School Of Electronics and Computer Science
Faculty of Physical and Applied Sciences
University of Southampton

Argyris Zardilis

April 25, 2013

Tool for parameter inference in dynamic biological systems

Project Supervisor: Dr. Srinandan Dasmahapatra
Second Examiner: Dr. Markus Brede

A final report submitted for the award of
BSc Computer Science

Acknowledgements

And I would like to acknowledge ...

Abstract

Recent advances in experimental technology have given us detailed and comprehensive information on networks of biological interactions governing various functions of living organisms. This had led to an increase in the use of theoretical mathematical models to describe dynamic biological systems which has in turn led to an increasing need for computational tools to assist in the process of constructing these models and estimating their parameters from available experimental data. Although there is a rich literature on parameter estimation using a number of different techniques, very few attempts have been made to systematically attack the parameter estimation problem for these kinds of systems. From those, almost all of them attempt a mere reproduction of experimental data, disregarding important properties of those systems like their qualitative features and the effect of global system dynamics.

In this study a computational tool has been produced tackling the parameter estimation problem in dynamic biological systems using different techniques and its success has been tested with real world models. Further to that, an attempt has been made to uncover the link and interplay between the practical considerations of the parameter inference process and the more theoretical tools of sensitivity and bifurcation analysis and therefore consequently to systems dynamics which are captured and understood through these.

Contents

List of Figures

Chapter 1

Introduction

Cells of living organisms contain thousands of networks of biochemical interactions that perform their functions. Despite being, at the molecular level, subject to thermal noise and the random tinkering of evolutionary events they are able to perform their functions almost always without error and with remarkable consistency. The apparent high complexity of such networks has been perhaps one of the reasons that the main bulk of biological research has been focused on understanding the details of individual components instead of a systems-level understanding of their interactions. So in a way Biology did not follow the route of other natural sciences, like Physics for example, to try and find simplifying and general principles and laws that govern the behaviour of the natural systems occurring in living organisms using the theoretical framework of Mathematics. It might be because the general consensus was that living organisms are so complex and their behaviour is so highly stochastic at the molecular level that one will not be able to find these laws and express them using a rigorous language like Mathematics as used for inanimate objects.

However recent advances in experimental technology like microarray experiments which enabled us to get the expression levels of a big number of genes at different time points have given us for the first time such detailed and comprehensive information on cellular processes. Although the structure of such interactions has been assembled from the effects of many random evolutionary events over time, with this new volume of information we are able to find common reoccurring network structures which evolution keeps turning to because exactly they are the ones that survive with their robustness. This leads us to think that they are common design principles behind these common network motifs that drive them so the goal of finding generalising and simplifying principles and gain systems level understanding through a theoretical framework inside the apparent complexity is revisited recently shifting some of the attention of biological research from experimentation to more theoretical work [Alon, 2007]. Commonly occurring motifs (network structures) include switches, buzzers, blinkers [Tyson et al., 2003], the inspiration for the decomposition and naming also coming from their resemblance to electronic-like components.

These common network motifs can then be assembled and be parts of larger networks of interactions that perform more complicated functions. Such networks that their network structure is though to be well understood and have been the subject of much study both experimentally but even more theoretically include circadian oscillators and metabolic pathways [Bass and Takahashi, 2010; Mirsky et al., 2009; Sahar and Sassone-Corsi, 2012]. As their name suggests one of the main properties of such networks of interactions is their structure, what are the interacting components and which one acts on what. The traditional way to capture this structure is with schematic means with the aid of a diagram an example of which can be seen in Figure 1.1. Components of the networks are gene mRNA products and proteins which can either serve a biochemical function or act as transcription factors for genes. Transcription factors can act on genes to either activate:increase the rate of production or repress:decrease the rate of production of one or more genes. The gene products themselves though control the production of other proteins which can act as transcription factors(activators or repressors) to other genes or even themselves thus creating networks of interactions of varying complexity. The components of the network are the nodes of the diagram with their differences highlighted with the use of different shapes or colours while the interactions(regulations) are the edges differentiated by their ends; for example the flat end signifies repression while the arrow signifies activation. Several attempts to standardise these diagrams in the example of circuit diagrams in electronics (Molecular Interaction Maps [Kohn et al., 2006]) have not been widely adopted so they still vary considerably across publications.

The network structure, though important, is only one of the properties that help us get insight into the workings of these systems. Another essential property which becomes even more apparent as the diagrams and networks grow in size is the systems dynamics. The diagrammatic way of describing the systems is very poor in that sense as it does not capture the dynamic properties that these systems exhibit [Kitano, 2002]. The natural way to capture the dynamic behaviour of systems that is being used throughout science is

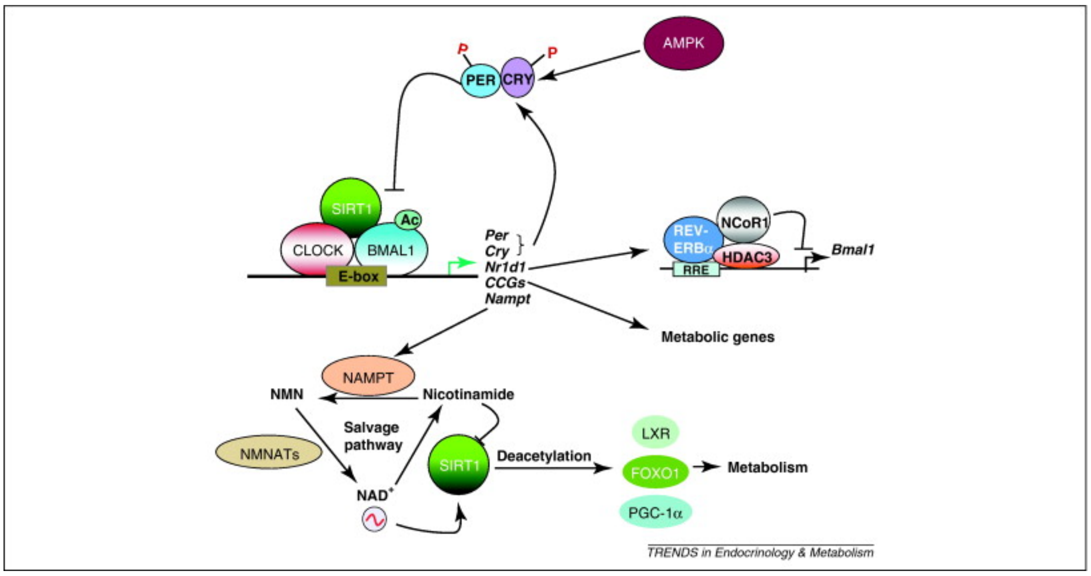


Figure 1.1: Networks of interactions between circadian clock and metabolic pathways [Sahar and Sassone-Corsi, 2012]

with differential equations. That way we can use a more theoretical and well researched framework of Mathematics and of dynamical systems in particular to talk about these systems and borrow tools from it that aid the analysis process and give us greater insight to eventually build a systems-level understanding of complex biological systems. The we can ask more complicated and interesting questions like: what are the control mechanisms that drive these systems? how does one part affect others? Which parts provide the stability and robustness to the system? How will the system behave in the future with the same or varying conditions? The diagrams which offer the static structure provide very little knowledge and surely cannot answer these questions.

Systems can be thought of as stochastic processes as they are subject to molecular noise and various fluctuations in their interactions. Therefore they can be expressed as stochastic differential equations or their evolution can be simulated with the Gillespie algorithm [Gillespie, 1977] or one of its more recent variants [Gibson and Bruck, 2000; Gillespie, 2001]. However the effects of these fluctuations are only considerable when the number of molecules of the reactants is very small so more often the average case is considered and the systems are expressed as deterministic Ordinary Differential Equations(ODEs) like so:

$$\dot{\mathbf{x}} = \mathbf{F}(\mathbf{x}, t, \theta) \tag{1.1}$$

These describe the evolution in the concentrations of components of the systems(variables)

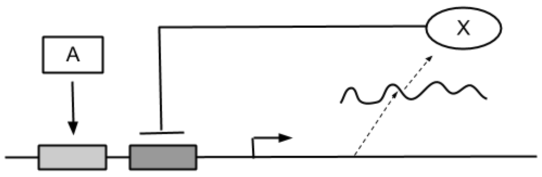


Figure 1.2: Negative autoregulation network motif: Gene X and its protein product are regulated by A. Protein product X also represses its own production [Alon, 2007](chapter 1)

over time and are parameterised by θ which mainly describes the kinetic rates of the reactions/regulation or other constants like initial concentrations. So the evolution of the concentration of component $x1$ might be described by $f(\mathbf{x}, t, \theta)$ which tells us how the concentration is altered through the effects of other variables(components) or itself over time. These equations are translations of the diagrams and the duality of the representation can be seen through an example. Let's consider the a simple synthetic network motif seen in Figure1.2. This is a common negative autoregulation motif which occurs in 40% of known transcription factors in E.Coli [Rosenfeld et al., 2002]. In this network gene X(and its protein product) is regulated by A. The protein X acts as a transcription factor for its own gene repressing its own production. We are concerned with the concentration of protein product X so that is one of our variables. Now we need to express the change in its concentration as an equation. The production of protein X is balanced by the dilution due to the growing area of the cell and the degradation due to its natural destruction. Therefore the change in the concentration can be expressed as the difference between the two processes(production and dilution/degradation). The regulation of X by A(transcription and translation) is orders of magnitude faster than the change in the concentration of X so we can assume the production rate due to A's regulation is constant at a rate β. However the production rate is also affected negatively by X itself so it should be a decreasing function of X with β as its maximum value when $X = 0$. One such function that is found to agree with observation is the Hill function: $f(X) = \frac{\beta}{1+\left(\frac{X}{K}\right)^n}$. The dilution/degradation process at a rate α is proportional to its concentration so the resulting differential equation for the concentration of X is:

$$\frac{dX}{dt} = \frac{\beta}{1 + \left(\frac{X}{K}\right)^n} - \alpha X \tag{1.2}$$

The Hill equation for the repressor can be derived from simple chemical kinetic laws like the law of mass-action or Michaelis-Menten kinetics (see Appendix A in [Alon, 2007] for a detailed derivation and explanation). Starting from these simple laws one can describe as equations all reactions and therefore all systems. The intuition and background knowledge of the modeler though is needed for simplifying the models, for example as we have done for the transcription factor A where we treated it as constant due the differences in timescales between the processes(Quasi-steady-state approximation).

For this simple model, one could easily understand the dynamics of it just by looking at it. However for most systems this is not very easy and most of these systems cannot be solved analytically. This is where the knowledge from the field of dynamical systems comes in. One of the main observations and driving force in field of dynamical systems is: we cannot solve these systems analytically so how can we understand their dynamic behaviour from the differential equations themselves. Of course now we can solve these systems numerically and get good idea of how these systems evolve over time. To solve numerically though we need values for the parameter quantities like β, K, n, α in Eq 1.2. These parameters in these models have a physical interpretation: production/degradation rates, binding affinities, transport rates, etc. This generally poses an issue because quantitative data on these parameters are rarely available because they are difficult to measure experimentally.

