uc3m | Universidad Carlos III de Madrid

Escuela Técnica Superior
de Ingeniería Informática

Computational and Applied Mathematics

Year 2021-2022

Master Final Project

BAYESIAN INFERENCE FOR COMPLEX NETWORK: APPLICATION TO DIFFERENT STAGES OF GLIOBLASTOMA EVOLUTION

Author: Vanessa García López-Mingo

Tutor: María Pilar Guerrero Contreras

Agradecimientos

En primer lugar, quiero mostrar mi más sincero agradecimiento a mi tutora Mª Pilar Contreras por su entera disposición y ayuda durante estos meses. Gracias a su guía me he integrado en un área de estudio hasta el momento desconocida pero que ha conseguido despertar en mí el interés y las ganas de seguir formándome.

Agradezco también a todas las personas que de algún modo han formado parte durante toda mi etapa universitaria hasta llegar a esta última fase haciendo entrega de mi Trabajo Fin de Máster. En primer lugar, a mi familia, por su esfuerzo, apoyo y confianza, y también a mis profesores, tanto de Educación Secundaria como Universitaria, por guiarme y ayudarme a crecer tanto a nivel personal como académico.

Acknowledgements

First of all, I would specially like to thank my supervisor, Mª Pilar Guerrero Contreras, for being always there to help and at my complete disposal during these months. I feel thankful for her guide, continuous encouragement and support that have made me fit into a research area that I didn't know but that has awaken my interest and desire of continue learning.

Thanks also to all the people who have been part of all my academic years until the presentation of this Final Master Project. First, to my family for their sacrifice, support and trust, and also to all my teachers, for guiding and helping me to grow personally and professionally.

Abstract

Biological systems are characterised by huge complexity giving rise to mathematical models highly non-linear and multiparametric, what makes difficult the determination of the parameters' values as they are often difficult to measure experimentally and so, usually fitted for specif conditions, making the conclusions drawn difficult to generalise.

This Master Final Project has as main objective to present the application of an algorithm for parameter estimation of dynamical systems. This algorithm belongs to the Approximate Bayesian Computation (ABC) family, which comprises methods that evaluate posterior distributions without having to obtain the likelihoods, and follows a sequential Monte Carlo approach. The algorithm is applied to two deterministic biological systems: prey-predator Lotka Volterra and Succeptible, Infected, Recovered (SIR), and finally to the evolution of a Glioblastoma Multiforme (GBM). The mathematical model considers the interaction of tumour cells with oxygen concentration (pseudopalisade and necrotic core formation).

Key words: Sequential Monte Carlo, Bayesian Computation, Glioblastoma, Parameters, Biological systems.

Resumen

Los sistemas biológicos se caracetrizan por un alto grado de complejidad, dando lugar a models matemáticos no lineales y multiparamétricos, dificultando de este modo determinar los valores de los parámetros involucrados puesto que estos suelen ser difícles de medir experimentalmete y por lo tanto, la extracción de conclusiones resulta compleja.

Este Trabajo Fin de Máster tiene como principal objetivo abordar el estudio de un algoritmo para la estimación de parámetros en sistemas dinámicos. Este algoritmo pertenece a la familia de métodos de Aproximación Bayesiana Computacional (ABC), cuyo objetivo se basa en obtener distribuciones posteriores sin necesidad de calcular las funciones de verosimilitud, y sigue un enfoque Monte Carlo secuencial. El algoritmo se aplica a dos sistemas biológicos deterministas: Presa-depredador (Lotka Volterra) y Succeptible, Infected, Recovered (SIR). Finalmete se emplea para estudiar la evolución del Glioblastoma Multiforme (GBM). El modelo empleado considera la interacción de las células tumorakes con la conccentración de oxígeno disponible (pseudopalisade y formación del núcleo necrótico).

Palabras clave: Sequential Monte Carlo, Approximate Bayesian Compupation, Glioblas-

toma, Parámetros, Sistemas biológicos.

Contents

List of Figures

Chapter 1

Introduction

1.1 Context

Mathematical biology intends to predict the course or outcome of a biological event through the use of dynamical systems derived from biological data. Most of them are modelled by ordinary, delay or stochastic differential equations which are characterised by the large number of species as well as the complexity of interactions between the components of the system (**Mathbio**) what makes their analysis difficult. Indeed, for the vast majority there is a lack of reliable information about parameter values and frequently have several competing models for the structure of the underlying equations.

One of the main issues when modelling biological networks is to move from the theoretical biological knowledge of the system to the equations that describe its temporal behavior due to the incompleteness and sparsity of the biological experimental data, moreover the likelihood surfaces of large models are complex. The analysis of such dynamical systems therefore requires new, more realistic quantitative and predictive models (**articleABC**).

The statistical analysis of the information generated by medical follow-up is a very important challenge as the evolutionary course of a disease, generates information that should be processed in order to review and update its prognosis and treatment (**Alvares2017SequentialMC**). In this final project we present a parameter estimation algorithm for dynamical systems

using a Sequential Monte Carlo framework which follows an Approximate Bayesian computation (ABC) approach, i.e. a method that tries to evaluate posterior distributions without calculating likelihoods.

The algorithm is applied to two deterministic biological systems (Predator-prey Lotka Volterra and Susceptible-Infected-Recovered, or SIR model for infectious diseases) in order to obtain posterior parameter distributions and draw inferences from different data frameworks. After proving its efficiency, it is used to infer the parameters governing the equations of a deterministic glioblastoma evolution model (**coarsegraining**) and the results are compared to the real data offered by the Manuel Doblaré's laboratory in Zaragoza.

1.2 Goals

The main objective of this project is the application of a sequential inference method for a biological model that studies the different stages of glioblastoma evolution from a Bayesian perspective.

More concretely, the following specific objectives are stablished.

- Review of the most relevant Approximate Bayesian Computation (ABC) schemes, paying special attention to Sequential Monte Carlo approach (ABC-SMC).
- Program ABC-SMC algorithm following the approach given in (**articleABC**).
- Apply ABC-SMC algorithm to some well known deterministic biological models (Lotka Volterra and SIR).
- Develop a numerical method to solve the partial differential equations that govern the Glioblastoma evolution model (**coarsegraining**).
- Apply ABC-SMC to the glioblastoma evolution model in order to infer parameter that resemble the real data given by the laboratory.

1.3 Organization

The project is organised as follows.

Chapter 1 is a brief presentation of the context that determines the topic we are going to focus our attention on and the main objectives have been established.

Chapter 2 is intended to present a simple and short introduction to Bayes' rule and its application to the inference of parameters.

Chapter 3 is an introduction to the Approximate Bayesian Computation (ABC) method. After setting the basic ideas, three of the most used algorithms based on Monte Carlo simulations are presented: Rejection sampling, Markov Chain Monte Carlo and Sequential Monte Carlo. A special section will be dedicated to the last one (ABC-SMC), as it will be the one to use in the following chapters.

Chapter 4 illustrates the application of the ABC-SMC to two well known Biology dynamical systems: Deterministic prey/predator Lotka Volterra, and deterministic SIR model for infectious diseases.

Chapter 5 is a presentation of the glioblastoma model to solve. The different stages of the model as well as all the parameters and phases that determine the whole procedure are briefly explained at the sections of this chapter.

Chapter 6 shows the results obtained after applying the ABC-SMC developed in the previous chapters to the Glioblastoma model. Real data obtained from a laboratory is now used and different simulations are done in order to get conclusions about the accuracy and correctness of the method.

Chapter 7 contains the final conclusions of the project.

The Appendices section includes some of the methods not explicitly explained or included in the project text but that can be revised in order to have a better understanding of the procedures. The Newton Raphson, the 4^{th} order Runge Kutta methods, the main code of the ABC-SMC algorithm and the extraction technique of data used in the experiment can

be checked.

Chapter 2

Bayes background and the application to parameter inference

This chapter is a short introduction of the main tool that establishes the basis of all the project: Bayesian inference, an important area of signal processing, relevant to a wide range of real-world applications (**bayinf**). Therefore, the basic ideas under this concept will be set to have a better understanding of the following sections. Bayesian statistics is intended to generate the posterior distribution of the unknown parameters given a prior distribution for them and some data. It provides a much more complete picture of the uncertainty in the estimation of the unknown parameters of a model.

2.1 Bayesian inference

Bayesian inference is essentially about reallocating credibility across possibilities, i.e, it is based on the idea of updating belief with new evidence (**gELMAN**). The distribution of credibility initially reflects prior knowledge about the possibilities, which may not be accurate. Then, after the observation of new data, the credibility is re-allocated according to these observations. Possibilities that are consistent with the data gain more credibility, while possibilities that are not consistent with the data lose credibility. Therefore, it is

necessary to define the set of possibilities over which credibility is allocated (**book**).

Let's start this project with the simple definition of Bayes' rule which can be expressed through the following mathematical equation.

Definition 2.1.1 (Bayes' rule) The probability of the event A conditional on the event B is given by

$$P(A \mid B) = \frac{P(A\&B)}{P(B)} \tag{2.1}$$

This is, the probability of A given B is the probability that they happen together relative to the probability that B happens at all.

Bayes rule shows how marginal probabilities relate to conditional probabilities when taking data into account. All unknown quantities in the model are treated as random variables and the aim is to compute (or estimate) the joint posterior distribution. This is, the distribution of the parameters, conditional on the observed data.

There might be more possibilities than the ones considered in the beginning. However, we start taking those ones of our interest. After analysing the results of the first observation it is checked whether the possibilities considered offer a good explanation of the data, or by the contrary it is necessary to consider some of those possibilities that were not taken into account at first. This process is called "a posterior predictive check".

2.2 Applied to data and parameters

As it has already been established, the main application of Bayesian inference is based on the idea of updating prior belief with new evidence. In this sense Bayes' rule relates the conditional probability (or density) of a particular parameter value θ involved in the mathematical model given data $\mathscr{D}$, to the probability of $\mathscr{D}$ given θ by the rule, i.e. to determine the posterior distribution of the parameter given the data, $\pi(\theta|\mathscr{D})$: The aim is to obtain the posterior probability distribution over the model parameters, which is proportional to the product of a suitable prior distribution (which summarizes the user's

prior knowledge) and the likelihood (the information that is obtained from the data).

Bayesian data analysis can be summarised in the following steps (**book**):

1. Identify the data relevant to the research questions. What are the measurement scales of the data? Which data variables are to be predicted, and which data variables are supposed to act as predictors?

2. Define a descriptive model for the relevant data. The mathematical form and its parameters should be meaningful and appropriate to the theoretical purposes of the analysis.

3. Specify a prior distribution on the parameters. The prior must be adequate.

4. Use Bayesian inference to re-allocate credibility across parameter values. Interpret the posterior distribution with respect to theoretically meaningful issues (assuming that the model is a reasonable description of the data).

5. Check that the posterior predictions mimic the data with reasonable accuracy (i.e., conduct a "posterior predictive check"). If not, then consider a different descriptive model.

According to the steps described above, Bayesian inference depends on two main functions, the likelihood and the prior which are briefly described in the following lines:

1. **Likelihood function**

   $P(\mathscr{D}|\theta)$: The likelihood function of the parametric vector θ, which gives the probability density function for the data given the parameters. This is the first and most influential component that specifies the plausibility of the data.

2. **Prior**

   $P(\theta)$: The prior distribution of θ, a probability distribution that contains all the available prior belief about the parameter θ. This distribution is chosen before seeing any data or run any experiment.

From these two terms, the inferential process should naturally be summarised by the probability distribution of θ after observing the value of $\mathscr{D}$. This distribution, $\pi(\theta|\mathscr{D})$, is known as the posterior distribution of θ, the revised or updated probability of an event occurring after taking into consideration new information. To obtain it, we will use the previously mentioned Bayes' rule.

Prior, Likelihood and Posterior

For data $\mathscr{D}$ and variable θ (parameter), Bayes' rule tells us how to update our prior beliefs about θ in light of the data to a posterior belief:

$$\underbrace{\pi(\theta \mid \mathscr{D})}_{\text{posterior}} = \frac{\underbrace{P(\mathscr{D} \mid \theta)}_{\text{likelihood}}\underbrace{P(\theta)}_{\text{prior}}}{\underbrace{P(\mathscr{D})}_{\text{marginal likelihood}}} \tag{2.2}$$

The marginal likelihood can also be referred as evidence.

The prior $P(\theta)$ reflects the credibility of the parameter θ values without considering any data $\mathscr{D}$. The posterior $\pi(\theta \mid \mathscr{D})$, is the credibility of θ values when data $\mathscr{D}$ is taken into account. The likelihood, $P(\mathscr{D}|\theta)$, is the probability that the data could be generated by the model with parameter value θ. The marginal likelihood function of the data $P(\mathscr{D})$, is the overall probability of the data according to the model, determined by averaging across all possible parameter values weighted by the strength of belief in those parameter values.

$$P(\mathscr{D}) = \sum_{\Theta} P(\mathscr{D}|\theta^*)P(\theta^*) \tag{2.3}$$

where Θ refers to all possible values of the parameter while θ^* is a specific value. If we consider continuous variables instead of discrete, this expression becomes:

$$P(\mathscr{D}) = \int P(\mathscr{D}|\theta^*) \cdot P(\theta^*) d\theta \tag{2.4}$$

A model of data specifies the probability of particular data values given the model's struc-

ture and parameter values. The model also indicates the probability of the various parameter values. In other words, a model specifies:

$$P(\text{data values} \mid \text{parameters values}) \quad \text{along with the prior} \quad P(\text{parameters values})$$

And by using Bayes' rule we convert that to what we really want to know, which is how strongly we should believe in the various parameter values given the data (the posterior):

$$P(\text{parameters values} \mid \text{data values})$$

A general phenomenon in Bayesian inference:

> The posterior is a compromise between the prior distribution and the likelihood function. Sometimes this is loosely stated as a compromise between the prior and the data. The compromise favors the prior to the extent that the prior distribution is sharply peaked and the data are few. The compromise favors the likelihood function (i.e., the data) to the extent that the prior distribution is flat and the data are many.
>
> (**book**)

In general, the more available data, the more precision is obtained in the estimation of the parameter(s) of the model. Samples with larger sizes have as a result greater precision of estimation.

Then, in Bayesian inference a good choice on the prior that makes use of accurate previous data puts high credibility over a narrow range of parameter values and as a consequence, it takes a lot of novel contrary data to budge beliefs away from the prior. By the contrary, if the prior is weakly informed and distributes credibility over a wide range of parameter values, it takes relatively little data to shift the peak of the posterior distribution toward the data (although the posterior may be relatively wide and uncertain).

2.2.1 Data analysis

In order to analyse data we start by considering the information provided by the accessible data sets where it is located. There exist some candidate descriptions for the data, which are mathematical formulas that characterize the trends and spreads in the data. The numbers involved in these formulas are the **parameter values**, that determine the exact shape of mathematical forms. Parameters are like control knobs on mathematical devices that simulate data generation (**book**). Any change in the value of a parameter, changes a trend in the simulated data.

