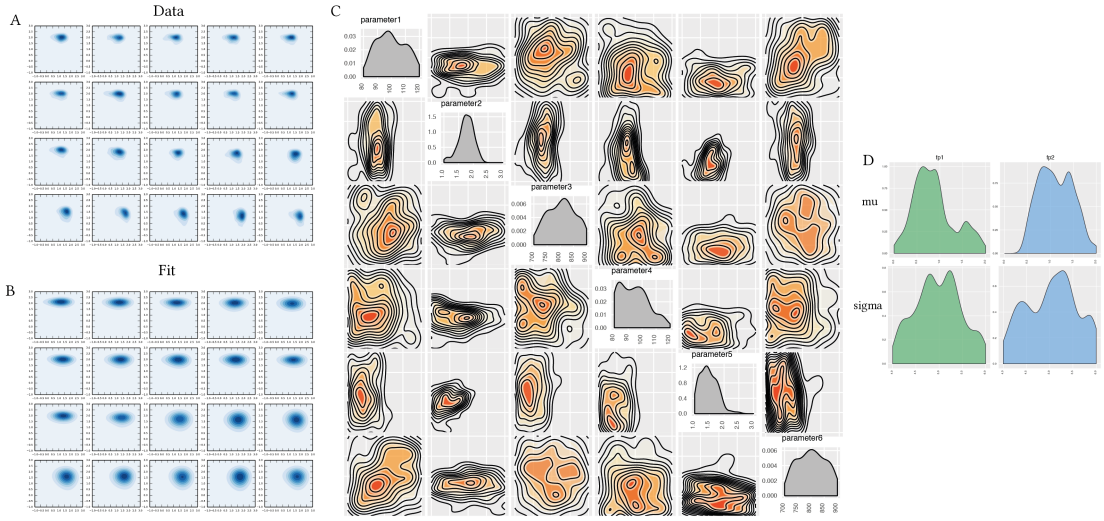


**Figure 5.16** Real data 2D fit

# 6 Designing new switches

# 7 Conclusions

## 7.1 Evaluation

## 7.2 Future work

# Bibliography

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.

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.

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

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

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.

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

Isaacs, F. J., Hasty, J., Cantor, C. R., & Collins, J. J. (2003). 'Prediction and measurement of an autoregulatory genetic module'. *Proceedings of the National Academy of Sciences of the United States of America* **100**(13), 7714–7719.

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.

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.

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

Loinger, A., Lipshtat, A., Balaban, N. Q., & Biham, O. (2007). 'Stochastic simulations of genetic switch systems'. *Physical Review E*.

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

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.

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.

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.

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.

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.

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.

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.

Wilkinson, D. J. (2006a). *Stochastic Modelling for Systems Biology*. CRC Press.

Wilkinson, J. D. (2006b). *Stochastic modelling for systems biology*. CRC Press.

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.

*

# A Appendix

## A.1 Ordinary Differential Equations

### A.1.1 CS-MA

$$
\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{geA}\cdot[\mathrm{gA}]) - V_{\mathrm{cell}} \\
\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 \\
\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( \\
&\quad -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{r} \\
&\quad -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}])
\end{aligned}
$$

### A.1.2 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 \\
&\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( \\
&\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}]) \\
&\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}} \\
\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}
$$

## A.2 Sequences

### A.2.1 pKDL071

## A.3 Algorithms

### A.3.1 Clustering algorithms

#### A.3.1.1 Deterministic case

**Algorithm 6** 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
```

#### A.3.1.2 Stochastic case

**Gap statistic**

**Algorithm 7** 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× 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**

**Algorithm 8** 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
```

**K-means clustering**