The problem therefore is to estimate those parameter values from experimental measurements of the concentrations of the network components over a time period, data which might be scarce or incomplete. This problem sometimes called the inverse problem can be seen as going from the real world observables to the mathematical models that represent those observables. Doing that for parametric models is equivalent to finding the parameters. This inverse problem is a common theme in Machine Learning so there is a large toolbox of available techniques to tackle the problem. However these classical methods are not always suited to such complex systems and might fail. Moreover there are more factors that come into play as far as biological systems are concerned which are not captured by these techniques. The increase in the available computational power has made possible the introduction of new techniques which are based on simulation which have been shown to be more suited to the task. However there have been very few attempts to produce computational tools that aid in the modeling process and attack this problem in this domain despite its importance(see 2 for a more detailed treaty on parameter estimation methods used in this domain and relevant tools). It is with this inference problem and these new techniques that this study is concerned with. The first part of this project has been to use these new techniques to produce a computational tool that

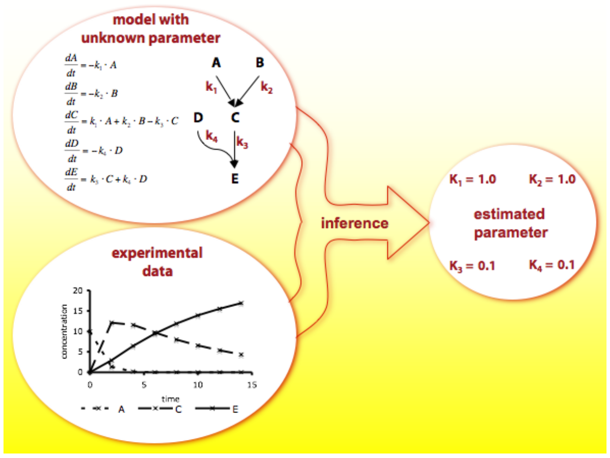

model with unknown parameter
$\frac{dA}{dt} = -k_1 \cdot A$
$\frac{dB}{dt} = -k_2 \cdot B$
$\frac{dC}{dt} = k_1 \cdot A + k_2 \cdot B - k_3 \cdot C$
$\frac{dD}{dt} = -k_4 \cdot D$
$\frac{dE}{dt} = k_3 \cdot C + k_4 \cdot D$
A
B
k_1
k_2
D
C
k_4
k_3
E
inference
$K_1 = 1.0$
$K_2 = 1.0$
estimated parameter
$K_3 = 0.1$
$K_4 = 0.1$
experimental data
concentration
time
A
C
E

Figure 1.3: Parameter inference methods combine experimental data with the knowledge about the underlying structure of a dynamical system to be investigated. Unknown model parameter (here k1, k2, k3 and k4) can be estimated.

automatically and systematically attacks the parameter estimation problem. Discussion and results are covered in chapter 1 of this study.

Work on the first task of this project has led to some observations that potentially link the parameter estimation procedure with some more theoretical tools and aspects of the systems in question. As has been previously discussed systems dynamics play an important role in our understanding of complex biological systems. The language of choice for their description(differential equations) enables us to use some of the tools developed in the field of dynamical systems to investigate the dynamics. Simple models like the one for the negative auto-regulation network motif described previously are easy to understand intuitively but as the models grow intuition alone cannot help. Sensitivity analysis on these models tells us how much the observable behaviour of the system depends on specific parameters of the model. The second part of this study is firstly concerned with the connection between the practical considerations of parameter estimation and the more theoretical tool of sensitivity analysis. Is there a connection between the estimate for a parameter and the model's sensitivity to it? Bifurcation is a change in a system's dynamics as one of its parameters passes through a bifurcation point. In parameter estimation the effects of global dynamics are hardly ever considered despite the fact that their importance in understanding biological systems has been demonstrated . Do these global dynamics affect the parameter estimation procedure? This is also something that we try to address in the second part of this study.

Chapter 2

Background Research

The parameter estimation problem as posed in the previous section is the inverse problem of going from world observables to models that represent those observables. To formalise this problem, we have a proposed model for our system which describes the evolution of all variables(components) $\mathbf{x} = \{x_1, x_2, \ldots, x_n\}$,

$$\begin{aligned} \dot{x}_1 &= f(\mathbf{x}, t, \theta) \\ \dot{x}_2 &= f(\mathbf{x}, t, \theta) \\ &\vdots \\ \dot{x}_n &= f(\mathbf{x}, t, \theta), \end{aligned} \tag{2.1}$$

We also have the observations of the world in the form of experimental data at N different time points, $\mathbf{Y_d} = \{\mathbf{y}_{t_1}, \mathbf{y}_{t_2}, \ldots, \mathbf{y}_{t_N}\}$ where the observation at time point t_k is the state of the system(concentration of n variables) at that point so $\mathbf{y}_{t_k} = \{x_1^{t_k}, x_2^{t_k}, \ldots, x_n^{t_k}\}$. The problem then become to find the parameter vector θ^* from the set of all the possible parameter vectors $\mathbf{\Theta}$ so that the model $\dot{\mathbf{x}} = \mathbf{F}(\mathbf{x}, t, \theta^*)$ 2.1 fits the experimental data best. The fitness of a particular parameter vector θ_s can be assessed with a distance metric between the experimental data $\mathbf{Y}_d$ and a simulated solution $\mathbf{Y}_s$ for proposed parameter θ_s. Then the problem becomes the minimisation of this distance metric and this is the common optimisation problem formulation:

$$\theta^* = \arg\min_{\theta}(\Delta(\mathbf{Y_d}, \mathbf{Y}_\theta))$$

where Δ is some distance metric, also called cost or objective function in optimisation language. Naturally much attention has been given on parameter estimation for deterministic systems with local and global optimisation methods [Moles et al., 2003]. Generally optimisation algorithms have the same outline:

1. Start with some choice θ_s for model's adjustable parameters from $\mathbf{\Theta}$
2. Simulate solution $\mathbf{Y}_s$ for proposed parameter θ_s and evaluate distance/cost function
3. Stop if some termination criterion is met, for example cost function is less than some threshold value ϵ
4. Generate a new guess θ_s from $\mathbf{\Theta}$
5. Return to 2

Then the difference between different choices for optimisation algorithms is how to more efficiently search the parameter space $\mathbf{\Theta}$ and all the classic techniques like steepest-descent, variants of Newton's method, Levenberg-Marquardt have been used in different applications with varying degrees of success [Mendes and Kell, 1998]. Other optimisation techniques used in some application include simulated annealing [Kirkpatrick et al., 1983] which makes stochastic decisions for the movement in the search space $\mathbf{\Theta}$ to avoid common problems of classic optimisation algorithms like getting stuck at local extrema. Attempts to compare different optimisation techniques generally come to the conclusion that the success of the algorithm depends on the application and that there is no single algorithm that outperforms the others in all applications while there is also the risk that some algorithms that perform well in some problems might even fail completely in others(for a full treatment and comparison of a number of different methods see [Mendes and Kell, 1998]).

A maximum-likelihood approach to parameter estimation in this domain has also been used for some applications. In this formulation of the problem, one tries to find the values of the parameters that are more likely to have been used to generate the observed data in other words trying the maximise the likelihood function of the data given the parameters. Likelihood functions are generally associated with probabilistic models and Maximum Likelihood Estimation(MLE) with information-theoretic approaches but subject to some assumptions MLE reduces to traditional least-square data fitting,

- The observational errors are normally distributed
- Equivalent positive and negative deviations from the expected values differ by equal amounts

- The errors between samples are independent.

(We will use these assumptions later in the chapter 2 to plot and investigate the shape of the likelihood functions under different values of the bifurcation parameter for different models). The models generally involve non-linear combination of the parameters so the least-squares formulated problem has no closed form solution. Then one has to resort to numerical optimisation techniques like the ones outlined in the previous paragraph. These general optimisation routines are commonly available in many packages for many programming languages and popular programming environments for scientific computation like MATLAB , Octave, scipy(Python).

In population genetics applications Approximate Bayesian Computation(we will to refer to them as ABC from now on) in its most basic rejection sampling form has been used [Pritchard et al., 1999]. The traditional Bayesian formulation of the problem is to try to find the posterior distribution of the parameters θ given data $\mathbf{Y}_d$, $\pi(\theta|\mathbf{Y}_d)$ given in this setting by $\pi(\theta|\mathbf{Y}_d) = L(\theta)\pi(\theta)$ where $L(\theta)$ is the likelihood function and $\pi(\theta)$ is the prior distribution of the parameters. ABC methods use a simulation-based procedure to eliminate the computation of the likelihood function in cases where it is intractable or impossible to do so. Other forms of ABC which build and improved on the same idea include Markov Chain Monte Carlo methods [Marjoram et al., 2003] and different variants of Sequential Monte Carlo [Del Moral et al., 2006; Sisson et al., 2007; Toni et al., 2009]. These became more popular recently due to the increase in computational power and that can be seen by the amount of recent publications. It is with these different forms of these ABC methods that we will be concerned with in this study. These methods have been shown to work without considerable modification in a diverse range of applications [Toni et al., 2009] and also provide richer information which can be used to get some insight into some aspects of the dynamical systems in question as we will do in the second part of this report.

One of the problems with these numerical approaches is that they disregard qualitative features of the models that are equally if not more important than quantitative and exact replication of experimental data. Some qualitative feature which their importance has been discussed and that do not show up in the numeric data include specific phase-response curves for circadian oscillators [Pfeuty et al., 2011]There have been a few attempts to capture some qualitative features like the shape of the data for model selection purposes. They rely on comparisons of the Fourier Transform of the original dataset and simulated dataset to select the model that best fits [Konopka and Rooman, 2010].

We use this comparison between Fourier Transforms as part of the distance function we use for the comparison of original and simulated dataset in the ABC implementations with various degrees of success.

So there is a wide variety of techniques available to use in the problem but as far as application tools go the choice is not that big. There are a few tools (XPP-Auto, GRIND, COPASI) that are used because they allow modelers to get an intuition and greater understanding of the model in question usually by graphical means such as plotting bifurcation diagrams and allowing them to see the changes in the dynamics of the system as parameters change. And although they help in parameter estimation they do not address the problem directly. One tool that addresses the problem directly is the ABC-Sysbio [Liepe et al., 2010] which can be embedded in external code or used as a stand-alone application. This tool does automatic parameter estimation using Sequential Monte Carlo method.