For example, in a normal distribution the two involved parameters are the mean and standard deviation:

- The mean controls the location of the distribution's central tendency (location parameter).
- The standard deviation controls the width or dispersion of the distribution (scale parameter).

The mathematical formula for the normal distribution converts the parameter values to a particular bell-like shape for the probabilities of data values.

The role of Bayesian inference is to compute the exact relative credibility of candidate parameter values, while also taking into account their prior distributions.

Chapter 3

Approximate Bayesian Computation

As it has already been stated in the previous section, Bayesian inference's goal is to get an accurate representation of the posterior distribution. One possibility to obtain it is by sampling a large number of representative points from the posterior; The larger the sample, the more accurate the approximation becomes. This is not a resampling of data (the data set is fixed), but instead, a sample of representative credible parameter values from the posterior distribution. What makes this method useful is that representative parameter values can be randomly sampled from complicated posterior distributions without solving the integral in Bayes' rule (Eq. 2.2), and by using only simple proposal distributions for which efficient random number generators already exist. In order to accept a proposed parameter value a decision must be made about the mathematical formulas for the likelihood function and prior distribution, which can be directly evaluated from their definitions.

Monte Carlo methods

Monte Carlo methods are non deterministic algorithms used to approximate intractable or non solvable mathematical problems through the simulation of random variables that relies on repeated random sampling and statistical analysis to compute the results.

In order to apply this method, it is necessary to produce random numbers drawn from any particular probability distribution $F(x)$, or, if it exists, from the probability density func-

tion. With a large enough number of samples, a sufficiently accurate numerical estimation of the distribution function can be obtained.

3.1 Approximate Bayesian Computation (ABC)

Nowadays, a lot of models in many different areas of science, such as biology, are able to describe nature pretty accurately, but, they are intractable to analytical treatment. However, through the use of computers, these models can be simulated and thereby replicate many complex phenomena such as the evolution of genomes or the dynamics of gene regulation (**Baycompscheme**) among many others. Such simulator-based models are often stochastic and have multiple parameters.

Given a set of values for the parameters involved in a model it is usually easy to generate data. Having a likelihood function, $f(x_0|\theta)$, and a prior distribution $\pi(\theta)$ on the parameter space Θ, the interest is set on the posterior distribution:

$$f(\theta \mid x_0) \propto f(x_0 \mid \theta)\,\pi(\theta),$$

which is the probability distribution of the parameters having observed the data, x_0. However, the inverse problem that deals with the identification of parameter values that would plausibly lead to data that are sufficiently similar to the observed data, is a difficult task and Bayesian inference provides a principled framework to solve it.

This process starts by considering prior beliefs about the model parameters which are expressed through the prior distribution $\pi(\theta)$ that is often specified by choosing a particular distribution among a set of well-known and tractable families of distributions (commonly, the Uniform distribution as it will be done in this project), such that both, the evaluation of prior probabilities and random generation of values of θ, are relatively straightforward (**articleABC**).

These beliefs are updated by observing data $d_{obs} \in \mathscr{D}$ through the likelihood function

$f(d_{obs}|\theta)$ of the model. Using Bayes' rule, the resulting posterior distribution would be:

$$\pi(\theta|d_{obs}) = \frac{f(d_{obs}|\theta) \cdot \pi(\theta)}{\int_{\Theta} f(d_{obs}|\theta) \cdot \pi(\theta) d\theta} \tag{3.1}$$

However, computing the likelihood function is mostly impossible for simulator based models due to the unobservable random quantities that are present in the model. Typically, the complexity of the model comes from the unavailability of the posterior distribution, $\pi(\theta|d_{obs})$, in closed form, so that numerical methods are needed to proceed with the inference. A common approach makes use of Monte Carlo integration to enumerate the necessary integrals, which has been shortly introduced at the beginning of the chapter.

Given this difficulty, Approximate Bayesian Computation (ABC) provides ways to do inference of posterior distributions in situations where the relationship between the data and the parameters does not lead to a computationally tractable likelihood function (**ABC**), (or this is too costly to evaluate) but where forward simulation of the data-generating process is possible. Therefore, the calculation of the likelihood is replaced by a comparison between the observed and simulated data in order to arrive at approximations to the true posterior distribution, which can not be analytically obtained. The basic ABC algorithm (**10.1214/aos/1176346785**, **Tavar1997InferringCT**) can be summarised in four steps.

All ABC-based methods approximate the likelihood function by comparing the outcomes of simulations with the real observed data.

First of all, the model parameters are taken from a prior distribution considered. These sampled parameters are used to fit a model in order to produce artificial data sets and compare how similar these are to the observed real data set by computing the distance between them. Parameter samples that produce artificial data sets which are close enough to the real data are collected as samples from the posterior distribution. This distance is determined by the tolerance (ε), a predefined hyper-parameter. Those which do not satisfy the distance condition are discarded.

Therefore, the general scheme of ABC inference is as follows:

- Sample a parameter vector (called particle) θ^* from a proposed prior distribution $\pi(\theta)$.
- Simulate a data set y from the model according to the generative model $f(y \mid \theta^*)$,
- Compare the simulated data set y with the experimental data x using a distance function Δ and a predetermined tolerance ε.
- If $\Delta(x, y) \leq \varepsilon$, accept the particle. The distance function measures the discrepancy between the two data sets.

This scheme is repeated until N particles are accepted and compose a sample from the posterior distribution

$$p_\varepsilon(\theta \mid x) \propto \int \mathbb{1}(\Delta(x, y) \leq \varepsilon) f(y \mid \theta) \pi(\theta) dy$$

which is an approximation of the posterior distribution $p(\theta \mid x)$.

As we have previously mentioned, Bayesian inference is about reallocation of beliefs. To have complete knowledge about a vector of parameters $\theta \in \Theta$ involved in a model M, we need to fit this model and observe the information contained in the posterior distribution.

The following subsections are aimed to present three of the most used ABC based methods to estimate parameters : Rejection Sampling, Markov Chain Monte Carlo (MCMC) and Sequential Monte Carlo (SMC), being the last, the one which will be developed and used in practice.

3.1.1 Rejection Sampling

The basic idea of rejection sampling is that although the probability density function f (target function) is intractable and therefore samples cannot easily be obtained from it, there exists another density function g (candidate function), such as a Normal distribution or a t-distribution, from which it is easier to sample. Therefore, the sample process is

done from g directly and then some of the samples are rejected in a strategic way to make the resulting "non-rejected" samples look like they came from f.

A set of parameter points is first sampled from the prior distribution $\pi(\theta)$. Given a sampled parameter point θ, a data set D^* is then simulated under the statistical model M specified by θ. If the generated data D^* is too different from the observed data D, the sampled parameter value is discarded. In precise terms, D^* is accepted with tolerance ε. The simplest form of ABC algorithm is known as rejection sampler and was introduced by **Pritchard1999**. Samples are not produced from the posterior $f(\theta|x)$ but from an approximation $p(\theta|\Delta(y,x))$. Posterior distribution of θ conditional on the event that the discrepancy of both data sets is lower than the tolerance ε chosen.

Rejection Sampling procedure scheme

1. Generate a candidate value $\theta^* \sim \pi(\theta)$ from the prior.
2. Generate a data set $x^* \sim f(x \mid \theta^*)$.
3. Accept θ^* if $d(x^*, x_0) \leq \varepsilon$ where x_0 is the target data.
4. If rejected, go to step 1.

Rejection is inefficient as it repeatedly samples candidate parameter vectors from the prior $\pi(\theta)$ which may be quite different to the posterior. Each accepted vector represents an independent draw from $f(\theta|\Delta(x,y) \leq \varepsilon)$, therefore, acceptance rates for the ABC-Rejection algorithm can be very low.

3.1.2 ABC Markov Chain Monte Carlo (ABC-MCMC)

Due to the limitations of rejection sampling, new methods to do ABC were introduced. (**doi:10.1073/pnas.0306899100**) proposed a method based on Markov Chain Monte Carlo (MCMC).

In order to introduce this concept, let's first give a simple definition of what a Markov Chain is.

Definition 3.1.1 (**Markov Chain**)
A stochastic process in which the distribution of a state only depends on the previous state. More formally:

Let $X_i, \{i = 0, 1, ..., n\}$ be a discrete-time stochastic process where each X_i takes values in some discrete set (state space) S. The stochastic process is said to be a Markov Chain if

$$P(X_{n+1} = i_{n+1} | X_n = i_n, X_{n-1} = i_{n-1}, ..., X_0 = i_0) =$$

$$P(X_{n+1} = i_{n+1} | X_n = i_n)$$

for all $n \geq 0$ and $i \in S$.

This means that, conditional on the present state of the system, future and past are independent (**article˙stoc**), i.e. the probability distribution of the process at time t, given all the previous values of the chain, is equal to the probability distribution if just the previous value is given. So, in the sequence of values that the chain takes, the distribution of the next value is determined just by the current value.

The collection of states that a Markov chain can visit is called the state space and the quantity that governs the probability that the chain moves from one state to another state is the transition kernel or transition matrix.

MCMC methods are based on the idea of simulating a discrete time Markov chain, $\{\theta_m\}_{0 \leq m}$, in parameter space Θ, whose stationary distribution is the posterior of interest $\pi(\theta|x)$. (**10.1093/biomet/82.4.711**; **Roberts˙2004**). Hence, by sampling from this Markov chain repeatedly, samples from the joint posterior distribution are obtained after a number of iterations.

There exist different inference methods based on MCMC, being the most representative one Metropolis-Hastings (**10.1093/biomet/57.1.97**, **1953JChPh21.1087M**), a popular

MCMC algorithm, in which transitions from state θ_m to a proposed new state θ^* occur with probability proportional to the relative posterior density between the two locations. The proposals are determined through sampling a proposal kernel distribution.

MCMC Metropolis Hasting procedure scheme

1. Initialize $\theta_i, i = 0$.
2. Propose θ^* according to a proposal distribution $q(\theta \mid \theta_i)$.
3. Simulate a dataset x^* from $f(x \mid \theta^*)$.
4. If $d(x_0, x^*) \leq \varepsilon$, go to step 5, otherwise set $\theta_{i+1} = \theta_i$ and go to step 6.
5. Set $\theta_{i+1} = \theta^*$ with probability

$$\alpha = \min\left(1, \frac{\pi(\theta^*)\, q(\theta_i \mid \theta^*)}{\pi(\theta_i)\, q(\theta^* \mid \theta_i)}\right)$$

   and $\theta_{i+1} = \theta_i$ with probability $1 - \alpha$. $\pi(\theta)$ is the prior.
6. Set $i = i + 1$, go to step 2.

To do so, at each time t, a particle θ is simulated from the previous particle θ_{t-1} according to a perturbation kernel $K(\cdot|\theta_{t-1})$ (or from the prior distribution for $t = 0$); the simulated data $y \sim f(\cdot|\theta)$ is compared with the experimental data; and θ_t is set to be equal to θ with a Metropolis Hasting acceptance rate if $\Delta(y, x) \leq \varepsilon$ and to θ_{t-1} otherwise. This algorithm is guaranteed to converge, however it is very difficult to assess when the Monte Carlo Chain reaches the stationary regime, and the chain may get trapped in local modes.

3.1.3 Sequential Monte Carlo

Sequential Monte Carlo (SMC) methods are considered as one of the most efficient methods for inference (**LopesTsay**) and have a wide range of applications in many areas such

as artificial intelligence, bioinformatics, computational science, machine learning, molecular chemistry, pharmacokinetic, phylogenetics, among others (**jouin:hal-01303393**). They can be referred as a set of simulation-based procedures which provide an appropriate approach to sequentially update complex posterior distributions.

The basic idea of SMC methods derives from the Sequential Importance Sampling (SIS) procedure, where the main goal is to sample from the desired target distribution (which may be hard or even impossible to sample from) indirectly through sampling from a proposal distribution. In SMC, the target distribution is approximated by using a set of simulated samples (particles) and their respective weights. Through a sequential procedure, the distribution is updated by incorporating the information provided by new data.

The methods obtained with the introduction of SIS and SMC samplers in the ABC framework (**doi:10.1073/pnas.0607208104**; **Baycompscheme**; **10635˙125047**) aim to sample sequentially from a series of distributions. These are obtained through the estimation of intermediate distributions $p_t(\theta \mid x)$ according to a decreasing sequence of thresholds $\{\varepsilon_t\}_{1 \leq t \leq N}$.

As this is the method we are going to focus our attention on, let's see a more detailed explanation in the following section.

3.2 ABC Sequential Monte Carlo (ABC-SMC)

Approximate Bayesian Computation through Sequential Monte Carlo (ABC-SMC) is an ABC approach where a sequence of distributions is eventually obtained by gradually decreasing the tolerance ε. In a Bayesian approach, the unknown parameter is considered random and assigned a suitable prior distribution. The posterior density of this parameter given the observations is to be characterised, however, deriving its analytical expression is neither possible nor necessary (**Andrieu**).

The ABC-SMC algorithm starts by sampling a finite number of parameters sets (particles) $\{\theta^1, \theta^2, ..., \theta^N\}$, from the prior distribution $\pi(\theta)$, which are later propagated through a

sequence of intermediate distributions (populations) $\pi(\theta|d(D_0,D^*)<\varepsilon_i), i=1,...,T-1$. Each intermediate distribution is again described by N parameters sets, $\theta_t^{(j)}$, $1 \leq j \leq N$ and their corresponding weights, $\omega_t^{(j)}$, which together we refer to as particles $(\theta_t^{(j)}, \omega_t^{(j)})$. Successive distributions are constructed by perturbing parameters through some kernel function, $\theta \sim K_t(\theta^*|\theta)$, generating simulated data, $f(y|\theta^*)$ and, upon acceptance, calculating the corresponding new weights. This procedure is repeated until it represents a sample from the target distribution, $\pi(\theta|d(D_0,D^*)<\varepsilon_T)$. The tolerances, are chosen such that $\varepsilon_i > ... > \varepsilon_T > 0$, $i \in \{0,...,T-1\}$ thus the distributions gradually evolve towards the target posterior.

Therefore, at first stage, a large value of the threshold, ε_1 must be chosen in order to accept many particles. Then, the ABC algorithm is used sequentially with thresholds ε_t but, instead of sampling the parameters from the prior distribution, they are sampled from the set of accepted particles at the previous stage and perturbed according to a suitable perturbation kernel. The performance of these methods depends on the number of particles, i.e, the larger, the better the representation of the target distribution is.

In the following image, obtained from (**s121216291**), a scheme of ABC-SMC is shown. See Figure 3.1.

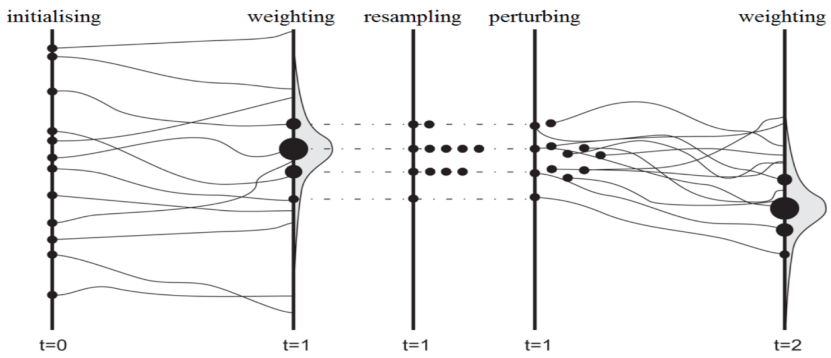


Figure 3.1: Scheme representation of ABC-SMC method (**s121216291**)

At the onset of the procedure, at time $t=0$ the particles are initially distributed over the domain of the values where the parameter θ is defined, with all its particles having the same weights (in all our applications we will use a Uniform prior distribution with weights set to 1). After that, at time $t=1$ the first observations have been obtained and

the sequential update procedure is initialised. First of all, new weights are assigned to the particles according to the information provided by the new observations. Then, the set of particles is resampled with probabilities proportional to their weights which are then reset to be equally likely. Finally, the particles are perturbed to avoid their accumulation in a few values. Once all this procedure has finished, the next time step $t = 2$ starts with the access to new observed data and the restarting of the update procedure as described above.

To sum up, the basic idea of SMC is to obtain for a given time t a large set of N weighted random samples $(\omega_t^{(j)}, \theta_t^{(j)}), j = \{1,...,N\}, \omega_t^{(j)} > 0, \sum_{j=1}^{N} \omega_t^{(j)} = 1$. SMC method works by constructing a series of intermediate distributions that start out from a suitably specified prior distribution and increasingly resemble the (unknown) approximate posterior distribution.

The following subsections 3.2.1 and 3.2.2 aim to offer a brief description of two of the tools that determine the performance of the algorithm to develop: The appropriate choice for the perturbation kernel and the distance function used.

3.2.1 Perturbation Kernel

There are two main choices that determine the behaviour of the SMC algorithm: The decreasing sequence of $\{\varepsilon_t\}_{1\leq t\leq T}$ and the perturbation kernels $\{K_t(\cdot \mid \cdot)\}_{1\leq t\leq T}$.

The first one lies in the difference between two successive thresholds ε_t and ε_{t+1}. If this is small, the distributions $p_t(\theta \mid x)$ and $p_{t+1}(\theta \mid x)$ are similar and a small number of simulations are required to generate N draws from the next intermediate distribution, $p_{t+1}(\theta \mid x)$, through sampling from the weighted population $\left\{\theta^{(j,t-1)}, \omega^{(j,t-1)}\right\}_{1\leq j\leq N}$. But a slowly decreasing sequence of thresholds $\{\varepsilon_t\}_{1\leq t\leq T}$ leads to a large number of iterations in order to obtain $\varepsilon_T = \varepsilon$.

The second one, which is the choice of the perturbation kernel $\{K_t(\cdot \mid \cdot)\}_{1\leq t\leq T}$ has considerable influence on the speed of convergence to the approximate posterior. A local

perturbation kernel hardly moves the particles and has the advantage to produce new particles which are accepted with high probability; on the other hand, a widely spread out or permissive perturbation kernel enables a full exploring of the parameter, but does so at the cost of achieving only low acceptance rates.

The computational efficiency of an ABC SMC algorithm is measured through the evolution of the acceptance rate of the algorithm, which is defined as the number of required simulation of data y to obtain N accepted particles at every time step t, this is, $\left\{\theta^{(j,t)}\right\}_{1\leq j\leq N}$, divided by the total number of simulated particles. The perturbation kernel used has great influence on this process.

A perturbation kernel with high acceptance rate requires a smaller number of expensive simulations and is therefore computationally more efficient than a kernel with smaller acceptance rate. The perturbation of a parameter θ consists of sampling a new particle according to a probability centred on θ. Different probability models may be used, with the most common being the uniform and the Gaussian distributions.

One of the most efficient choices for a kernel to perturb the particles is a multivariate normal distribution with a covariance matrix $\Sigma(t)$, which depends on the covariance of the previous population.

Let's introduce a concept necessary to understand the choice of this covariance matrix:

Definition 3.2.1 (Kullback-Leibler divergence) Non-symmetric measure of the difference between two probability distributions over the same variable x, $p(x)$ and $q(x)$. It is denoted by $D_{KL}(p(x),q(x))$, and measures the information lost when $q(x)$ is used to approximate $p(x)$. Given two probability distributions of a discrete random variable x (respectively continuos), such that both $p(x),q(x) > 0 \forall x \in X$ and sum up to 1, D_{KL} is given by:

- Discrete version:

$$D_{KL}(p(x)|q(x)) = \sum_{x\in X} p(x) \ln \frac{p(x)}{q(x)}$$

- Continuous version:

$$D_{KL}(p(x)|q(x)) = \int_{-\infty}^{\infty} p(x) \ln \frac{p(x)}{q(x)} dx$$

The covariance matrix $\Sigma(t)$ that minimizes the Kullback-Leibler divergence between the target posterior distribution, $\pi(\theta|x)$, and the proposal is the matrix given by $2 \cdot Cov(\{(\theta_t|x)\})$, since both $\theta_{(t-1)}$ and $\theta_{(t)}$ are sampled from the same distribution $p(\cdot|x)$. A complete analysis of this deduction can be found in **kernel**. Therefore, according to those calculations, an optimal choice of the covariance matrix $\Sigma(t)$ in the ABC-SMC for the multivariate normal perturbation kernel is:

$$\Sigma(t) = 2 \cdot Cov(\{(\theta)_{k,t}\}_{1 \leq k \leq N}) \tag{3.2}$$

3.2.2 Distance function

Definition 3.2.2 (SSE) The sum of squared errors (SSE) is defined as:

$$SSE = \sum_{i=1}^{n} (y_i - f(x_i))^2$$

This statistical value is then simply the error, added for all points, committed by considering the model instead of the data. The best model, in that regard, is then the model for which the error is the smallest.

3.2.3 ABC-SMC procedure scheme

After introducing all the elements and procedures required to develop an ABC-SMC algorithm, let's present the Pseudocode in 1 to understand how the scheme proceeds in practice.

The following sections offer different applications of this ABC-SMC scheme, which has been implemented from scratch in Python. The access to all the codes used in the project

Algorithm 1 ABC SMC algorithm

Require: : a list of decreasing thresholds ε_t, $1 \leq t \leq T$
Require: : target data x
  for $1 \leq t \leq T$ **do**
    for $1 \leq i \leq N$ **do**
      if $t = 1$ **then**
        sample θ^* from the prior $\pi(\theta)$
      else
        sample θ from the previous population $\{\theta^{(i,t-1)}, w^{(i,t-1)}\}_{1 \leq i \leq N}$
        perturb the particle using the kernel: $\theta^* \sim K_t(\cdot|\theta)$
      end if
      Apply method f to obtain data y using θ^*: $f(\cdot|\theta^*)$
      if $\Delta(x,y) \leq \varepsilon_t$ **then**
        Accept the particle $\theta^{(i,t)} \leftarrow \theta^*$
      end if
    end for
    Calculate the weights
    for $1 \leq i \leq N$ **do**
      if $t = 1$ **then**
        $w^{i,1} \leftarrow 1$
      else

$$\omega^{(i,t)} \leftarrow \frac{\pi\left(\theta^{(i,t)}\right)}{\sum_{j=1}^{n} \omega(j,t-1) K_t\left(\theta^{(i,t)} \mid \theta(j,t-1)\right)}$$

      end if
      Normalize the weights
    end for
  end for

are in /vgarcialopezm/ABC-SMC. The access to the general code used for all the models and an explanation of its procedure can be done through the following link: `https://github.com/vgarcialopezm/ABC-SMC/blob/main/ABC_SMC_ALGORITHM.ipynb`. It is also included in Appendices A.

Chapter 4

Applications to Dynamical Systems

In this chapter Approximate Bayesian Computation based on Sequential Monte Carlo (ABC-SMC), is going to be applied to two simple biological dynamical systems in order to illustrate its use on deterministic models and evaluate its performance. We aim to obtain posterior parameter distributions, and these distributions can then be further analysed in order to obtain additional insights into the system;

The algorithm is applied to the Lotka Volterra predator-prey model and to the SIR model. For these two examples we will use small simulated data sets in order to obtain our results. The simulated data points for the examples shown have been obtained by solving the models for some fixed parameters at chosen time points. The sizes of the input data sets have been chosen to reflect what can typically be expected in real-world data sets in ecology and molecular systems biology. In the following chapters, real experimental data sets will be considered.

4.1 Deterministic Lotka Volterra

The algorithm is going to be first applied to the well-known deterministic Lotka Volterra predator-prey model, which describes the interaction between two species that act respec-

tively as prey and predator. The model is defined by the following ODE system:

$$\begin{cases} \frac{dx}{dt} = f(x,y) = ax - bxy \\ \frac{dy}{dt} = g(x,y) = -cy + dxy \end{cases}, \quad x(0) = x_0, y(0) = y_0, \tag{4.1}$$

where x represents the density of the prey (usually considered, rabbits) and y represents the density of the predator (usually, foxes). The parameters involved in the model are (a,b,c,d) and are explained below. The dynamics underlying this model is the following:

- In the absence of predator, $\frac{dx}{dt} = ax$, the population of preys grows exponentially at a constant rate a.
- In the absence of preys, $\frac{dy}{dt} = -cy$, the population of foxes decreases until it gets extinct.
- The interactions between the two species is proportional to the product of the population densities, which contributes to the decrease of the prey population, b, and the increase of the predator population, d.

The parameter vector is given by $\theta = (a,b,c,d)$, positive real parameters describing the interaction of the two species where:

1. a: Growth rate of prey in the absence of interaction with predators.
2. b: Death rate per encounter of preys due to predation,
3. c: Natural death rate of foxes in the absence of food (rabbits),
4. d: The term dx denotes the net rate of growth of the predator population in response to the size of the prey population.

Eight points are going to be sampled for each specie and the parameters will be set to $(a,b) = (1,1)$. The ODE system will be solved by the Runge Kutta method of order 4.

4.1.1 ABC-SMC for Lotka Volterra model

In order to apply the ABC-SMC scheme developed to infer the parameters involved in the Lotka Volterra system we are going to consider the following conditions. The prior distributions for the parameters a, b, c and d are taken to be uniform, $a, b, c, d \sim U(0, 10)$, and the perturbation kernels for both parameters are multivariate normal, as we have seen before in section 3.2.1. Other kernels can also be used.

The function selected to measure the distance between the data $\{x_d[i], y_d[i])\}$, $i = 1, ..., 8$, and a simulated solution for proposed parameters, $\{(x[i], y[i])\}$, is the sum of squared errors (SSE); Some simulations are going to be obtained by changing the number of accepted particles at each population. To ensure the gradual transition between populations, we take 5 populations with tolerances $\varepsilon = (30.0, 16.0, 6.0, 5.0, 4.3)$.

First of all, let's solve the Lotka Volterra system with the four parameters set to 1 using Runge Kutta-Method of order 4 (code included in Appendices, B.2) taking 1000 time points in a domain $0 \leq t \leq 15$. The initial conditios for both populations are set to $x_0 = 0.5, y_0 = 1$. The result obtained is shown in figure 4.1.

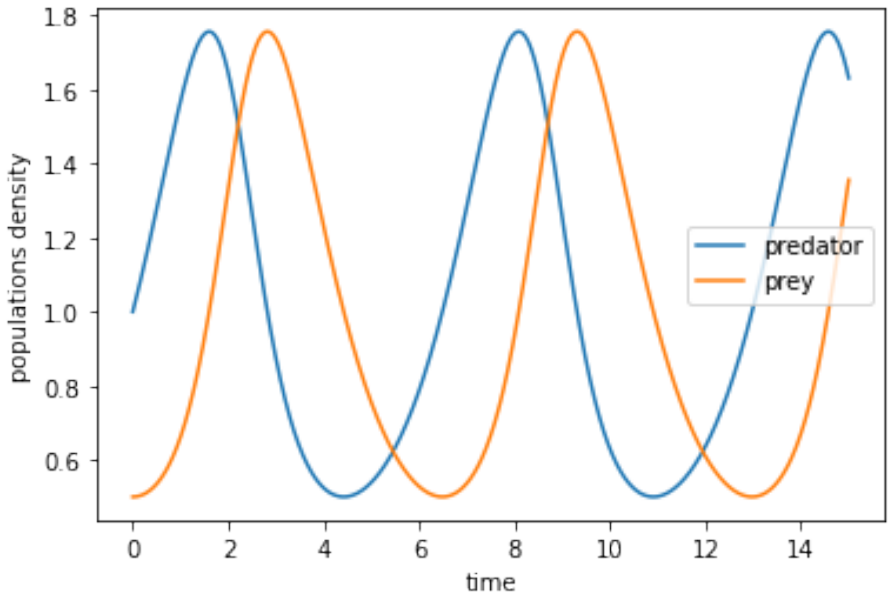


Figure 4.1: Lotka volterra system (Eq. 4.1) solved taking 1000 points. Orange solid line represents the evolution of prey population and blue solid line represents the evolution of predator on time.

In order to apply ABC-SMC, let's choose only eight points in the time domain:
$t = [1.1, 2.4, 3.9, 5.6, 7.5, 9.6, 11.9, 14.4]$.
After accepting 1000 particles at each population, the best result obtained in the last iteration is shown in figure 4.2.
The minimum distance obtained using ESS has been 0.004602423672076042. A really accurate result that shows the ABC-SMC scheme is working properly and converges to the target distribution.