Chapter 3

Parameter Estimation

In this study we have mainly focused on ABC methods as outlined in the previous section. The use of ABC methods arose in applications where the computation of the likelihood function,

$$L(\theta) = f(\mathbf{Y_d}|\theta), \tag{3.1}$$

where $\mathbf{Y_d}$ are realisations of the data and θ is the parameter vector associated with the model, is impossible or impractical. The main theme in ABC approaches is the use of systematic comparisons between real data and simulated data to approximate the Likelihood function and therefore arrive at the desired posterior distribution,

$$p(\theta|\mathbf{Y_d}) \propto f(\mathbf{Y_d}|\theta)\pi(\theta), \tag{3.2}$$

where $\pi(\theta)$ is the prior distribution of parameters θ. Typically simulating from $f(\mathbf{Y_d}|\theta)$ is straightforward. We then compare the simulated dataset $\mathbf{Y_s}$ with the real data $\mathbf{Y_d}$ and accept only those simulations where some distance metric between the two $\Delta(\mathbf{Y_s}, \mathbf{Y_d})$ falls below a certain threshold value ϵ. Generally ABC methods have the following form:

1. Sample a parameter vector θ^* from a proposal distribution $\pi(\theta)$
2. Simulate a dataset $\mathbf{Y_s}$ from the model characterised by conditional probability distribution $f(\mathbf{Y_d}|\theta)$
3. Compare the simulated dataset $\mathbf{Y_s}$ and observations $\mathbf{Y_d}$ according to some distance function Δ. If $\Delta(\mathbf{Y_s}, \mathbf{Y_d}) < \epsilon$ where ϵ is the error tolerance of accepted solutions, then accept $\theta*$ otherwise reject.

The output of the ABC algorithm is a sample from a distribution of the form,

$$p_\epsilon(\theta|\mathbf{Y_d}) \propto \int f(\mathbf{Y_s}|\theta)(\Delta(\mathbf{Y_d}, \mathbf{Y_s} \leq \epsilon)\pi(\theta)\mathrm{d}\mathbf{Y_s}, \tag{3.3}$$

If the chosen value for ϵ is small enough then the obtained distribution $p_\epsilon(\theta|\mathbf{Y_d})$ will be a good approximation for $p(\theta|\mathbf{Y_d})$. Sometimes when the comparison between real and simulated datasets is difficult summary statistics of the datasets, $S(\mathbf{Y_d})$ and $S(\mathbf{Y_s})$ can be compared instead. The ABC scheme as outlined above is not very efficient in its search of the parameters space as it spends a lot of time in areas of very low likelihood where the simulated solution will no resemble the data. To this direction different computational schemes have been devised like Markov Chain Monte Carlo[Marjoram et al., 2003] where previous good choices advise the next choices thus creating a Markov Chain with stationary distribution the posterior and Sequential Monte Carlo methods where you sample with a decreasing ϵ-schedule and from a series of distributions that increasingly resemble the target posterior[Del Moral et al., 2006; Sisson et al., 2007; Toni et al., 2009]. These are the basic flavours of ABC methods although there are variants of each and these were the methods implemented and incorporated for the parameter estimation tool. In the next section there are more detailed accounts for the implemented algorithms followed by test results on 2 systems.

3.1 Simple Rejection Sampler

The simpler ABC algorithm is a simple rejection sampler[Pritchard et al., 1999] which basically follows the general form as outlined before without any modifications. It samples from a prior distribution without taking into account correct guesses and 'blindly' continues to draw samples from the prior:

1. Sample a parameter θ^* from prior distribution $\pi(\theta)$.
2. Simulate a dataset $\mathbf{y}^*$ from model with parameters θ^*.
3. If $\Delta(\mathbf{Y_s}, \mathbf{Y}) < \epsilon$ accept θ^*, otherwise reject.

The distance function used is a simple Euclidean distance metric,

$$\sum_{i=0}^{N}(\mathbf{Y_d}^i - \mathbf{Y_s}^i)^2, \tag{3.4}$$

where $\mathbf{Y_s}$ is the simulated dataset and $\mathbf{Y_d}$ is the observed dataset. This distance metric is related to traditional Maximum Likelihood Estimation subject to some assumptions and we will show that in detail in a later section. Obviously the acceptance rate for this algorithm is really low if the prior distribution is very different from the posterior since it does not take into account previous good guesses to inform the search and keeps 'blindly' jumping in parameter space. This low acceptance rate means a high number of simulation steps are needed to get a reasonable number of accepted samples to approximate the posterior and although the simulation does not have a high computational cost the number of steps needed make this algorithm a poor choice and it was just implemented as a reference point to see the improvements with the more sophisticated algorithms. For precise results see later section.

3.2 Markov Chain Monte Carlo(MCMC)

An improvement to the random search of the simple rejection sampler is offered by a Markov Chain Monte Carlo method in the spirit of the traditional Metropolis-Hastings algorithm but slightly modified to fit this problem[Marjoram et al., 2003]. The intuition behind this algorithm is that you use previous good guesses as a basis for the next sampling thereby creating a Markov Chain with stationary distribution the goal posterior:

1. Initialise $\theta_i, i = 0$.
2. Sample a candidate parameter vector θ^* from a proposal distribution $q(\theta^*|\theta_i)$.
3. Simulate a dataset $\mathbf{y}^*$ from model with parameters θ^*.
4. If $d(\mathbf{y}^*, \mathbf{Y}) < \epsilon$ jump to new value θ^* - making it the next value in the chain, $\theta_{i+1} = \theta^*$- with a probability given by:

$$a = min\left(1, \frac{\pi(\theta^*)q(\theta_i|\theta^*)}{\pi(\theta_i q(\theta^*|\theta_i)}\right) \tag{3.5}$$

5. Set i=i+1 and continue from 2.

One of the decisions that was taken was to use an adaptive Gaussian distribution as a proposal distribution q. The proposal Gaussian from which we sample has big variance at first so that it will make big jumps in the parameter space. Once a good sample is found the variance of the proposal distribution is decreased to stay in the area. But that has the problem of getting stuck at local optimum areas for a long period of time so a rejection count is employed. If in the close vicinity we keep rejecting then the variance of the proposal distribution is increased to move out of that area. That way areas with many accepted samples are oversampled in the final population. It should be noted that univariate Gaussians were used for the proposal distributions, with each parameter in the parameter vector θ coming from a different proposal distribution. The relations between the parameters are however captured in the fact that when a sample is accepted it is accepted as whole. Despite the attempts to avoid the problem of getting stuck at local sub-optima, this algorithm is still prone to doing exactly that. One possible improvement is to change it to the original Metropolis-Hastings and therefore accepting (with low probability) samples that do not come close to the observed data. That way sometimes jumps will be made to other areas which can result in better exploration of the parameter space. This can also give a resulting population with more than one candidate value and then other criteria can be used to evaluate the fitness of the model with both set of candidate parameters.

3.3 Sequential Monte Carlo(SMC)

Another variation of the ABC form is the Sequential Monte Carlo approach[Toni et al., 2009] which combines the common ABC algorithm form with a Sequential Monte Carlo approach. In this algorithm instead of having a single ϵ value as tolerance we have an ϵ schedule of a series of decreasing values $\epsilon_T < \epsilon_{T-1} \cdots < \epsilon_1$. We start by sampling from the prior $\pi(\theta)$ as before at algorithmic time $t = 1$ with a tolerance level ϵ_1. Then we accept or reject the generated samples with a comparison between real and simulated data in the standard ABC fashion. The population of accepted particles is therefore an approximation for $p_{\epsilon_1}(\mathbf{Y_d}|\theta)$. Then at algorithmic time $t = 2$ instead of sampling again from the prior $\pi(\theta)$ we sample from the distribution obtained at the previous step $p_{\epsilon_1}(\mathbf{Y_d}|\theta)$. The new samples are perturbed with a perturbation kernel. The perturbed particles are then accepted or rejected as before subject to new tolerance level ϵ_2 thus creating the second population. This procedure is repeated for all all T tolerance levels in the ϵ-schedule therefore creating distributions $\{p_{\epsilon_1}(\mathbf{Y_d}|\theta)$,$p_{\epsilon_2}(\mathbf{Y_d}|\theta)$,$\ldots$, $p_{\epsilon_T}(\mathbf{Y_d}|\theta)\}$ along the way. If the final ϵ_T is small enough then the final population should be a good approximation of the target posterior. The algorithm therefore describes a particle-filtering procedure -we pass and filter a population of parameter values sampled from a prior $\pi(\theta)$ from a series of distributions with the decreasing ϵ-schedule so that the

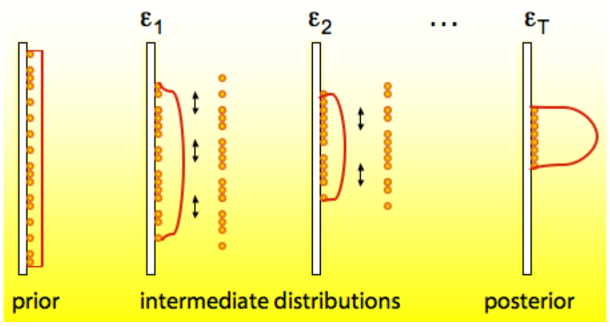


Figure 3.1: In the SMC algorithm a sample from the prior distribution $\pi(\theta)$ is passed through filtering steps where the probability of the particles to represent the data best is updated during the algorithm via intermediate distributions. The final distribution p_{ϵ_T} is an approximation of the posterior distribution. The SMC algorithm follows a particle-filtering approach.

intermediate distributions gradually go towards the posterior(this particle-filtering view can be better seen with Figure3.1). The particular SMC algorithm that we use is from [Toni et al., 2009] which also uses the notion of importance sampling, assigning weights to each particle in the population and then in the next step sampling from the weighted distribution,

1. Initialise the ϵ-sequence $\epsilon_1, \dots, \epsilon_T$
2. If $t = 0$, sample $\theta^* *$ from prior $\pi(\theta)$
   Else, sample θ^* from previous population θ^i_{t-1} with weights w_{t-1} and perturb it with a perturbation kernel $K_t(\theta|\theta^*)$ to get particle θ^{**}.
3. Simulate dataset $\mathbf{y}^*$ from model with parameters θ^{**}.
4. if $d(\mathbf{y}^*, \mathbf{Y}) > \epsilon_t$ go back to 2, otherwise accept particle -set $\theta^{(i)}_t = \theta^{**}$- and calculate its associated weight by:

$$w_t^{(i)} = \begin{cases} 1, & \text{if } t = 0 \\ \frac{\pi(\theta_t^{(i)})}{\sum_{j=1}^N w_{t-1}^{(j)} K_t(\theta_{t-1}^{(j)}, \theta_t^{(i)})}, & \text{if } t > 0 \end{cases} \tag{3.6}$$

5. Continue for all $\epsilon \in \{\epsilon_1, \dots, \epsilon_T\}$

It should be noted that in the special case where $T = 1$ the algorithm reduces to the simple rejection algorithm described in section 3.1. The success of the algorithm both in computational and quality of the solution depends on the choice of the ϵ schedule and the perturbation kernel K_t. The effect of the ϵ-schedule is easy to see: if two successive tolerance levels ϵ_t and ϵ_{t+1} are close to each other then the corresponding intermediate distributions $p_{\epsilon_t}(\mathbf{Y_d}|\theta)$ and $p_{\epsilon_{t+1}}(\mathbf{Y_d}|\theta)$ will be similar and therefore the acceptance rate of the filtering procedure at that step will be high with the disadvantage however that the procedure will not much progress towards the posterior. A slowly decreasing ϵ-schedule will therefore results in a high number of filtering steps(large value of T). On the other hand if two successive tolerance values are further apart the opposite holds: the acceptance rate will suffer but the progress of the algorithm will be faster with a lower number of iterations. Despite the easy to understand intuition determining what tolerance values are close and which ones are not depends on the application at hand. Therefore there is the danger that the algorithms completely fails with particular choices of the ϵ-schedule. The common choice until recently was to manually fix the schedule of tolerance values beforehand according to the model however there have been proposals for an adaptive schedule. Although we started with a hand-tuned schedule we changed that to an adaptive choice. At step $t+1$ the choice for the tolerance level, ϵ_{t+1} at that step is based on the a-th quantile of the distances between the accepted simulated data $\{\mathbf{Y_s}^{(i,t)}\}_{1\leq i\leq N}$ and real data $\mathbf{Y_d}$ at previous step t. We have found that the adaptive choice for the schedule with the correct choice of a improves the efficiency of the algorithm significantly for our tests cases. However in [Silk et al., 2012] it is argued that some choices of a may result in the algorithm not converging to the required posterior $p_\epsilon(\cdot|\theta)$ so the value of the choice of the quantile is an important one.