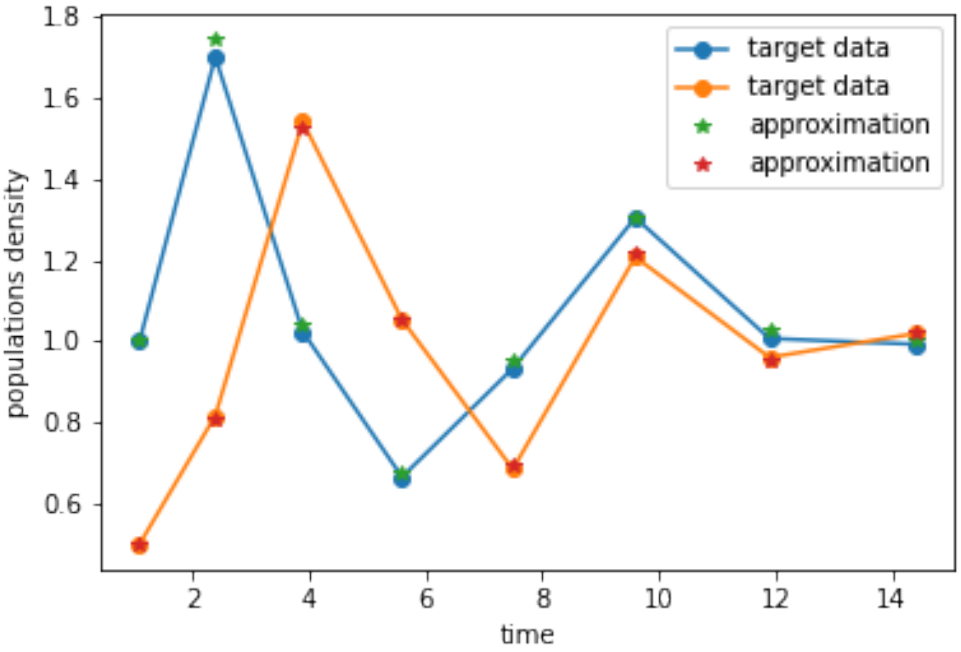
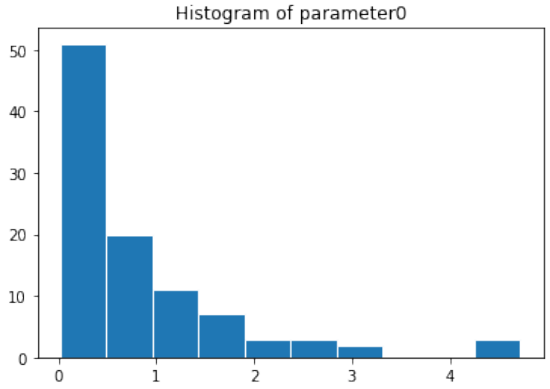


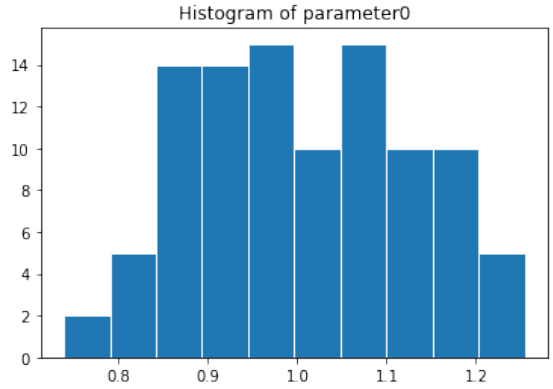
Figure 4.2: Orange and blue solid lines join the points obtained with parameters set to 1 (of both populations). The $*$ points show the best approximation obtained in the last population after applying ABC-SMC

Acceptance rate of parameters

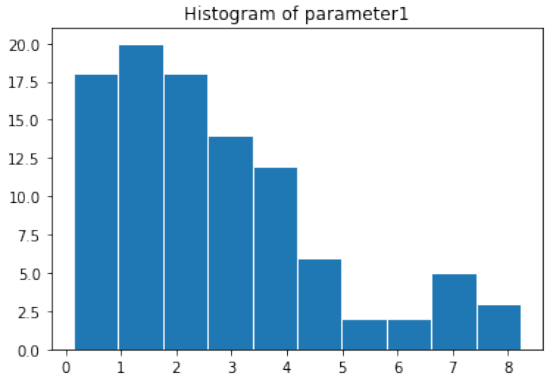
The following histograms show the accepted particles at the first and last population (the intermediate distributions are not shown here but are accessible in the code: `https://github.com/vgarcialopezm/ABC-SMC/blob/main/sequential_monte_carlo_datos_articulo1000_LV.ipynb`. It can be seen how they evolve to the target distribution.



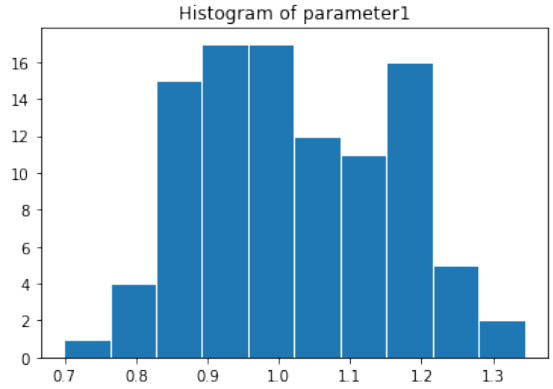
(a) Distribution of the parameter a first population.



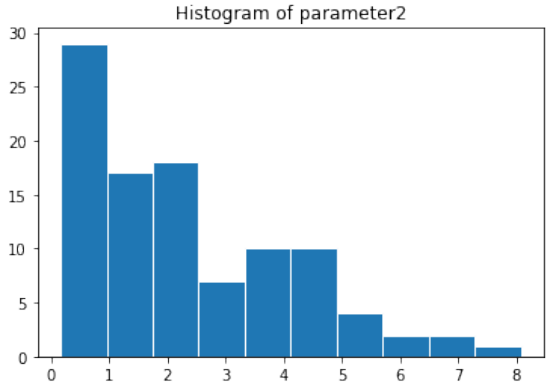
(b) Distribution of the parameter a last population.



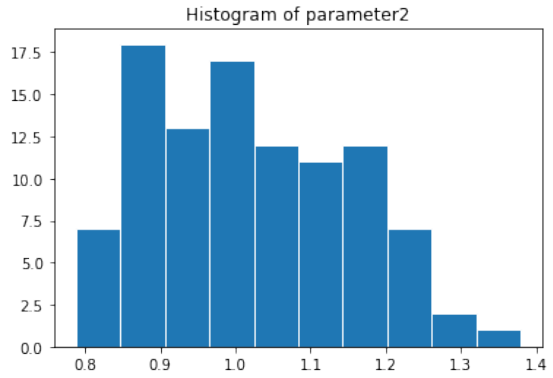
(c) Distribution of the parameter b first population.



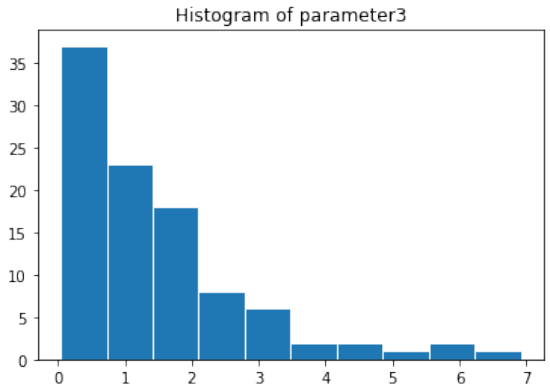
(d) Distribution of the parameter b last population.



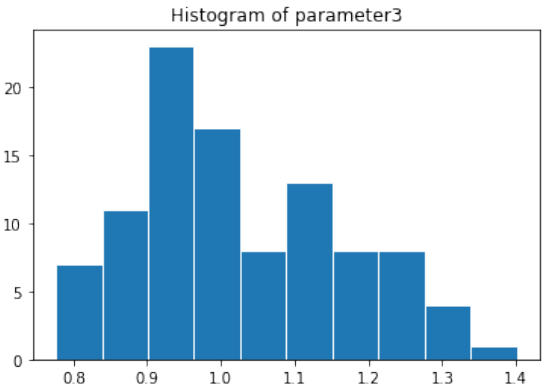
(e) Distribution of the parameter c first population.



(f) Distribution of the parameter c last population.



(a) Distribution of the parameter d first population.



(b) Distribution of the parameter d last population.

In the first step (the first column: figures 4.3a, 4.3c, 4.3e, 4.4a) , which corresponds to sample from the prior distributions, parameters with values over the whole domain are accepted. However, from the last population (Second column: figures 4.3b, 4.3d, 4.3f, 4.4b)we can see how the acceptance range has been reduced and the accepted parameters are near the ones which model the target data. Therefore all the parameters are well inferred.

4.2 Succeptible, Infected Recovered (SIR) model

We are going to use a model for the epidemiology of infectious diseases, the SIR model, which describes the spread of such disease within a population of N individuals that are classified into three compartments which vary as a function of time: susceptible (S), infected (I) and recovered (R) individuals (**Anderson1991**). According to the most basic mathematical version of this deterministic model, every individual can be infected only once and there is no time delay between the individual getting infected and their ability to infect other susceptible individuals.

These are the equations that model the SIR:

$$\begin{aligned}\frac{\mathrm{d}S}{\mathrm{d}t} &= -\frac{\beta SI}{N},\\ \frac{\mathrm{d}I}{\mathrm{d}t} &= \frac{\beta SI}{N} - \gamma I,\\ \frac{\mathrm{d}R}{\mathrm{d}t} &= \gamma I.\end{aligned}$$

- $S(t)$ are those susceptible but not yet infected with the disease;
- $I(t)$ is the number of infectious individuals;
- $R(t)$ are those individuals who have recovered from the disease and now have immunity to it.

The following assumptions are made:

1. The recovered individuals R, can not become infected again and do not transmit the disease to any susceptible individual.
2. The size of the population is considered to be fixed, i.e., $S(t)+I(t)+R(t)=N$
3. The probability of getting infected is the same for all the population at a rate proportional to the number of interactions between infected and susceptible $\frac{\beta SI}{N}$.
4. Infected individuals recover at a rate γ.

According to the above description, the changes in the population at each of these groups are governed by two parameters, β and γ. β is associated with transmission of the infection; It describes the effective contact rate of the disease: an infected individual comes into contact with βN other individuals per unit time (of which the fraction that are susceptible to contracting the disease is S/N). γ is the rate of recovery from infections, that is, $\frac{1}{\gamma}$ is the mean period of time during which an infected individual can pass it on.

4.2.1 ABC-SMC for deterministic SIR model

Experimental data consist of 10 equally spaced data points taken in a time domain $t = [0, 17.0]$ from each of the three groups (S, I and R). Initial conditions will be set to: $I_0 = 1, R_0 = 0, S_0 = N - I_0 - R_0$ and the total population (number of individuals) is $N = 100$ The parameters governing the model equations will be set to $(\beta, \gamma) = \{1.5, 0.5\}$. These premises will give the target data in order to later apply the ABC-SMC scheme. In this case, the python solver offered by scipy: integrate.odeint has been used to solve the system and the results are shown in figure 4.5.

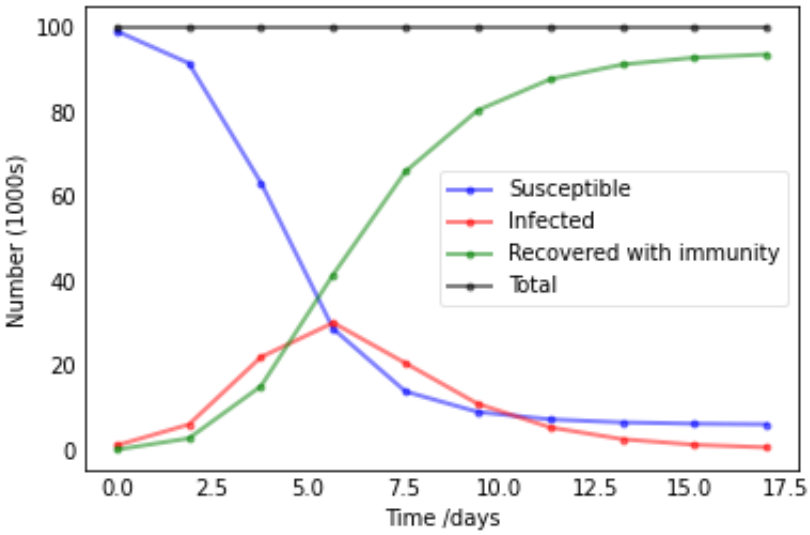
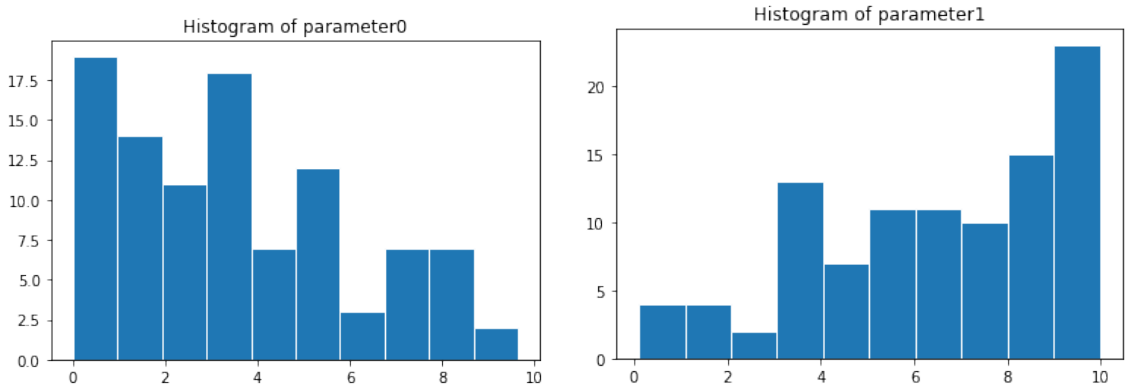


Figure 4.5: SIR model with parameters $(\beta, \gamma) = \{1.5, 0.5\}$

The prior distributions for the parameters β and γ are taken to be uniform, $\beta, \gamma \sim U(0, 10)$, and the perturbation kernels for both parameters are again multivariate normal.

The function selected to measure the distance between the data $\{x_d[i], y_d[i])\}$, $i = 1, ..., 10$, and a simulated solution for proposed parameters, $\{(x[i], y[i])\}$, is again the sum of squared errors (SSE); Some simulations are going to be obtained by changing the number of accepted particles at each population. To ensure the gradual transition between populations, we take 10 populations with tolerances $\varepsilon = (60, 50, 40, 20, 10, 5, 2, 1.5, 1, 0.5)$.

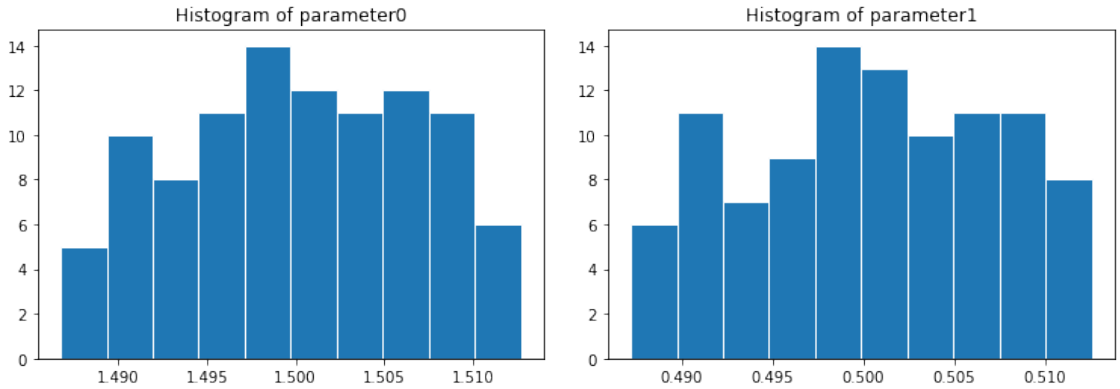
After accepting 100 particles at each population, the best result obtained in the last iteration with the minimum distance obtained using ESS is shown in figure 4.9. The following images 4.6 and 4.7 show respectively the particles values obtained for the first population (with tolerance $\varepsilon_0 = 60$) and for the last population (with tolerance $\varepsilon_9 = 0.5$).



(a) Distribution of parameter β.

(b) Distribution of parameter γ.

Figure 4.6: Distributions for the particles $\theta = (\beta, \gamma)$ at first time step. Population 0.



(a) Histogram of parameter β.

(b) Histogram of parameter γ.

Figure 4.7: Distributions for the particles $\theta = (\beta, \gamma)$ at last time step. Last population.

As we can see, the domain of parameter values of the first particles (figure 4.6 is quite

wide as they are sampled from the prior, a uniform distribution $\beta, \gamma \sim (0, 10)$. The tolerance has been set to $\varepsilon_0 = 60$), a large value such that a large number of particles are accepted at first.

Particles obtained at the last population (figure 4.7 are not sampled from the prior, but from the previous population, with its corresponding weights and after perturbing them with the kernel K. The tolerance has simultaneously decreased up to the last value $\varepsilon_9 = 0.5$.

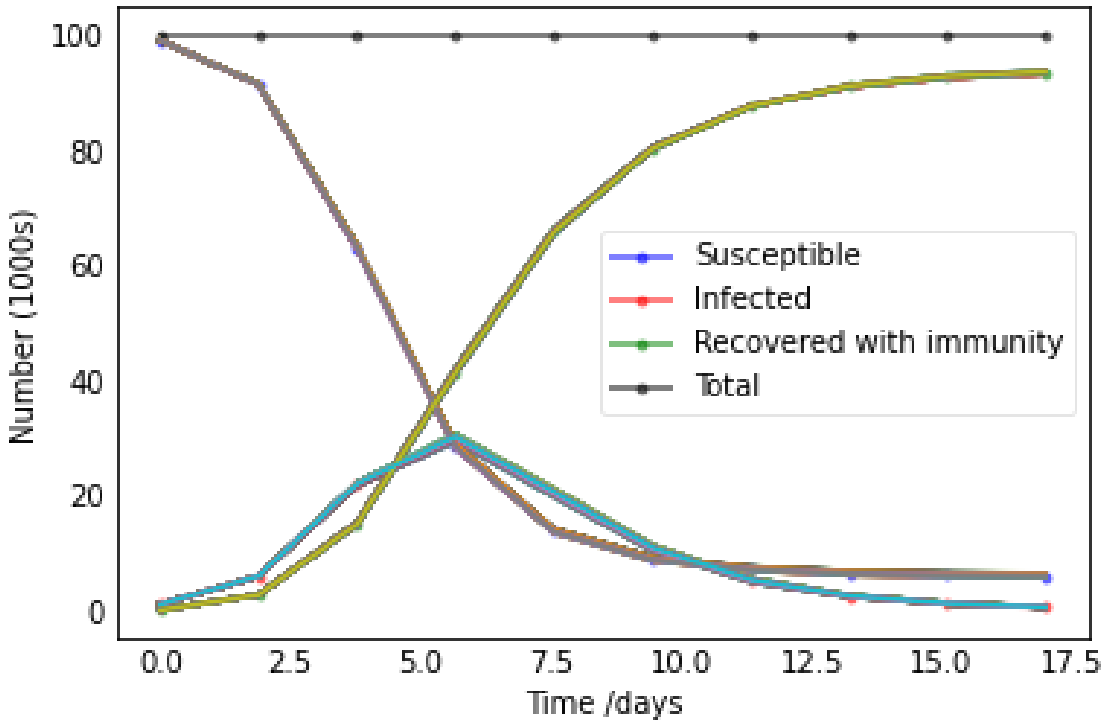


Figure 4.8: Approximations obtained with the 100 particles of the last population. It is observed all the particles accepted offer such a really accurate result that it is difficult to distinguish them.

Figure 4.9 is the best approximation obtained, with distance to the target data
$d = 0.024391843809122776$.
The particle value obtained is $(\beta, \gamma) = \{1.4996163693385132, 0.4998150410360696\}$.

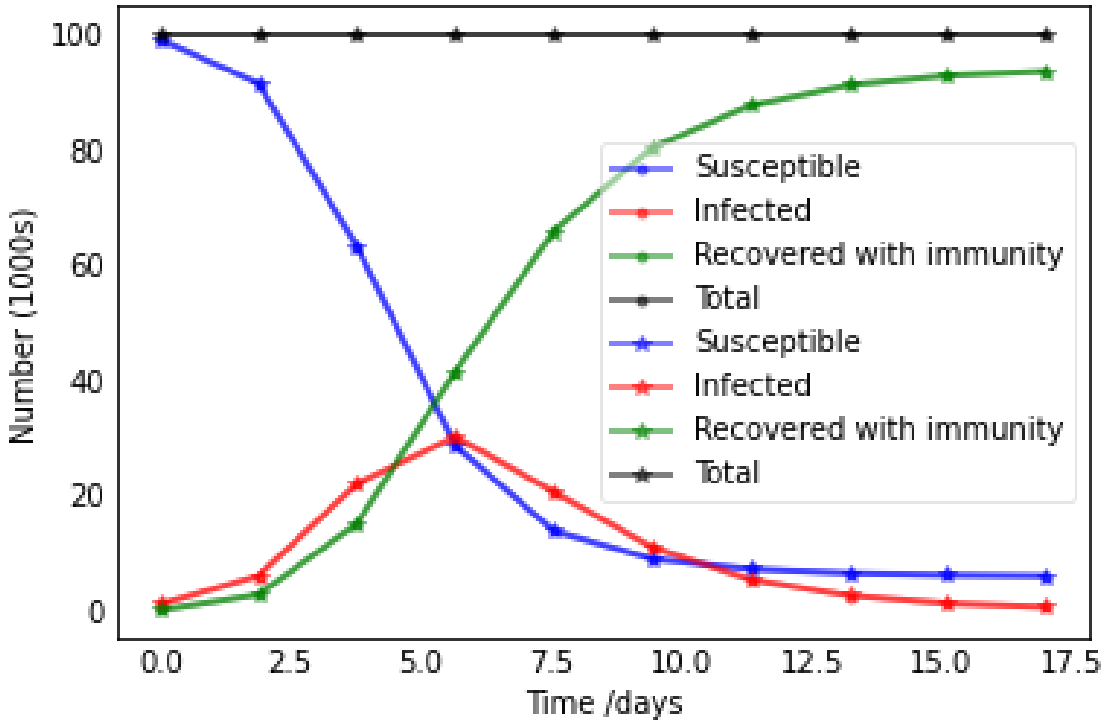


Figure 4.9: Best approximation obtained. The accuracy obtained is remarkable as the lines that join the approximation (* points) and the target data ('o' points) are indistinguishable

.

The whole procedure of the algorithm can be seen on the code `https://github.com/vgarcialopezm/ABC-SMC/blob/main/SIR_BUENO.ipynb`.

Chapter 5

Glioblastoma evolution model

Cancer is a complex disease caused by an uncontrollable growth and reproduction of a group of cells which provokes destruction of normal tissues and invasion of vital organs, leading to tumor formation. Depending on many different criteria such as degree of differentiation, rate of growth, local invasion and metastatic ability (**OZUGURLU20151504**), these tumor formations can be first classified as benign and malignant; being the first ones those which do not have the ability to invade.

Cancer research has attracted broad resources and scientists from many disciplines, and has produced some impressive advances in the treatment and understanding of this disease, which is considered as one of the most significant health challenges for the world. Mathematical modelling approaches have become increasingly abundant in recent years and have been proved to be useful for deriving a detailed understanding of mechanisms and processes in cancer. They have the ability to generate data, guide new research, suggest different treatment modalities and alter risk prognoses (**article˙cancermaths**), offering an understanding that would not otherwise be possible via traditional biological research techniques.

Mathematics give ”quantitative descriptions of cancer-driving mechanisms at multiple length and timescales which lead to new questions that can be addressed with novel experiments and mathematical models that integrate empirical evidence” (**article˙cancermaths**,

p.730). Such models then systematically evaluate assumptions, investigate alternative mechanisms and make predictions that can be experimentally validated. The power of mathematical modelling lies with its ability to reveal previously unknown or counter intuitive physical principles that might have been overlooked or can not be determined by intuition or observation.

5.1 Tumor characteristics

Tumor growth is a complex phenomenon involving many interrelated processes. It is tipically divided into three main stages: avascular growth, angiogenesis, and vascular growth (**angiog**).

Cancer starts with the mutation of one or a small number of normal cells, triggering an uncontrolled cell growth (neoplasm). In the earliest stage, the avascular, the tumor develops in the absence of blood supply as it has no direct connection to the vascular system. Therefore, it obtains all nutrients through diffusion from a nearby blood vessel. Its growth is limited by the availability of resources and nutrients, specially oxygen.

At this first stage of the tumor, three different cell layers can be distinguished. The most external one is composed of proliferating cells that are exposed to the highest concentrations of nutrients and in consequence, they drive the growth of the tumour. Below this thick layer there is a quiescent one consisting of cells that are in a static phase due to the reduced availability of nutrients. These cells no longer divide to generate new cells but rather their population only increases if a proliferating cell transitions to this static state. The more internal layer, which forms the centre of the tumour and often accounts for a majority of the tumor mass, is a necrotic core made up of dead tumour cells that result from the decrease of nutrients available below a survival threshold or through mechanisms such as apoptosis. The transition between the proliferative and quiescent layers is bidirectional while the transition to necrosis is unidirectional.

Due to the natural limitations associated to the avascular stage of growth, tumours utilise

mechanisms to advance to vascular tumours, giving rise to the second stage. Some of the cells of the avascular tumor mass produce and release substances called tumor angiogenic factors (TAFs) such as VEGF that trigger angiogenesis (**vascend**), the process of sprouting new blood vessels from existing ones in order to obtain a richer source of nutrients than that obtained through diffusion alone. At this third stage, the new vasculature provides potentially unlimited resources that allow the tumor grow beyond its limited avascular size, and eventually, to migrate and invade other tissues and organs of the body provoking what is known as metastasis. Thus, whereas in the avascular phase tumors are basically harmless, once they become vascular they are potentially fatal (**tumor**).

5.2 Multi-scale model

This section provides an overview of the structure of the model and a brief discussion of the different elements involved in its formulation. This multi-scale model, i.e. the inclusion of the physiological structure associated with the cell-cycle variables, intends to describe the growth of cellular populations in a spatially heterogeneous environment under the restriction of finite amount of available resources, in this case, oxygen $c(t,x)$ (**coarsegraining**).

The evolution of the concentration of oxygen $c(t,x)$ is modelled by:

$$\frac{\partial c}{\partial t} = \mathbf{D_c}\frac{\partial^2 c}{\partial x^2} - \mathbf{k_1} nc - k_2 c$$

The terms involved in this equation are the following:

- $\mathbf{D_c}$: The oxygen diffusion coefficient.
- k_1: Oxygen consumption rate.
- $k_2 c$: This term is introduced for two reasons: In its absence the oxygen concentration ahead of the front grows boundlessly, and regarding its biological interpre-

tation: it is associated with a native (passive) population which keeps the oxygen concentration finite, and against which the tumour modelled by our stochastic multi-scale system is growing (**coarsegraining**).

5.2.1 Cell cycle

The cell cycle is described as a series of consecutive events that takes place in a cell as it grows and divides. A cell spends most of its time in interphase. During this time it grows, replicates its chromosomes, and prepares for cell division. After leaving the interphase, the cell undergoes mitosis, and completes its division. Each of the resulting cells, known as daughter cells, enters its own interphase and begins a new round of the cell cycle.

The transitions through the different stages of this cycle occur when favourable conditions are detected, i.e, cells control their internal state and the cues they get from the exterior. This information then leads to the transition between the different stages of the cell cycle:(gap phase 1 (G1), DNA synthesis (S), gap phase 2 (G2), and mitosis (M)) ending with the cell division. The onsets of these stages that constitute the cell cycle mentioned above are associated with the activation of cell cycle regulators known as cyclin dependent kinases (CDKs) (**cdk**). Our attention will be put on the $G1/S$ transition, which is associated to the activation of CycB.

5.2.2 $G1/S$ transition age ($a_{G1/S}$)

The entrance of the cell in the S phase is strongly dependent on the concentration of oxygen (the more abundant oxygen is, the faster the cell-cycle goes through the G1/S transition (**coarsegraining**) and the relative concentration of enzymes (p_6, p_3) regulating the activity of SCF, an inhibitor of CycB. This first passage time for the cell to enter the S phase is one of the fundamental quantities in the multi-scale model and is referred as the age at the $G1/S$ transition $a_{G1/S}$ since this time starts counting when the cell is born. It determines the age-dependent birth rate and exhibits remarkable regularities (with respect

to oxygen concentration dependence and the cell-cycle parameters p_6, p_3) that simplify the multi-scale methodology.

After some analysis done in **CRUZ2016161**, it is seen there exists a power-law dependence of $a_{(G1/S)}$ on $\frac{c}{c_{cr}(p_6/p_3)} - 1$, and a good scaling approximation for its value is given by:

$$a_{G1/S}\left(c, \frac{p_6}{p_3}\right) \simeq \begin{cases} a_+\left(\frac{p_6}{p_3}\right) e^{-c/c_0} \text{ if } \frac{p_6}{p_3} > r_{cr} \\ a_-\left(\frac{c}{c_{cr}(p_6/p_3)} - 1\right)^{-\beta} \text{ if } \frac{p_6}{p_3} < r_{cr} \end{cases}$$

Here, $c_0, a_\pm$, and β are constants, p_6 and p_3 are respectively the SCF-inactivating and SCF-activating enzymes momenta coordinates, r_{cr} is critical value of the ratio p_6/p_3 above which there is no transition to quiescence, this term is fixed $r_{cr} = 1.004$.

The critical concentration of oxygen for quiescence. $c_{cr}\left(\frac{p_6}{p_3}\right)$ can be analytically calculated by solving the following equation:

$$c_{cr}\left(\frac{p_6}{p_3}\right) = 1 - \frac{1}{\beta_1}\log\left(\frac{1}{a_3 H_0}\left(a_1 + \frac{a_2 d_2}{d_1 [e2f]_t}\left(1 - \frac{1}{1 - a_0\left(\frac{p_3}{p_6}\right)^2}\right)\right)\right)$$

The parameters $a_1, a_2, a_3, d_1, d_2, \beta_1, [e2f]_t$, and H_0 are kinetic parameters of the mean-field model defined in (**coarsegraining**) and are included in table 5.1.