The second algorithm setting that affects the performance is the choice for the perturbation kernel K_t. A local perturbation kernel hardly moves the particles and therefore results in high acceptance rates in the filtering steps if the successive tolerance levels are close enough. A perturbation kernel with higher variance spreads the search more resulting in better exploration of the parameter space but at the cost of low acceptance rates. Various perturbation kernels have been proposed, component-wise normal and uniform, multivariate normal and uniform, more local ones depending on a part of the previous population or on the local Fisher Information. In [Filippi et al., 2011] they also try to find optimality criteria for these kernels. In our implementation at first a component-wise perturbation uniform kernel was used so each parameter was sampled and perturbed independently. The k-th component of parameter vector θ was perturbed with uniform kernel $\sim \mathcal{U}(\theta_k - \sigma_j^t, \theta_k + \sigma_j^t)$. We set the value for σ_j^t to the scale of the previous population,

$$\sigma_j^t \approx (\max_{1\leq j\leq N}\{\theta_k^{t-1}\} - \min_{1\leq j\leq N}\{\theta_k^{t-1}\}). \tag{3.7}$$

We have also tried multivariate kernels and in particular one with M-nearest neighbours. This proceeds in the following way: for each sampled particle θ its M nearest neighbours from the previous population $\{\theta^{(1,t-1)}m1 \leq i \leq N\}$ are selected and the particle is perturbed according to a multivariate Gaussian distribution with mean θ and covariance matrix the empirical covariance matrix of those M-nearest neighbours, $\Sigma^{(t)}_{\theta,M}$.

We provide some results in the Results section?? in an attempt to compare the efficiency between the different settings of the algorithm but for a more detailed and formal treaty of the topic we refer to [Filippi et al., 2011].

3.3.1 Using Fourier Distance as a distance metric

In the work of [Konopka and Rooman, 2010], the observations for the system's variables are treated as signals and their Fourier Transforms are used as a tool for model selection. We try to use the Fourier Transform as part of the distance function Δ that is used to compare real and simulated data in the SMC algorithms in order to try and capture some qualitative features of the system in question like the shape of the functions.For example this is particularly important for applications that require the presence of oscillations. In those cases a mere reproduction of the experimental data is not enough. We created a new value called 'Fourier Distance' between real $\mathbf{Y_d}$ and simulated datasets $\mathbf{Y_s}$ defined as the average distance between the dominant components of the Fourier Transforms of the constituent signals(variables) of the two.

3.4 Results

In this section we will test the implemented algorithms on two models. For the first model we will focus on the differences between ABC flavours, rejection sampling, MCMC, SMC. For the second model we will only test the SMC algorithm and focus on the differences between different settings of the algorithm namely the choice for ϵ-schedule and perturbation kernel. The second model is also chosen purposefully to illustrate some of the richness of information obtained from the SMC algorithm regarding system's dynamics and sensitivity.

3.4.1 Parameter inference for the Lotka-Volterra model

The algorithms were first tested on the simple Lotka-Volterra predator-prey system[Lotka, 1925]. It is a well studied system which consists of two coupled non-linear differential equations which correspond in the original interpretation to populations of predators and prey,

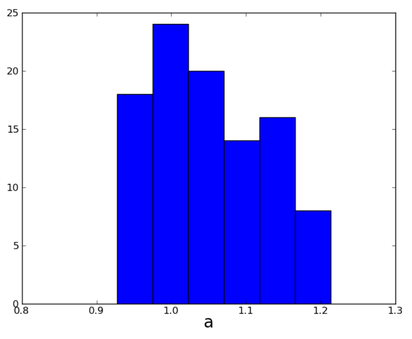
$$\begin{aligned} \dot{x} &= ax - xy \\ \dot{y} &= bxy - y \end{aligned} \tag{3.8}$$

The system has two fixed points at $(0,0)$ and $(a, 1/b)$. We are interested in the second fixed point because closed trajectories surround it in phase space corresponding to solutions which oscillate around that fixed point. We set the parameter values $(a, b) = (1, 1)$ which make the second fixed point $(1, 1)$ so we set initial conditions around it $(x, y) = (1, 0.5)$ which result in oscillating solutions. We create a synthetic dataset from 8 points chosen at regular intervals in the range $[0, 15]$(for exact dataset used see Appendix). Then we add white Gaussian noise to each data point according to $\sim \mathcal{N}(0, (0.5)^2)$ to account for observational errors and make the dataset more realistic. We use the standard euclidean distance $\Delta(\{x_d, y_d\}, \{x, y\})$ between the real dataset $\{x_d[i], y_d[i]\}$ and simulated dataset $\{x[i], y[i]\}$ given as usual as the sum of squared errors between them,

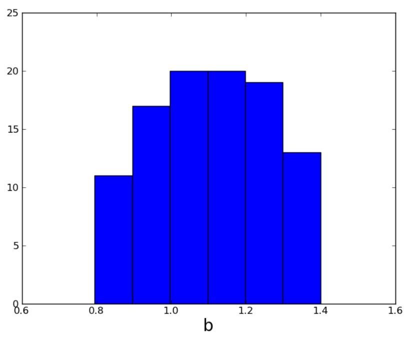
$$\Delta(\{x_d, y_d\}, \{x, y\}) = \sum_i \left((x[i] - x_d[i])^2 + (y[i] - y_d[i])^2 \right). \tag{3.9}$$

The choice for the euclidean distance comes from its relation to the likelihood function which is demonstrated in a later section(4.1.2). We tested the distance between real and noisy datasets and found that averaged over many runs it is equal to ≈ 4 so we deemed the tolerance level of $\epsilon = 4.5$ as a good choice.

We first tested with the simple rejection sampler as described in section3.1. The priors were taken to be uniform for both parameters: $a \approx \mathcal{U}(-10, 10), b \approx \mathcal{U}(-10, 10)$. In order to accept 100 particles 307816 simulation steps were needed which gives an acceptance rate of around 3×10^{-4} which is, as expected, too low. The parameters are well inferred, a : median=1.04, interquartile range=0.12, b: median=1.1, interquartile range=0.25 and the inferred posterior distributions, $p(a|\mathbf{Y_d})$ and $p(b|\mathbf{Y_d})$, can be seen in Figure3.2. The only setting that has to be manually adjusted for this algorithm is the tolerance level ϵ which makes it , despite its obvious inefficiencies, an attractive choice especially for applications where there are information on the parameter values beforehand and therefore the priors are close to the desired posteriors.



(a) $p_{\epsilon=4.5}(a|\mathbf{Y_d})$



(b) $p_{\epsilon=4.5}(b|\mathbf{Y_d})$

Figure 3.2: Posteriors distributions for the two parameters a and b of the Lotka-Volterra model inferred with the simple rejection sampler.

We then test the SMC algorithm as described in section3.3 with different settings of the algorithm. Again we try to infer parameters posteriors distribution for the two parameters, $p(a|\mathbf{Y_d})$ and $p(a|\mathbf{Y_d})$. The priors for both parameters as before are set to $a \approx \mathcal{U}(-10, 10), b \approx \mathcal{U}(-10, 10)$. We tested with a manually tuned ϵ-schedule, $\{30.0, 16.0, 6.0, 5.0, 4.3\}$ and an adaptive ϵ-schedule based on the 0.1-th quantile of the distances of the previous population. The perturbation kernel K_t was set to a component-wise uniform one to the scale of the previous population. For the manually tuned schedule in order to accept 100 particles in the last population 26210 steps were needed in total over all iterations. The cumulative number of steps taken over the course of the algorithm's iterations can be seen in Table3.1. The numbers verify the claims in the previous section: the big difference between ϵ_1 and ϵ_2 is reflected in the big number of simulation steps needed to go from $t = 1$ to $t = 2$. On the other hand very few simulation steps are needed in later iterations due to the small difference between the corresponding tolerance levels. The inferred posteriors in Figure3.4 agree with the values used to generate the synthetic dataset, a:[median=1.04, interquartile range=0.13], b:[median=1.03, interquartile range=0.32]. The quality of the solution corresponding to the inferred parameter values can also been seen in Figure3.4.