Parameters	
a_0	0.45185
a_1	0.51
a_2	1
$a_3 H_0$	0.0085
d_1	0.2
d_2	0.1
$[e2f]_t$	1
β_1	2.5

Table 5.1: Parameters extracted from (**coarsegraining**)

In our analysis, we will consider the ratio p_6/p_3 as a constant set to 1. Under this condition, the value of $c_{cr}\left(\frac{p_6}{p_3}\right)$ has been calculated, and the result is $c_{cr}(1) = 0.02268930984435391$.

Therefore, the calculation of $a_{G1/S}$ is given by $a_-\left(\frac{c}{c_{cr}(1)} - 1\right)^{-\beta}$.

The evolution of the population of cells $n(t,x)$ is given by:

$$\frac{\partial n}{\partial t} = \mathbf{D_n}\frac{\partial^2 n}{\partial x^2} - \lambda(c)n.$$

The terms involved in this equation are the following:

- D_n: The population diffusion coefficient.
- $\lambda(c)$, which solves the equation:

$$\frac{\tau_p}{2} = \frac{e^{-(\lambda+\nu)a_{G_1\setminus S}(c)}}{\lambda+\nu+\tau_p^{-1}}$$

  This equation of $\lambda(c)$, which measures the birth rate of the cells depending on the available quantity of oxygen is going to be solved by using Newton Raphson method (code included in Appendices, B.1.1). In order to avoid computational problems, logarithms will be taken, obtaining:

$$f(\lambda) = log(\tau_p) - log(2) + (\lambda+\nu)a_{G_1\setminus S}(c) - log(\lambda+\nu+\tau_p^{-1})$$

  Let's analyze this last equation: τ_p is the average waiting time to division after the $G1/S$ transition, then, the inverse, τ_p^{-1} is considered a constant rate which determines when cell division occur, provided cells have undergo the $G1/S$ transition. ν is the death rate.

5.3 Solving the system. Numerical Method

The method of finite difference is studied for the solution of the reaction-diffusion equation. This method consists in the approximate solution of PDEs by using a discretization on a mesh. In order to solve the system of equations of the tumor growth model, Implicit Euler method will be applied.

Mathematical Model of tumor growth

$$\frac{\partial n}{\partial t} = \mathbf{D_n}\frac{\partial^2 n}{\partial x^2} + \lambda(c)n.$$

$$\frac{\partial c}{\partial t} = \mathbf{D_c}\frac{\partial^2 c}{\partial x^2} - \mathbf{k_1}nc - \mathbf{k_2}c$$

With Neumann boundary conditions:

$$\frac{\partial n}{\partial x|_{x_0}} = 0, \qquad \frac{\partial n}{\partial x|_{x_L}} = 0, \qquad \frac{\partial c}{\partial x|_{x_0}} = 0, \qquad \frac{\partial c}{\partial x|_{x_L}} = \beta_f$$

The simulations will be done in one dimension $x = [0, L]$, and time $t = [0, T]$.

The set of finite difference equations will be obtained on a grid with discrete points (x_i, t_j):

* $x_i = i\Delta x \qquad 0 \leq i \leq L.$

* $t_j = j\Delta t \qquad 0 \leq j \leq T.$

Where Δx and Δt are the spatial and temporal discretization steps respectively. So, for example, for the population of cells, we denote by $n_{i,j} \sim n(x_i, t_j)$ the numerical approximation of n at discrete point x_i in the time t_j.

5.3.1 Implicit Euler

Implicit Euler is a popular implicit method to solve parabolic equations since it is second order accurate and unconditionally stable. The finite differences approximations used in this method for the cell population $n(x,t)$ (the same are deducted for the oxygen concentration $c(x,t)$) will be:

$$\left(\frac{\partial n}{\partial t}\right)_{i,j} = \frac{n_{i,j+1} - n_{i,j}}{\Delta t}$$

$$\left(\frac{\partial^2 n}{\partial x^2}\right)_{i,j+1} = \frac{n_{i+1,j+1} - 2n_{i,j+1} + n_{i-1,j+1}}{(\Delta x)^2}$$

Introducing them in our model, the system to solve is:

$$\frac{n_{i,j+1}-n_{i,j}}{\Delta t} = D_n \left[\frac{n_{i+1,j+1}-2n_{i,j+1}+n_{i-1,j+1}}{(\Delta x)^2}\right] + \lambda(c_{i,j})\cdot n_{i,j}$$

$$\frac{c_{i,j+1}-c_{i,j}}{\Delta t} = D_c \left[\frac{c_{i+1,j+1}-2c_{i,j+1}+c_{i-1,j+1}}{(\Delta x)^2}\right] - k_1 c_{i,j}\cdot n_{i,j} - k_2 c_{i,j}$$

Rearranging:

$$n_{i,j+1}-n_{i,j} = \frac{D_n \Delta t}{(\Delta x)^2}\left[n_{i+1,j+1}-2n_{i,j+1}+n_{i-1,j+1}\right] + \Delta t\cdot\lambda(c_{i,j})\cdot n_{i,j}$$

$$c_{i,j+1}-c_{i,j} = \frac{D_c \Delta t}{(\Delta x)^2}\left[c_{i+1,j+1}-2c_{i,j+1}+c_{i-1,j+1}\right] + \Delta t\cdot(-k_1 c_{i,j}\cdot n_{i,j} - k_2 c_{i,j})$$

Considering $\sigma_n = \frac{D_n\Delta t}{(\Delta x)^2}$, $\sigma_c = \frac{D_c\Delta t}{(\Delta x)^2}$, multiplying the expressions by 2 and separating the j terms on one side and $(j+1)$ on the other side of the equations we have for $i = 1,...,L-1$:

$$-\sigma_n n_{i-1,j+1} + (1+2\sigma_n)n_{i,j+1} - \sigma_n n_{i+1,j+1} = n_{i,j} + \Delta t\lambda(c_{i,j})n_{i,j}$$

$$-\sigma_c c_{i-1,j+1} + (1+2\sigma_c)c_{i,j+1} - \sigma_c c_{i+1,j+1} = c_{i,j} + \Delta t(-k_1 c_{i,j} n_{i,j} - k_2 c_{i,j})$$

As there are boundary conditions for both equations, there will be $L+1$ unknowns and $x_0 = 0$ and $x_f = L$ will be obtained by using ghost points x_{-1} and x_{L+1} using centered differences:

$$0 = \frac{n_{L+1,j}-n_{L-1,j}}{2\Delta x} \quad , \quad 0 = \frac{n_{1,j}-n_{-1,j}}{2\Delta x}.$$

$n_{L+1,j} = n_{L-1,j}$ and $n_{-1,j} = n_{1,j}$

$$0 = \frac{c_{1,j}-c_{-1,j}}{2\Delta x} \quad , \quad \beta_f = \frac{c_{L+1,j}-c_{L-1,j}}{2\Delta x}.$$

$c_{L+1,j} = 2\Delta x\beta_f + c_{L-1,j}$ and $c_{-1,j} = c_{1,j}$

Obtaining the equations:

- $n_{x_0} = 0$:
  $(1+2\sigma_n)n_{0,j+1} - 2\sigma_n n_{1,j+1} = n_{0,j} + \Delta t\lambda(c_{0,j})n_{0,j}$,

- $n_{x_f} = L$:
  $-2\sigma_n n_{L-1,j+1} + (1+2\sigma_n)n_{L,j+1} = n_{L,j} + \Delta t\lambda(c_{L,j})n_{L,j}$,

- $c_{x_0} = 0$:
  $(1+2\sigma_c)c_{0,j+1} - 2\sigma_c c_{1,j+1} = c_{0,j} + \Delta t(-k_1 c_{0,j} n_{0,j} - k_2 c_{0,j})$

- $c_{x_f} = L$:
  $-2\sigma_c c_{L-1,j+1} + (1+2\sigma_c)c_{L,j+1} - \sigma_c(2\Delta x\beta_f) = c_{L,j} + \Delta t(-k_1 c_{L,j} \cdot n_{L,j} - k_2 c_{L,j})$

5.3.1.1 Matrix vector form of the Crank-Nicholson method

The Implicit Euler method can be written in a matrix vector form as follows.

- System to solve the equation that governs the population of cells $n(x,t)$:

$$\begin{bmatrix} 1+2\sigma_n & -2\sigma_n & 0 & \cdot & \cdot & \cdot & \cdot \\ -\sigma_n & 1+2\sigma_n & -\sigma_n & & & & \cdot \\ 0 & & \cdot & & & & \cdot \\ \cdot & & & \cdot & & & \cdot \\ \cdot & & & & \cdot & & \cdot \\ \cdot & & & & -\sigma_n & 1+2\sigma_n & -\sigma_n \\ \cdot & & & & 0 & -2\sigma_n & 1+2\sigma_n \end{bmatrix} \begin{bmatrix} n_{0,j+1} \\ \cdot \\ \cdot \\ \cdot \\ \cdot \\ \cdot \\ n_{L,j+1} \end{bmatrix} =$$

$$\begin{bmatrix} n_{0,j} \\ n_{1,j} \\ . \\ . \\ . \\ . \\ n_{L,j} \end{bmatrix} + \Delta t \begin{bmatrix} \lambda(c_{0,j}) n_{0,j} \\ . \\ . \\ . \\ . \\ . \\ \lambda(c_{L,j}) n_{L,j} \end{bmatrix}$$

- System to solve the equation that governs the oxygen concentration $c(x,t)$:

$$\begin{bmatrix} 1+2\sigma_c & -2\sigma_c & 0 & . & . & . & . \\ -\sigma_c & 1+2\sigma_c & -\sigma_c & & & & . \\ 0 & & . & & & & . \\ . & & & . & & & . \\ . & & & & . & & . \\ . & & & & -\sigma_c & 1+2\sigma_c & -\sigma_c \\ . & & & & 0 & -2\sigma_c & 1+2\sigma_c \end{bmatrix} \begin{bmatrix} c_{0,j+1} \\ . \\ . \\ . \\ . \\ . \\ c_{L,j+1} \end{bmatrix} = \begin{bmatrix} c_{0,j} \\ c_{1,j} \\ . \\ . \\ . \\ . \\ c_{L,j} \end{bmatrix} +$$

$$2\Delta t \left(-k1 \begin{bmatrix} c_{0,j} \cdot n_{0,j} \\ c_{1,j} \cdot n_{1,j} \\ . \\ . \\ . \\ . \\ c_{L,j} \cdot n_{L,j} \end{bmatrix} - k2 \begin{bmatrix} c_{0,j} \\ c_{1,j} \\ . \\ . \\ . \\ . \\ c_{L,j} \end{bmatrix} \right) + \begin{bmatrix} 0 \\ 0 \\ . \\ . \\ . \\ . \\ 2\Delta x \sigma_c \beta_f \end{bmatrix}$$

5.3.2 Initial conditions

The mathematical model explained above is going to be used to analyse the most aggressive and lethal primary brain tumour: the evolution of hypoxia-driven migratory structures in Glioblastoma Multiforme (GBM).

Initial conditions for the cells population $n(x,0)$ are taken from the experiment carried out in **doblare**, using the same extraction technique for the data (a resume can be seen in Appendices, C). We will consider as the initial time, the data corresponding to day 3 of the laboratory, which contains 191 equally spaced points, see Figure 5.1.

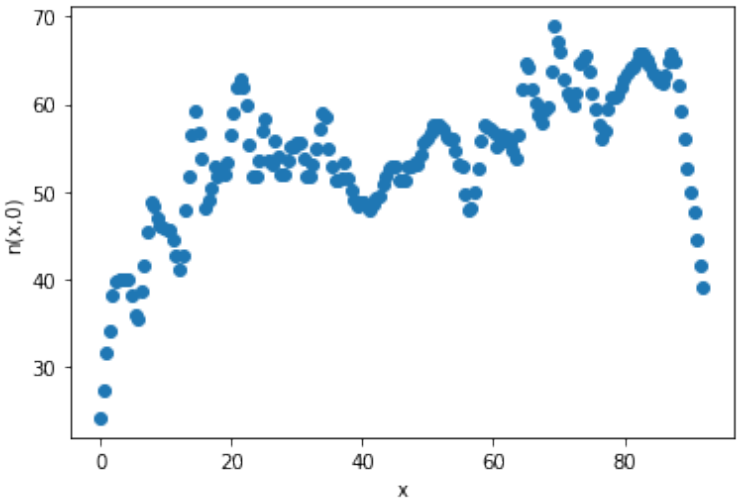


Figure 5.1: Initial condition of the cells population taken from day 3 of Doblaré data set (**doblare**).

The initial condition for the oxygen concentration $c(x,0)$ will be calculated as the steady state of the equation, which gives as a result:

$$c(x,0) = \frac{D_c \beta_f}{L(k_1 \overline{U} + k_2)}$$

where $\overline{U}$ is the mean value of cells over the whole space.

The main goal of the project is to infer some of the parameters involved in order to approximate the results to the data corresponding to day 6 of the laboratory.

Chapter 6

ABC-SMC for the glioma multi-scale model

Before applying the scheme to infer parameters according to data given on day 6, some simulations are done by using fixed values for all parameters in order to prove the Bayesian method works when it is applied to a system of partial differential equations, as the two applications described in the previous chapter are governed by ODE. Moreover, the number of parameters to infer and the number of points considered were smaller than in the model to solve now.

The values of the parameters involved in the model in this first simulations have been obtained from the literature and are given in table 6.1.

Parameters	
k_1	1.45773318
k_2	$9.42e-02$
β_f	40.012
β	0.5
a_-	8250
ν	$4.16667e-06$
D_n	$60e-03$
D_c	60

Table 6.1: Parameters extracted from **coarsegraining**

The initial condition for the oxygen concentration $c(x,0)$ is set to a constant value $c_0 = 0.1$

along the whole space domain, and the initial condition for the cells population $n(x,0)$ is the one corresponding to day 3 of the data obtained through the experiment in **doblare** (shown in the previous section, see Figure 5.1) which consists of 191 equally spaced points in a range $(0,916)$. By applying the Implicit Euler method (using the matrix version described) to our model equations using these parameter values the result obtained is shown in the following graphics, see Figure 6.1:

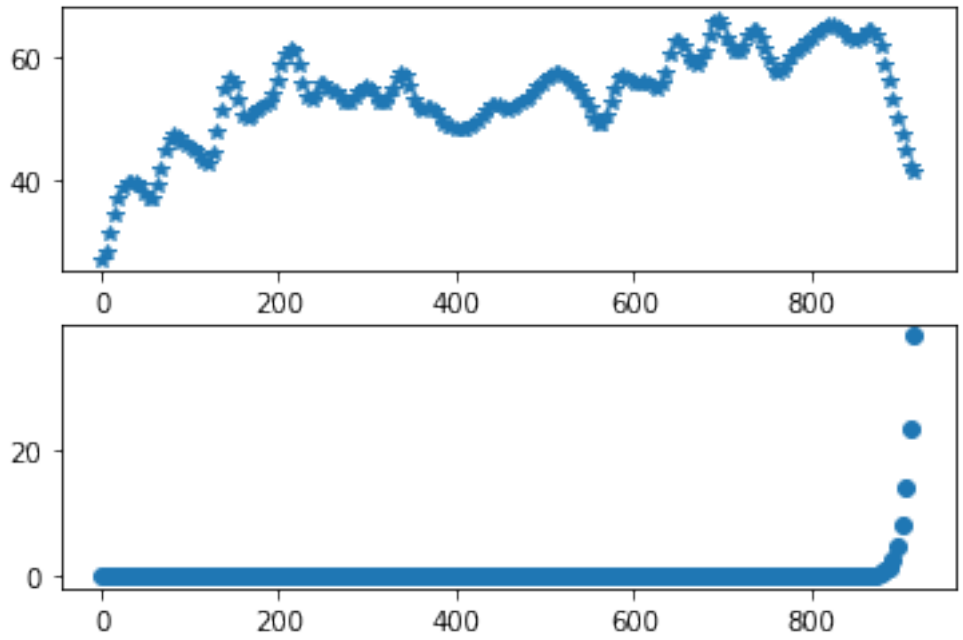


Figure 6.1: The graphic above represents the population of cells $n(x,T)$, the graphic below represents the oxygen concentration $c(x,T)$, at final time T

As a first experiment only three of the parameters are considered as unknowns to infer, both diffusion coefficients: D_n, D_c and the term involved in the calculation of $a_{G_1/S}$: a_-.

The prior distributions for the parameters are taken to be uniform:

* $D_n \sim U(0,1)$,

* $D_c \sim U(50,70)$,

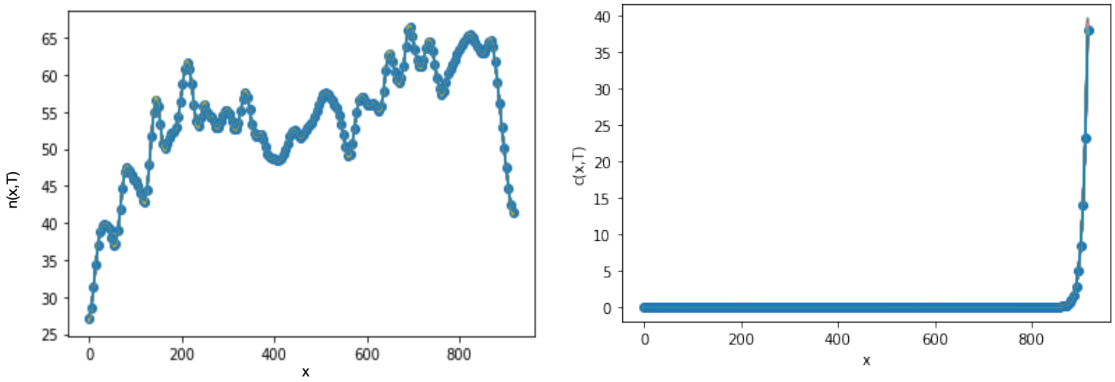
* $a_- \sim U(8000,9000)$,

and the perturbation kernels for all the parameters are again multivariate normal. The function selected to measure the distance between the data $\{x_d[i], y_d[i])\}$, $i = 1,...,191$, and a simulated solution for proposed parameters, $\{(x[i], y[i])\}$, is the sum of squared

errors (SSE).

The number of accepted particles at each population is set to 20, which is quite small compared to the applications done in chapter 4. To ensure the gradual transition between populations, we take 11 populations with tolerances:

$\varepsilon = [3000.0, 2000.0, 1000.0, 500.0, 300.0, 200.0, 100.0, 50.0, 20.0, 10.0, 5.0]$.



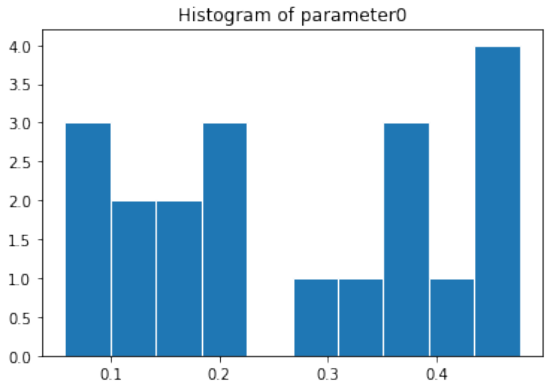
(a) Approximations at last population for $n(x,t)$. The solid line represents the approximation and the points are the target data.

(b) Approximations at last population for $c(x,t)$. The solid line represents the approximation and the points are the target data.

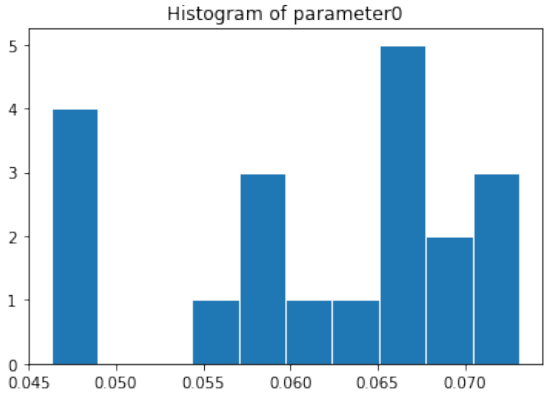
Figure 6.2: Estimated population of cells and oxygen at final time T

Acceptance rate of parameters

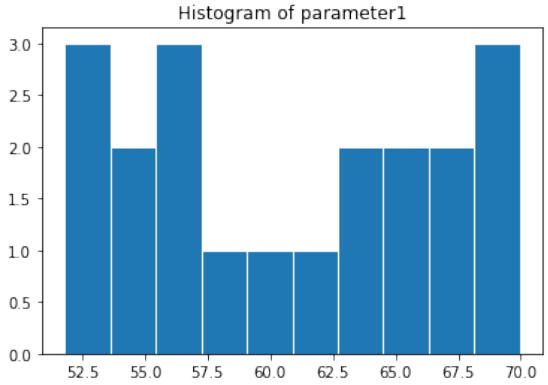
These are the histograms corresponding to first and last population of all the parameters inferred. In the first step (First column: Figures 6.3a, 6.4a, 6.4c) which corresponds to sample from the prior distributions, parameters with values over the whole domain are accepted. The last population (Second column: Figures 6.10a,6.4b, 6.4d) shows how the acceptance range has been reduced and the accepted parameters are gathered near the values used to obtain the target data, giving rise to accurate results in the inference process:



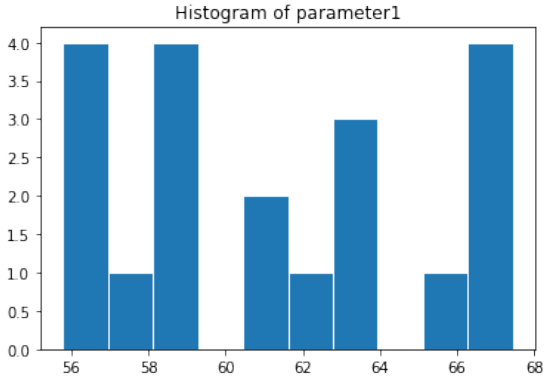
(a) Distribution of parameter D_n first population



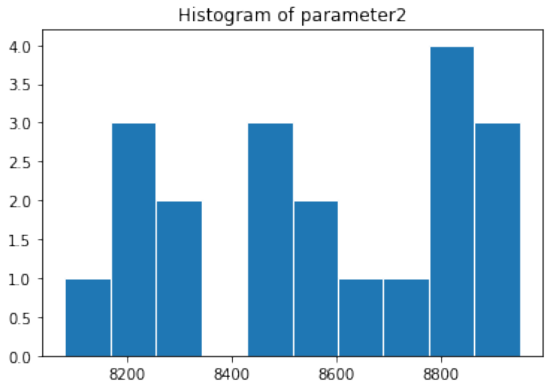
(b) Distribution of parameter D_n last population



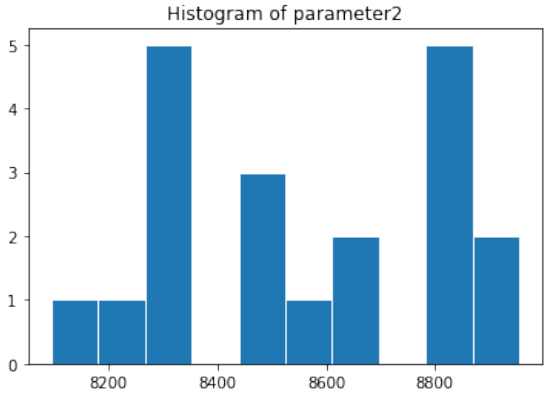
(a) Distribution of parameter D_c first population.



(b) Distribution of parameter D_c last population



(c) Distribution of parameter a_- first population



(d) Distribution of parameter a_- last population

6.1 Inference of parameters

After observing the previous results, we conclude the ABC-SMC scheme applied to the cancer model offers good parameter inference results. However, looking at data proportioned by the laboratory on day 6, it is quite remarkable the results obtained do not reflect

reality. It seems to be caused by a problem of ill conditioned matrices, as we are working with large data. Therefore, in the following sections the Implicit Euler method is used without using matrices. The computational cost grows due to the number of iterations but it will give the correct results.

Applying Implicit Euler to the model using the parameters given in table 6.1 the approximation obtained to data on day 6 can bee seen in the following images, 6.5 and 6.6.

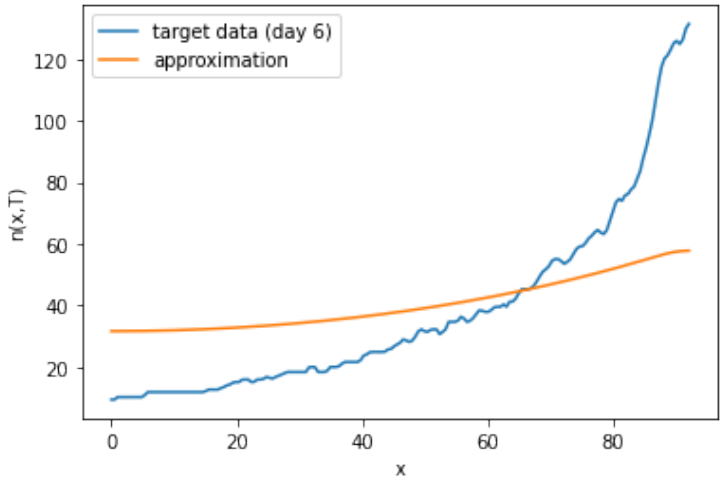


Figure 6.5: Approximation of the cells population to day 6 of the experiment using the parameters given in table 6.1

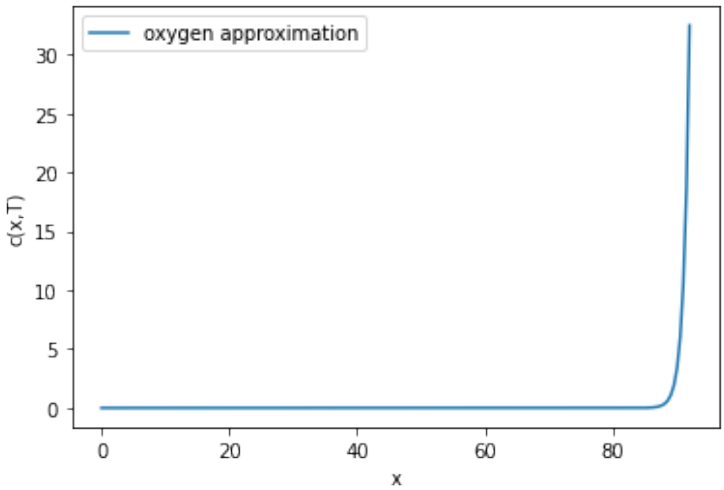


Figure 6.6: Approximation of the oxygen to day 6 of the experiment using the parameters given in table 6.1

As the approximation observed in figure 6.5 with the parameters given in table 6.1 is not accurate, the main goal of section 6.1.1 is to obtain a better approximation to the target data by doing parameter inference with the ABC SMC method. The number of parameters involved in the equations that govern the model is large, therefore, some of them will be fixed with values found in the literature.

6.1.1 ABC-SMC on the cancer model

In this case, initial conditions for the cells population $n(x,0)$ are those taken from the experiment in **doblare**, presented in the previous section (the data corresponding to day 3 of the laboratory which contains 191 equally spaced points), see figure 5.1. The initial condition for the oxygen concentration is, as already mentioned, the equilibrium from homogeneous equation, $c(x,0) = \frac{D_c \beta_f}{L(k_1 \overline{U} + k_2)}$.

Five parameters have been inferred:$D_n, D_c, a_-, k_1, \beta_f$. The prior distributions are:

* $D_n \sim U(0.000001, 0.01)$,
* $D_c \sim U(10.0, 150.0)$,
* $a_- \sim U(20000.0, 29000.0)$,
* $k_1 \sim U(1.000, 10.0)$,
* $\beta_f \sim U(20.0, 50.0)$

The function selected to measure the distance between the data $\{x_d[i], y_d[i])\}$, $i = 1, ..., 191$, and a simulated solution for proposed parameters, $\{(x[i], y[i])\}$, is the $L_2 - norm$ which is just the square root of (SSE). To ensure the gradual transition between populations, we take 8 populations with tolerances $\varepsilon = [350.0, 250.0, 180.0, 130.0, 110.0, 100.0, 90.0]$. Some simulations have been done and after accepting 5 particles at each population, the best results obtained in the last iteration after applying ABC-SMC scheme with the minimum distance are shown below.

The number of particles accepted at each population (5) is low compared to the ones used in the deterministic models analysed in the previous section. This is due to the fact of the computational cost of solving the method.

Third Best approximation

Table 6.2 shows one of the particle of parameters inferred at the last population of ABC-SMC. The distance of the data obtained using these parameter values to the target data (data of day 6 of the experiment) is 73.79457173365871.