For the adaptive scheduled in order to accept 100 particles at the last iteration 11030 steps were needed in total for all iterations which is half of that needed in the manually tuned schedule experiment. The adaptive choice can be more efficient but care has to be taken when choosing the value of α because a too aggressive choice can sometime pay off making the process more efficient but other times can cause the algorithm to get stuck and fail. The cumulative number of steps needed at each iteration can be seen in Table

3.2. We can see from the inferred posteriors and the quality of the solution(Figure 3.3) that the estimates with the adaptive ϵ-schedule are more narrow and very close to the true value. This is due to the fact that the last tolerance level is not set beforehand so if the tolerance level values have not converged at that point the algorithm can keep going with smaller tolerance values until it does.

population	1	2	3	4	5
data generation steps	208	24 844	25735	25 997	26210

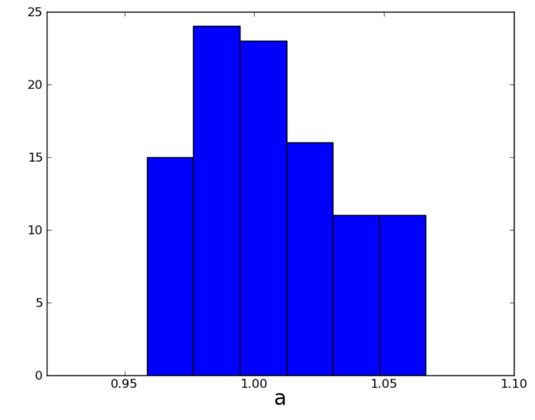
Table 3.1: Cumulative data generation steps needed for each population in the SMC algorithm using a manually tuned ϵ-schedule for the Lotka-Volterra model(component-wise uniform kernel to the scale of the previous population).

population	1	2	3	4	5	6	7	8
data generation steps	162	2972	4983	9359	9841	10202	10602	11030

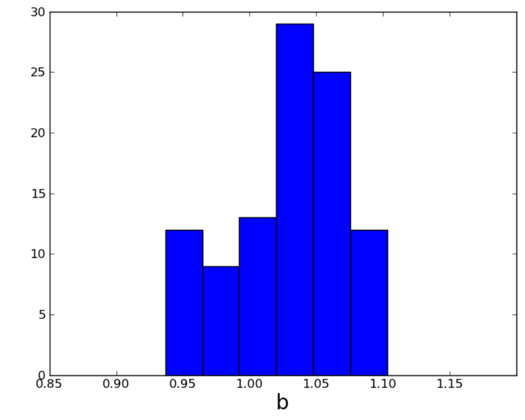
Table 3.2: Cumulative data generation steps needed for each population in the SMC algorithm using an adaptive ϵ-schedule based on the 0.1-th quantile of the previous population for the Lotka-Volterra model(component-wise uniform kernel to the scale of the previous population).

We also tested the algorithm with local multivariate normal kernels with an adaptive choice for the ϵ-schedule as before. The kernels are local in the sense that their covariance matrix is the empirical covariance matrix of the M nearest neighbours from the previous population as explained in the previous section. For this particular example for values of M less than half the size of the previous population we observed that less data generation steps are needed compared to the uniform component-wise case. The total data generation steps needed for different values of M are summarised in Table **??**. As far as the inferred parameters go the results are similar to the results using the uniform kernels.

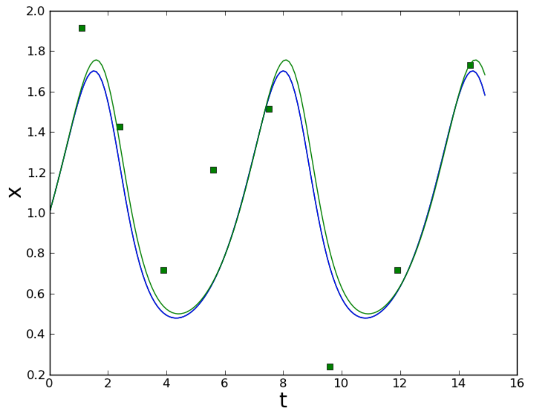
Finally we test the algorithm using the Fourier Distance metric, introduced in a previous section3.3.1. The number of datapoints taken in the interval of interest in previous tests to serve as the dataset in not enough in this case. In order to treat the time series as a signal and take its Fourier Transform we need more datapoints in the interval so for both the original and simulated datasets in the process we took 150 points in the range $[0, 15]$ at regular 0.1 intervals, a number which is not realistic in practise but which we use here to demonstrate the relevance of the Fourier Distance as an adequate summary descriptor for the system in question. The results are comparable to the ones obtained before, a:



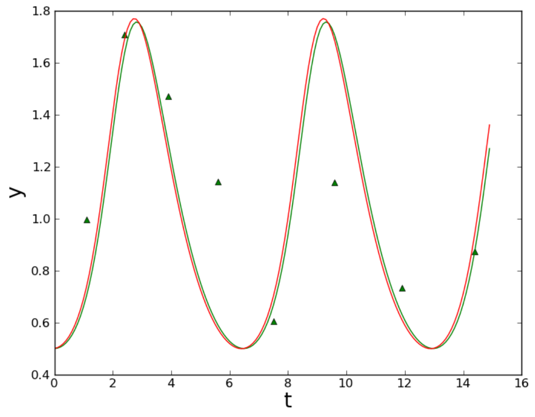
(c)



(d)

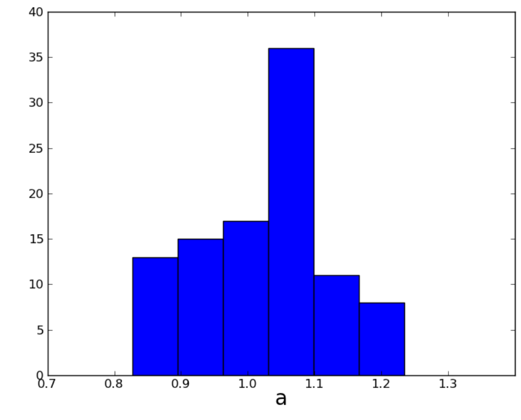


(e)

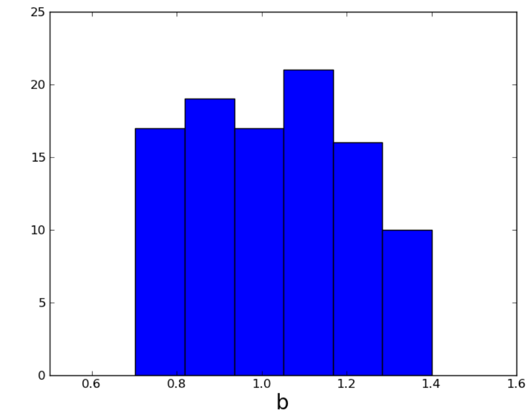


(f)

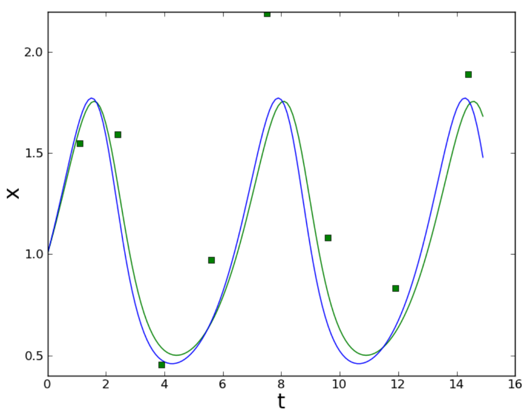
Figure 3.3: Inferred posteriors and quality of solution



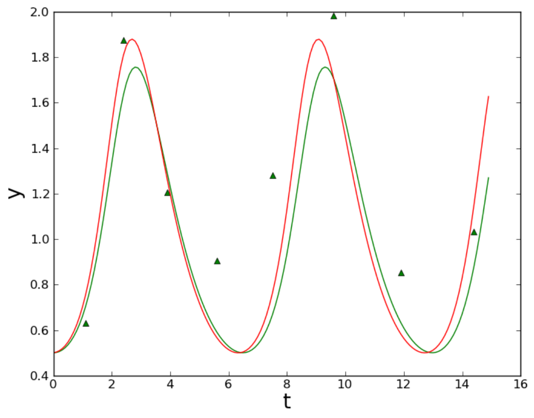
(g)



(h)

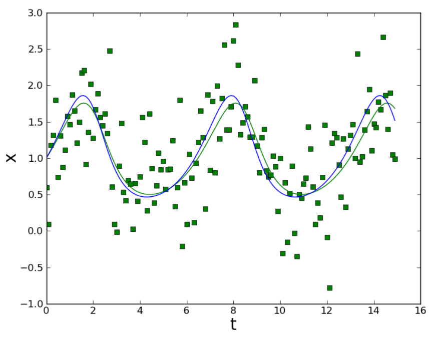


(i)

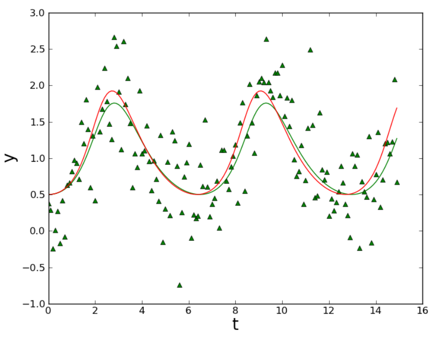


(j)

Figure 3.4: Inferred posterior distributions.



(a) $p_{\epsilon=4.5}(a|\mathbf{Y_d})$



(b)

Figure 3.5: Posteriors distributions for the two parameters a and b of the Lotka-Volterra model inferred with the simple rejection sampler.

median=1.05, b: median=0.99 and the kind of solution that this gives can be seen along with the noisy dataset in Figure3.5. In most of these applications however

3.4.2 Parameter inference for repressilator model

To test the implemented SMC algorithm we turn into a molecular biology example, the repressilator model which was purposefully chosen to highlight some of the challenges associated with parameter estimation and also the utility of the information gathered from ABC algorithms. The repressilator model[Garcia-Ojalvo et al., 2004] is a popular synthetic gene regulatory network. It consists of three genes connected in a feedback loop and each of the genes transcribes for a protein that represses the next gene in the loop with the last one repressing the first(Figure3.6). The evolution of the concentrations of the mRNA products and their corresponding proteins can be written as a system of ODEs,

$$\begin{aligned}
\dot{m}_1 &= -m1 + \tfrac{\alpha}{1+p_3^n} + \alpha_0 \\
\dot{p}_1 &= -\beta(p_1 - m_1) \\
\dot{m}_2 &= -m2 + \tfrac{\alpha}{1+p_1^n} + \alpha_0 \\
\dot{p}_2 &= -\beta(p_2 - m_2) \\
\dot{m}_3 &= -m3 + \tfrac{\alpha}{1+p_2^n} + \alpha_0 \\
\dot{p}_3 &= -\beta(p_3 - m_3),
\end{aligned} \tag{3.10}$$

where m_i denotes the mRNA concentration and p_i the corresponding protein one. The model is parameterised by parameter vector $\theta = (\alpha, \alpha_0, \beta, n)$ where n is the Hill coefficient,

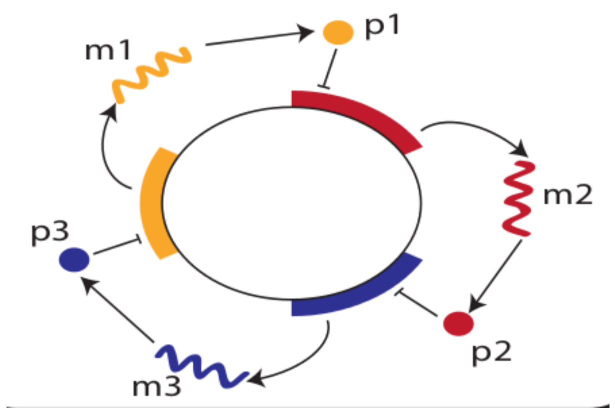


Figure 3.6: The repressilator model. Three genes connected in a feedback loop where the protein product of each gene represses the production of the next gene in the loop.

α the repression strength, β the ratio of the protein decay rate to the mRNA decay rate and a_0 the basal expression rate.

We assume that only the mRNA concentrations $\{m_i\}_1 \leq i \leq 3$ are known. We fix the parameter values at $(\alpha, \alpha_0, \beta, n) = (216.0, 0.216, 5.0, 2.0)$ and create a synthetic dataset consisting of the mRNA concentrations at timepoints taken at regular intervals in the range $[0, 50]$(for the exact dataset used see Appendix).

Chapter 4

System dynamics and infer-ability

In this section we take a more theoretical approach to parameter estimation and investigate how the global dynamics of the system affect the parameter inference process. The qualitative structure of the solutions of dynamic systems can change as their parameter are varied. These qualitative changes in the dynamics are called bifurcations and the parameter values at which they happen are called bifurcation points. There are different kinds of bifurcations depending on different changes in the dynamics of the system. Here we are concerned with bifurcations that give rise to limit cycles in the phase plane of the solutions and therefore are capable of producing oscillatory behaviour. Oscillators are present in many biological systems: cell cycles, circadian rhythms, calcium oscillations, pace maker cells, protein responses. Their behaviour in theoretical models is often captured through limit cycle oscillators which are driven by different kinds of bifurcations. In this section we will investigate two bifurcation capable of creating/destroying limit cycles in two-dimensional systems, Hopf Bifurcation and Saddle-Node bifurcation of Invariant Cycle(SNIC). Systems driven by these bifurcations have been used for real and synthetic oscillatory systems in biology. Their dynamics have also been contrasted in designs of the same synthetic genetic oscillator logical motif [Guantes and Poyatos, 2006]. Since models that use these kinds of bifurcations are used extensively in biology, it is of value to investigate the challenges associated with parameter estimation in those types of models.

In Hopf Bifurcation as the control parameter passes through the bifurcation point, a fixed point of the system loses its stability and the systems shows a limit cycle around that fixed point. In general before the bifurcation point, when the system is temporally perturbed we have damped oscillations which decay and settle down to an equilibrium state(stable fixed point). However in the presence of noise the perturbations are constant and the system does not settle down and therefore it been shown that sustained oscillations can be produced albeit with fluctuating amplitude. After the value of the control parameter passes through the bifurcation point the fixed point loses its stability, a limit cycle is formed around the fixed point and the systems shows sustained oscillations. Both mechanisms, limit-cycle oscillator with unstable fixed point and noise-induced oscillator with stable fixed point, have been proposed as a driving force for oscillatory systems.

In SNIC bifurcation a limit cycle appears as fixed points are created from nothing when the control parameter passes through the bifurcation point. In this case however the stability of the fixed point does not change.

4.1 Methods

4.1.1 Models

For our investigation of the effects of a Hopf bifurcation we chose a very simple system that undergoes a Hopf bifurcation[Strogatz, 2001, p.250]. The simplicity of the model is convenient for our investigation while still maintaining the relevant characteristics. The system given in polar coordinates,

$$\begin{aligned} \dot{r} &= \mu r - r^3 \\ \dot{\theta} &= \omega + br^2. \end{aligned} \tag{4.1}$$

The system has three parameters: μ, ω, b. The control parameter for the bifurcation is μ so we consider the other two known, $(\omega, b) = (1, 1)$ and investigate the behaviour as μ varies. The system has a fixed point $(0, 0)$ which undergoes a Hopf Bifurcation losing its stability as the control parameter μ goes through 0.

For our investigation of the effects of the SNIC bifurcation we again chose a very simple system[Strogatz, 2001, p.250]. The system in polar coordinates,

$$\begin{aligned} \dot{r} &= \mu r + r^3 - r^5 \\ \dot{\theta} &= \omega + br^2. \end{aligned} \tag{4.2}$$

The control parameter for the bifurcation is μ. The radial equation $\dot{r} = \mu r + r^3 - r^5$ undergoes a saddle-node bifurcation of fixed points at $\mu = -1/4$ and two new fixed points are created. When view in the two-dimensional setting these correspond to the creation of limit cycles.

4.1.2 Maximum Likelihood for ODEs

For any mathematical model our goal is to infer true parameters θ^* given some observed data $\mathbf{Y_d} = \{\mathbf{y_{t_1}}, \mathbf{y_{t_2}}, \ldots, \mathbf{y_{t_M}}\}$ at times $\{t_i\}_{1\leq i\leq M}$. To this end we wish to know the likelihood function $L(\theta|\mathbf{Y_d})$ which tells us how the probability of observing the data $\mathbf{Y_d}$ changes with θ. However, a direct approach to obtaining the likelihood function is not possible for deterministic system of ODEs but subject to some assumptions the likelihood function is equivalent to the distance or error between real(observed) and predicted data. That means that the probability of observing the data $\mathbf{Y_d}$ for parameter vector θ_s is equivalent to $\Delta(\mathbf{Y_d}, \mathbf{Y}_{\theta_s})$ where $\mathbf{Y_d}$ is the real data,$\mathbf{Y_d}$ data simulated with parameter θ_s and Δ is the Euclidean distance. This is also the intuition behind using comparisons of real and simulated data with the Euclidean distance as an approximation for the Likelihood function in the ABC algorithms.

Here we follow the likelihood approach for ODE parameter estimation given in . We assume that any observed data point $\mathbf{y_{t_i}}$ is distributed according to $\mathcal{N}(\boldsymbol{\mu}_{t_i}(\theta), \Sigma_{t_i})$, where $\boldsymbol{\mu}_{t_i}(\theta)$ is the solution of the system $\dot{x} = \mathbf{F}(\mathbf{x}, t, \theta)$ at time t_i and Σ_i is a covariance matrix which is the same for all i. Then the likelihood function is given by:

$$L(\theta|\mathbf{Y_d}) = \prod_{i=1}^{M} \frac{1}{(2\pi)^{3/2}|\Sigma|^{1/2}} \exp\left(-\frac{1}{2}(\mathbf{y_{t_i}} - \mu_i(\theta))^T \Sigma^{-1} (\mathbf{y_{t_i}} - \mu_i(\theta))\right). \tag{4.3}$$

We will investigate the behaviour through the log-likelihood function,

$$\begin{aligned} ln(L(\theta|\mathbf{Y_d}) &= Mln\left(\tfrac{1}{(2\pi)^{3/2}|\Sigma|^{1/2}}\right) - \tfrac{1}{2}\textstyle\sum_{i=1}^{M}((\mathbf{y_{t_i}} - \mu_i(\theta))^T\Sigma^{-1}(\mathbf{y_{t_i}} - \mu_i(\theta)) \\ &\propto -\tfrac{1}{2}\textstyle\sum_{i=1}^{M}(d_\Sigma(\mathbf{y_i} - \boldsymbol{\mu_i(\theta)}))^2, \end{aligned} \tag{4.4}$$

where $d_\Sigma(\mathbf{y_i} - \boldsymbol{\mu_i(\theta)}) = \sqrt{(\mathbf{y_{t_i}} - \mu_i(\theta))^T\Sigma^{-1}(\mathbf{y_{t_i}} - \mu_i(\theta)}$ is the Mahalanobis distance between $\mathbf{y_{t_i}}$ and $\boldsymbol{\mu_i(\theta)}$ with respect to Σ. Subject to our assumption that the covariance matrix Σ is equal for i then the Mahalanobis distance is equivalent to Euclidean distance and the log-likelihood function is given by,

$$ln(L(\theta|\mathbf{Y_d}) = -\frac{1}{2}\sum_{i=1}^{M}(d(\mathbf{y_{t_i}}, \mu_i(\theta)))^2, \tag{4.5}$$

where $d(\mathbf{y_{t_i}}, \mu_i(\theta))$ is the Euclidean distance between $\mathbf{y_{t_i}}$ and$\mu_i(\theta))$. It is clear that maximising the likelihood function is equivalent to minimising the distance function d. Obviously the maximum likelihood estimate will be at the value of the true parameter where real and simulated data are the same and the distance value is equal to 0. Therefore this likelihood function is related to the problem of finding how far apart are solutions with the same initial conditions as the parameters vary. This is how the likelihood function is also related to the parameter inference process. The landscape of the likelihood function around the true parameter value can give some information on how easy the parameter estimation process will be. For example flat regions imply regions of highly likely values which are close to the true solution therefore suggesting high acceptance rates in the filtering process of ABC schemes. In the next section we investigate how the shape of the likelihood function and consequently the parameter estimation process changes as the systems undergo a bifurcation.

4.2 Results

4.2.1 Hopf Bifurcation

To understand the effects of Hopf bifurcation we plot the log-likelihood function, $\ln(L(\mu|\mathbf{Y_d}(\mu^*)))$ for parameter μ which is the control parameter for the bifurcation, for different values of μ^* both sides of the bifurcation point. The resulting plots are in 4.1. As expected the maximum likelihood estimator is at the true value. However there is a big qualitative change in the shape of the log-likelihood functions when $\mu > 0$ and when $\mu < 0$.

For the landscape of the log-likelihood plot for values $\mu^* < 0$ we observe a relatively flat region in the interval [-2.0, 0.0] and then a relatively steep part with negative gradient in the interval [0.0, 2.0]. The flat part in the region $-2 \leq \mu \leq 0$ tells us that parameter estimates in that domain are all highly likely.This is easy to understand as the corresponding solutions for estimates in that domain are qualitatively similar to each other and to the solution for μ^*. Therefore the distance between them and also between them and the solution for μ^* is small which is reflected on the likelihood plot. As the value of μ goes through the bifurcation point, the estimates become increasingly unlikely which can also be explained by the fact that the solutions corresponding to estimates in this domain are qualitatively very different from the solution for μ^*. Moreover the bifurcation point at $\mu = 0$ which marks the point of change in system dynamics also marks the change in the behaviour and shape of the likelihood function.

For the landscape of the log-likelihood plot at the other side of the bifurcation for values $\mu^* > 0$ we observe the maximum likelihood estimate, as expected at the true value, but this occurs at a spike in the curve. That tells us that solutions that correspond to parameter estimates which are near the true value μ^* are not very likely so small changes in the control parameter near the true value μ^* result to big changes in the corresponding solution. Again the bifurcation point at $\mu = 0$ is recognisable by the change in the behaviour of the graph at that point. Solutions for estimates in the region [-2, 0] are similar so their distances from the solution for true parameter value are similar, hence the flat part of the graph.On the other hand solutions corresponding to estimates in the region [0, 2] are relatively dissimilar, hence the bigger changes in the distances to the true solution we observe via a steeper descent in the graph in that region.

To understand the change in the distances between adjacent and solutions and the change in the shape of the likelihood function as the control parameter goes through the bifurcation point we can look at what happens to the system dynamics. This will be easier if we look at solutions as trajectories in the phase plane4.4. At one side of the bifurcation point for values of $\mu^* < 0$ the fixed point of the system is stable therefore trajectories near the fixed point will get attracted to it. This makes solutions for nearby parameter estimates in that region qualitatively very similar which makes the distance between them very small. Given the equivalence of distance with likelihood, we expect estimates in that region to be highly likely which is verified by our findings. When the control parameter passes through the bifurcation point, the fixed point of the system becomes unstable therefore trajectories that start near it will be pushed away from it towards the limit cycle around it. This makes solutions that start from the same initial conditions more dissimilar and the distance between them higher. In order to approach the observed data for μ^* the parameter estimate will have to be close to it.

The connection between the shape of the likelihood function and the distance between nearby solutions and the practicalities of the parameter inference process should be apparent. For values $\mu^* < 0$ nearby solutions are similar so we expect a high acceptance rate and less iteration steps at the filtering steps of ABC schemes that use the distance function as a filter. The population obtained from the ABC scheme however will be spread out(high variance) which might decrease the confidence for the estimate. On the other hand from the other side of the bifurcation point and for values $\mu^* > 0$ where nearby solutions are not very similar we expect a lower acceptance rate and more iterations steps. But we expect the obtained population to be more concentrated giving a higher confidence for the estimate. These are indeed verified as we test the SMC scheme for

There is a also an apparent link between the shape of the likelihood function and parameter sensitivity. A spike in the log-likelihood curve tells us that a the system is very sensitive to changes in that parameter. Small changes in the parameter value get amplified in the solution. On the other hand a relatively flat region around the true parameter value tells us that the model is not very sensitive to that parameter as changes to it result in small changes in the corresponding solutions. Hence this is another way to see the importance of the rich information obtained from ABC schemes. ABC schemes return a distribution for the parameter estimate instead of a single value. As we have seen populations of parameter estimates that are concentrated around a specific value with low variance imply low likelihood estimates around the true parameter. This can be translated to a spike in the shape of the log-likelihood function, for example in **??** and therefore to high sensitivity of the system to that parameter. Populations of parameter estimates which are more spread out with high variance imply highly likely estimates around the true parameter value and hence a flat region of the likelihood function which tells us that the system is not very sensitive to that parameter.

4.2.2 SNIC

We proceed to investigate the effects of the SNIC bifurcation in the same way as we have for the the Hopf bifurcation. We plot the likelihood function $\ln(L(\mu|\mathbf{Y_d}(\mu^*)))$ for values of μ^* either side of the bifurcation point at $\mu = -1/4$ for a range of values of μ in the range [-2, 2] taken at regular intervals of 0.01. The resulting log-likelihood plot are in Figure4.3. As expected the maximum likelihood estimator is always at the true value μ^*, however we again notice a qualitative change in the plots either side of the bifurcation point.

For the landscape of the plots of log-likelihood functions for values of parameter before the bifurcation point $\mu^* < -1/4$ there is flat region in the area from [-2, -1/4], then an elbow in the curve at the bifurcation point and then a steeper area with negative gradient. This is qualitatively the same as the log-likelihood functions at one side of the Hopf bifurcation point in the previous section. The flat part tells us that solutions in that range are all relatively close to the true solution and the steep part that solutions in that area become increasingly unlikely. Again this change is due to the change in the system dynamics either side of the bifurcation point.

For the landscape of the plots of log-likelihood function for values of parameter after the bifurcation again we get similar results as in the Hopf bifurcation case when the limit cycle appears in the phase plane. The maximum likelihood estimate at the true value happens as a spike in the curve which tells us that nearby estimates are unlikely so solutions that start at the same initial condition vary considerably as the parameters change. The findings can again be explained by the change to the system dynamics as the system undergoes the bifurcation and to the corresponding solutions as trajectories in the phase plane. For values of the control parameter before the bifurcation point $\mu < -1/4$ the fixed points in the radial equation annihilate each other as it undergoes a saddle-node bifurcation and the limit cycles are destroyed. The fixed point is stable though so trajectories starting near it will get drawn towards it. In that way solutions resemble solutions of the Hopf bifurcation system before the bifurcation point. Because solutions get drawn towards the fixed point which makes solution for different parameter values qualitatively very similar hence the flat part in the region. After the control parameter goes through the bifurcation point at $\mu = -1/4$ the fixed point does not lose its stability as in the Hopf bifurcation case but two fixed points appear in the radial equation which correspond to limit cycles in the two-dimensional case. Therefore solutions that start near that fixed point will get pushed out to the limit cycle which is the same kind of behaviour that we get from the Hopf bifurcating system in the previous section. Nearby solutions will be qualitatively not very similar unless their corresponding control parameter values are really close hence the spike in the log-likelihood curves for $\mu > -1/4$.

So we have found that the two bifurcation that can create limit cycles in the two-dimensional case behave similarly as limit cycles are created even though the mechanism for their creation is not the same.

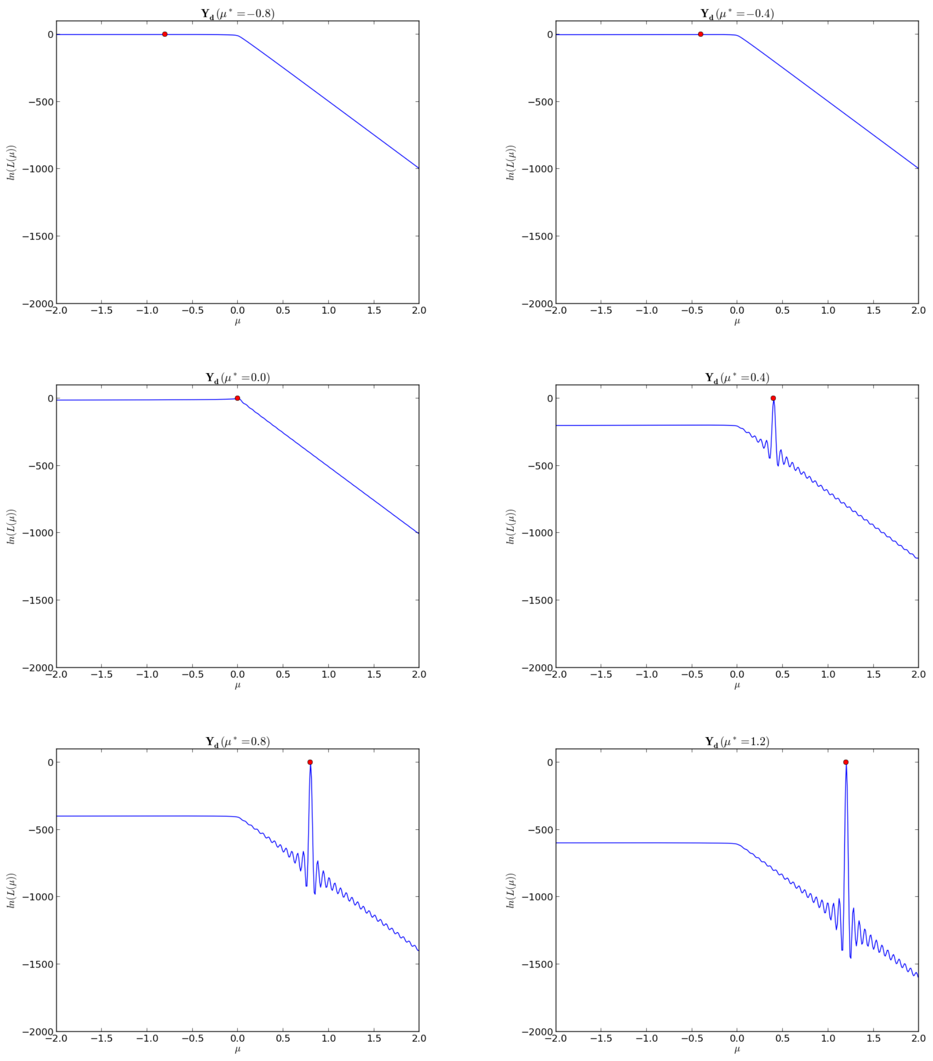


Table 4.1: Log-likelihood plots for different values of μ^* specified in the header. The maximum likelihood is denoted by the red dot which is as expected at the true value.

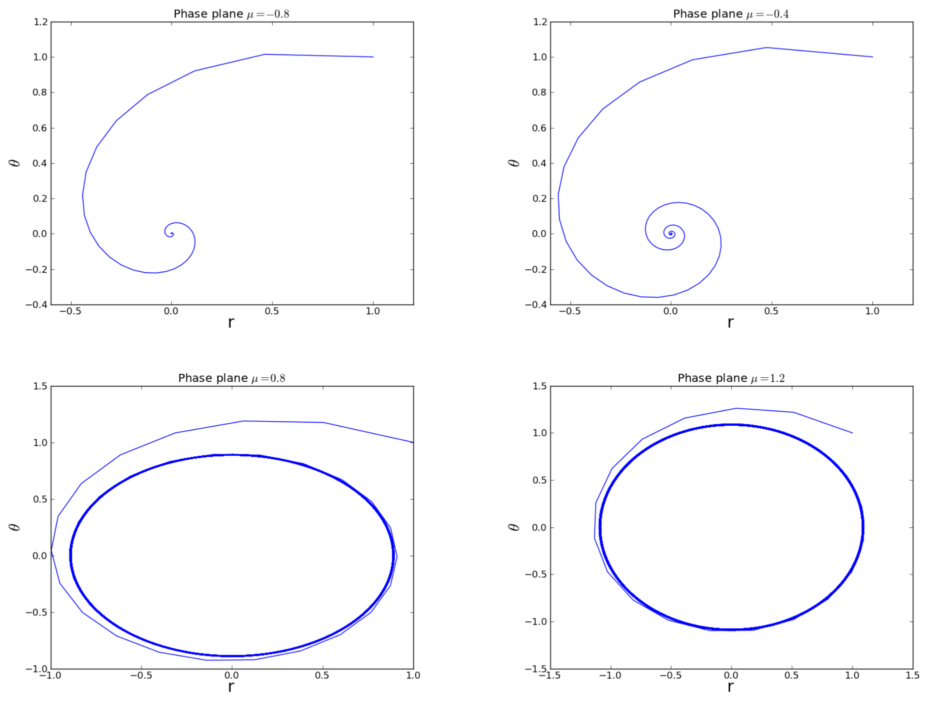


Table 4.2: Solutions corresponding to trajectories in the phase plane for different value of control parameter μ either side of the bifurcation point $\mu = 0$.

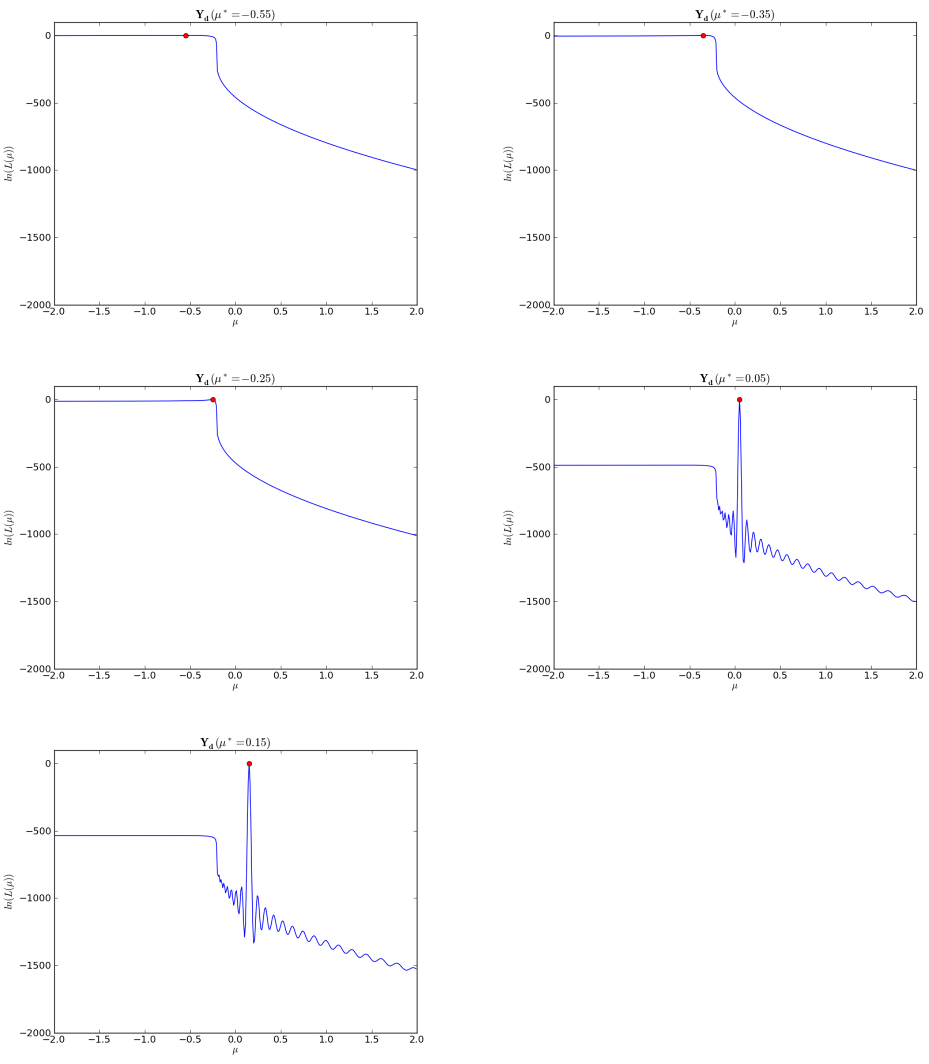


Table 4.3: Log-likelihood plots for different values of μ^* specified in the header. The maximum likelihood is denoted by the red dot which is as expected at the true value.

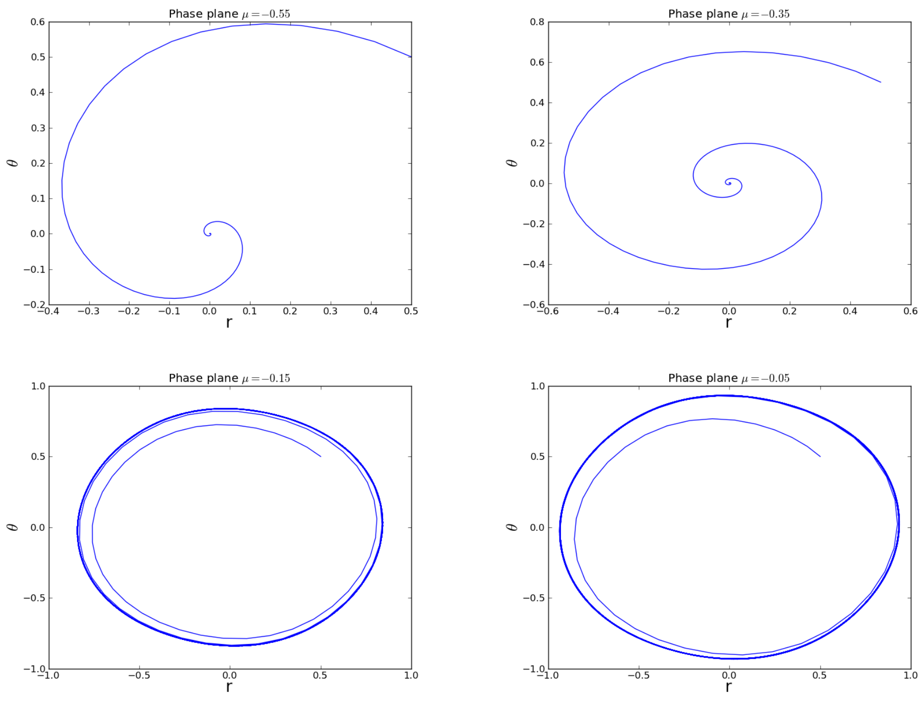


Table 4.4: Solutions corresponding to trajectories in the phase plane for different value of control parameter μ either side of the bifurcation point $\mu = 0$.

Chapter 5

Conclusions

Appdx A

and here I put a bit of postamble ...

Appdx B

and here I put some more postamble ...

References

Uri Alon. *An introduction to systems biology: design principles of biological circuits*, volume 10. Chapman & Hall/CRC, 2007. v, 2, 4

J. Bass and J.S. Takahashi. Circadian integration of metabolism and energetics. *Science Signalling*, 330(6009):1349, 2010. 2

Pierre Del Moral, Arnaud Doucet, and Ajay Jasra. Sequential monte carlo samplers. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 68(3):411–436, 2006. 9, 12

S. Filippi, C. Barnes, J. Cornebise, and M. P. H Stumpf. On optimality of kernels for approximate Bayesian computation using sequential Monte Carlo. *ArXiv e-prints*, June 2011. 16, 17

Jordi Garcia-Ojalvo, Michael B Elowitz, and Steven H Strogatz. Modeling a synthetic multicellular clock: repressilators coupled by quorum sensing. *Proceedings of the National Academy of Sciences of the United States of America*, 101(30):10955–10960, 2004. 23

Michael A Gibson and Jehoshua Bruck. Efficient exact stochastic simulation of chemical systems with many species and many channels. *The journal of physical chemistry A*, 104(9):1876–1889, 2000. 3

Daniel T Gillespie. Exact stochastic simulation of coupled chemical reactions. *The journal of physical chemistry*, 81(25):2340–2361, 1977. 3

Daniel T Gillespie. Approximate accelerated stochastic simulation of chemically reacting systems. *The Journal of Chemical Physics*, 115:1716, 2001. 3

Raúl Guantes and Juan F Poyatos. Dynamical principles of two-component genetic oscillators. *PLoS computational biology*, 2(3):e30, 2006. 25

Scott Kirkpatrick, D. Gelatt Jr., and Mario P Vecchi. Optimization by simulated annealing. *science*, 220(4598):671–680, 1983. 8

Hiroaki Kitano. Computational systems biology. *Nature*, 420(6912):206–210, 2002. 2

Kurt W Kohn, Mirit I Aladjem, John N Weinstein, and Yves Pommier. Molecular interaction maps of bioregulatory networks: a general rubric for systems biology. *Molecular biology of the cell*, 17(1):1–13, 2006. 2

T. Konopka and M. Rooman. Gene expression model (in) validation by fourier analysis. *BMC systems biology*, 4(1):123, 2010. 9, 17

Juliane Liepe, Chris Barnes, Erika Cule, Kamil Erguler, Paul Kirk, Tina Toni, and Michael P.H. Stumpf. Abc-sysbio approximate bayesian computation in python with gpu support. *Bioinformatics*, 26(14):1797–1799, 2010. doi: 10.1093/bioinformatics/btq278. URL `http://bioinformatics.oxfordjournals.org/content/26/14/1797.abstract`. 10

A.J. Lotka. *Elements of physical biology*. Williams & Wilkins Baltimore, 1925. 18

P. Marjoram, J. Molitor, V. Plagnol, and S. Tavaré. Markov chain monte carlo without likelihoods. *Proceedings of the National Academy of Sciences*, 100(26):15324–15328, 2003. 9, 12, 13

Pedro Mendes and D Kell. Non-linear optimization of biochemical pathways: applications to metabolic engineering and parameter estimation. *Bioinformatics*, 14(10):869–883, 1998. 8

H.P. Mirsky, A.C. Liu, D.K. Welsh, S.A. Kay, and F.J. Doyle. A model of the cell-autonomous mammalian circadian clock. *Proceedings of the National Academy of Sciences*, 106(27):11107–11112, 2009. 2

Carmen G. Moles, Pedro Mendes, and Julio R. Banga. Parameter estimation in biochemical pathways: A comparison of global optimization methods. *Genome Research*, 13(11):2467–2474, 2003. doi: 10.1101/gr.1262503. URL `http://genome.cshlp.org/content/13/11/2467.abstract`. 8

B. Pfeuty, Q. Thommen, and M. Lefranc. Robust Entrainment of Circadian Oscillators Requires Specific Phase Response Curves. *Biophysical Journal*, 100:2557–2565, June 2011. doi: 10.1016/j.bpj.2011.04.043. 9

J.K. Pritchard, M.T. Seielstad, A. Perez-Lezaun, and M.W. Feldman. Population growth of human y chromosomes: a study of y chromosome microsatellites. *Molecular Biology and Evolution*, 16(12):1791–1798, 1999. 9, 12

N. Rosenfeld, M.B. Elowitz, U. Alon, et al. Negative autoregulation speeds the response times of transcription networks. *Journal of molecular biology*, 323(5):785–793, 2002. 4

Saurabh Sahar and Paolo Sassone-Corsi. Regulation of metabolism: the circadian clock dictates the time. *Trends in Endocrinology & Metabolism*, 23(1):1–8, 2012. v, 2, 3

Daniel Silk, Saran Filippi, and Michael PH Stumpf. Optimizing threshold-schedules for approximate bayesian computation sequential monte carlo samplers: Applications to molecular systems. *arXiv preprint arXiv:1210.3296*, 2012. 16

SA Sisson, Y Fan, and Mark M Tanaka. Sequential monte carlo without likelihoods. *Proceedings of the National Academy of Sciences*, 104(6):1760–1765, 2007. 9, 12

Steven Strogatz. Nonlinear dynamics and chaos: with applications to physics, biology, chemistry and engineering. 2001. 26

Tina Toni, David Welch, Natalja Strelkowa, Andreas Ipsen, and Michael P.H Stumpf. Approximate bayesian computation scheme for parameter inference and model selection in dynamical systems. *Journal of The Royal Society Interface*, 6(31):187–202, 2009. doi: 10.1098/rsif.2008.0172. URL http://rsif.royalsocietypublishing.org/content/6/31/187.abstract. 9, 12, 14, 15

J.J. Tyson, K.C. Chen, and B. Novak. Sniffers, buzzers, toggles and blinkers: dynamics of regulatory and signaling pathways in the cell. *Current opinion in cell biology*, 15(2):221–231, 2003. 2