Parameters	
D_n	0.007049090938336662
D_c	96.01253560520095
a_-	25981.85387914363
k_1	2.115428342336973
β_f	42.31739218472538

Table 6.2: Parameters inferred. Third best approximation

Figure 6.7 is a representation of the result:

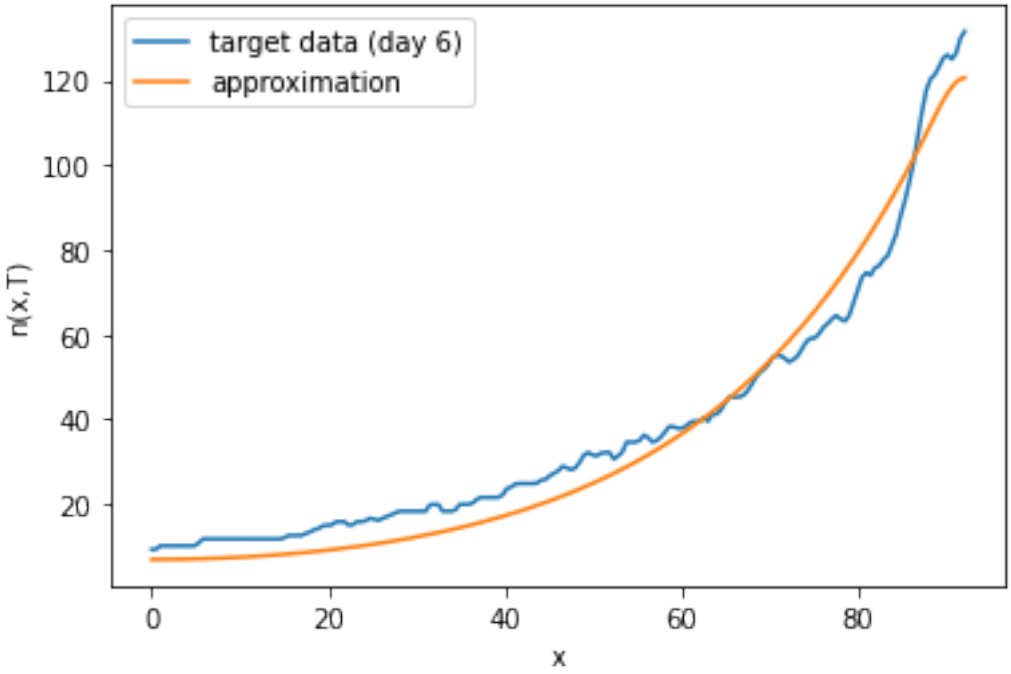


Figure 6.7: Third best approximation obtained in the last population with parameter values presented in table 6.2. Blue line represents real data, and the orange line is the approximation obtained.

Second best approximation

Table 6.3 shows one of the particle of parameters inferred at the last population of ABC-SMC. The distance of the data obtained using these parameter values to the target data (data of day 6 of the experiment) is 73.46239844770133.

Parameters	
D_n	0.009317505047591068
D_c	130.57613724264993
a_-	27447.069045209162
k_1	2.54346670111937
β_f	44.116759054456516

Table 6.3: Parameters inferred

Figure 6.8 is a representation of the result:

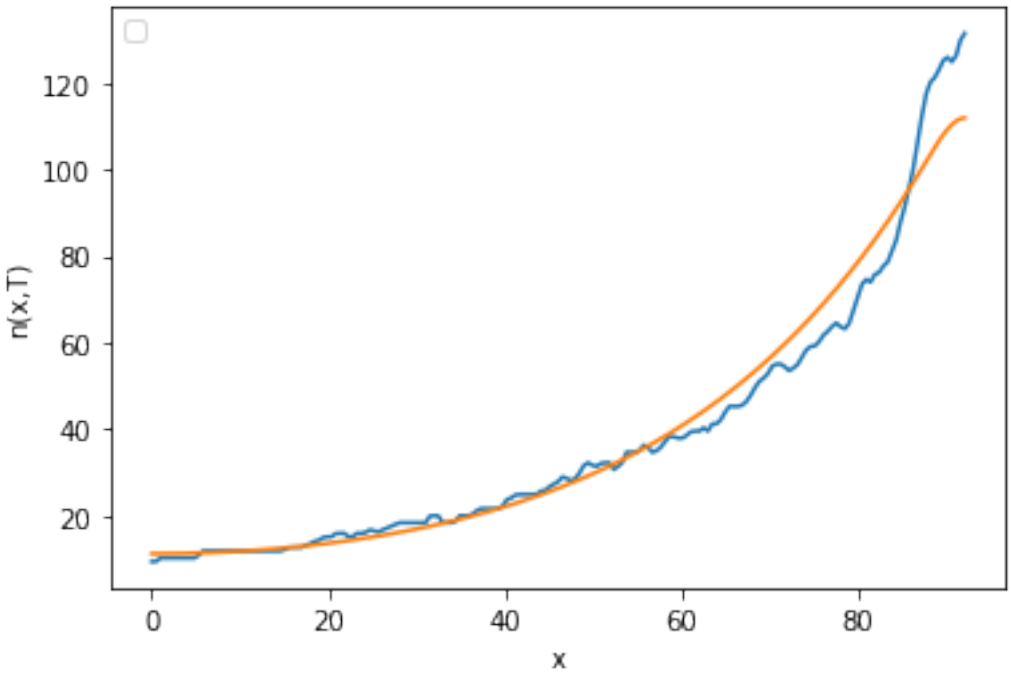


Figure 6.8: Third best approximation obtained in the last population with parameter values presented in table 6.3. Blue line represents real data, and the orange line is the approximation obtained.

Best approximation

Table 6.4 contains the particle of parameters inferred at the last population of ABC-SMC which best approximates to real data.

The distance of the data obtained using these parameter values to the target data (data of day 6 of Doblaré) is 71.43485739641646.

Parameters	
D_n	0.0073784225942921675
D_c	75.2638546327538
a_-	27009.525920887948
k_1	1.706101845276988
β_f	49.99839948737975

Table 6.4: Best Parameters approximated

Figure 6.9 is a representation of the result using the parameter in Table 6.4:

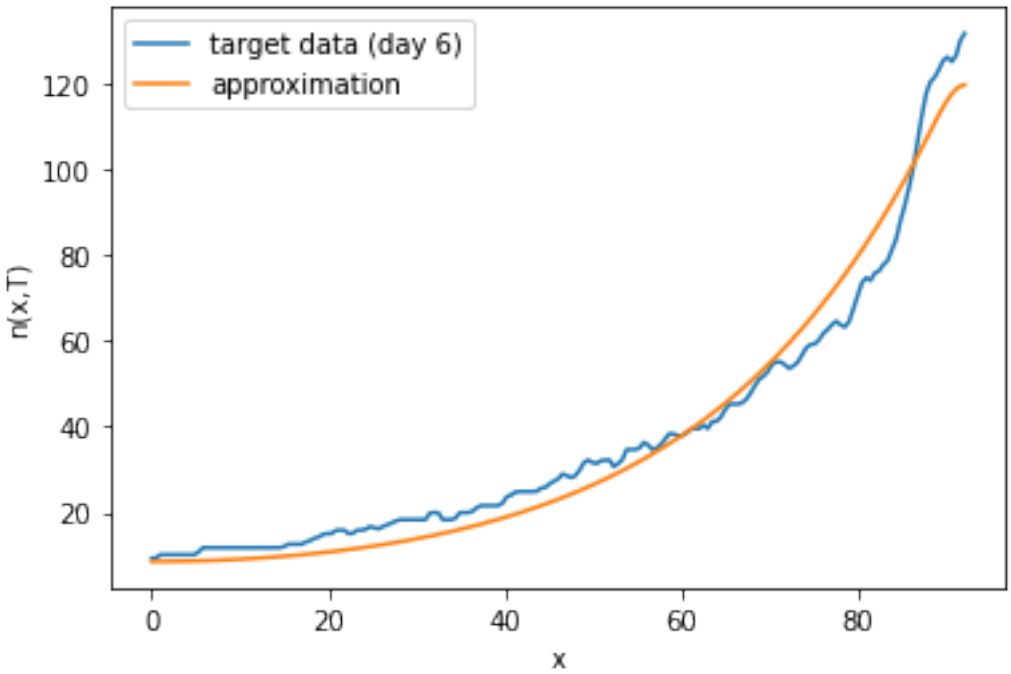
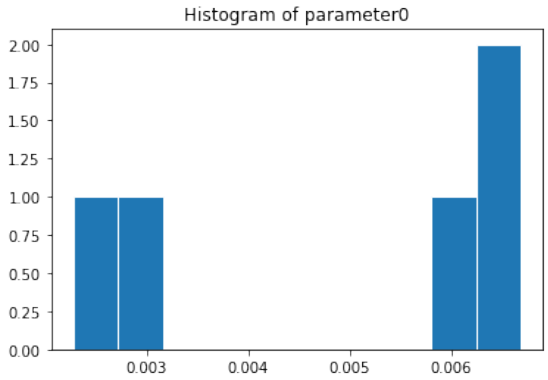


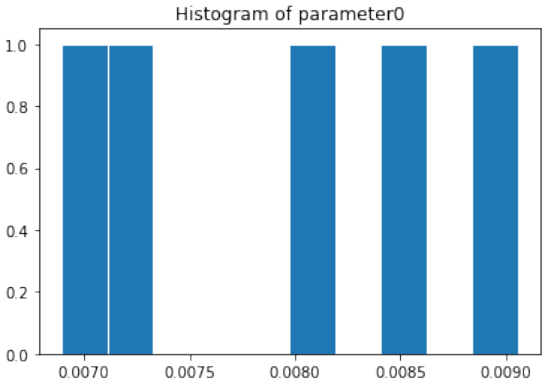
Figure 6.9: Best approximation obtained in the last population with parameter values presented in table 6.4. Blue line represents real data, and the orange line is the approximation obtained.

6.1.1.1 Histograms

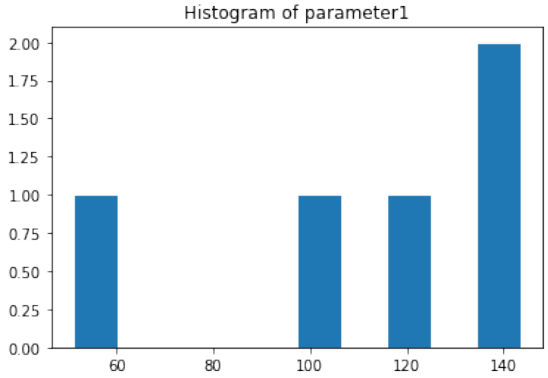
The following image shows the histograms of the particles values obtained for the first population (with tolerance $\varepsilon_0 = 350.0$) and for the last population (with tolerance $\varepsilon_7 = 90.0$).



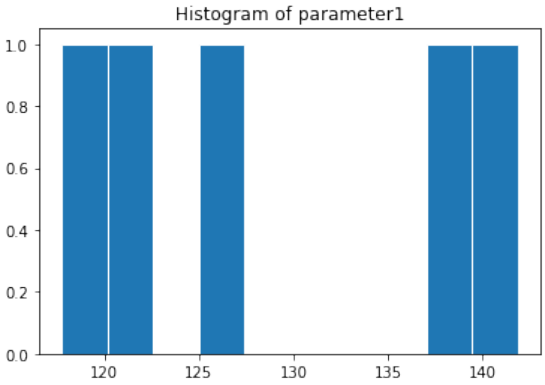
(a) Distribution of D_n first population



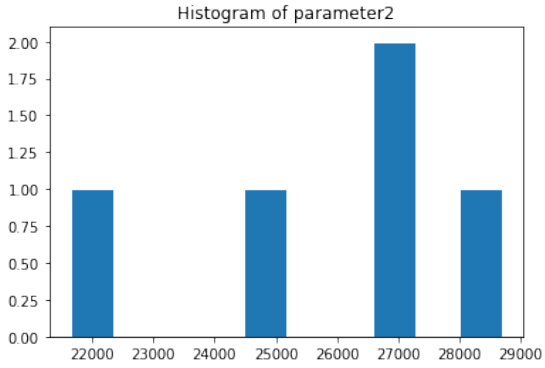
(b) Distribution of D_n last population.



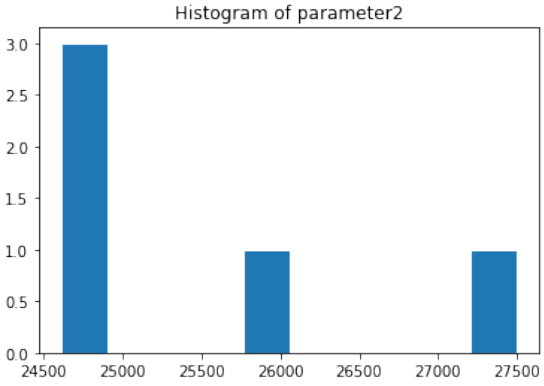
(c) Distribution of D_c first population



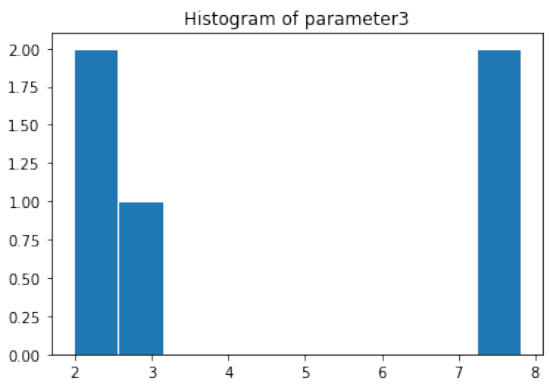
(d) Distribution of D_c last population



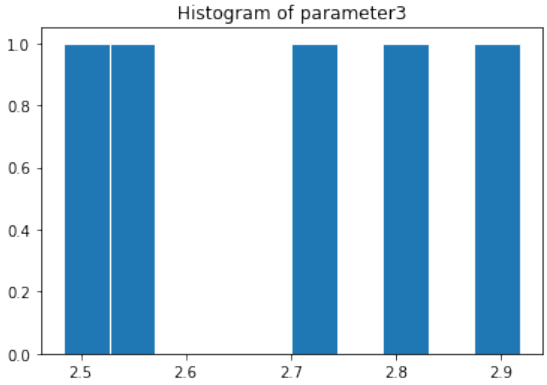
(e) Distribution of $a\_$ first population



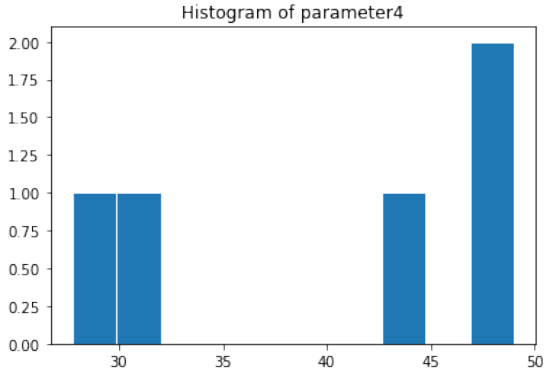
(f) Distribution of $a\_$ last population



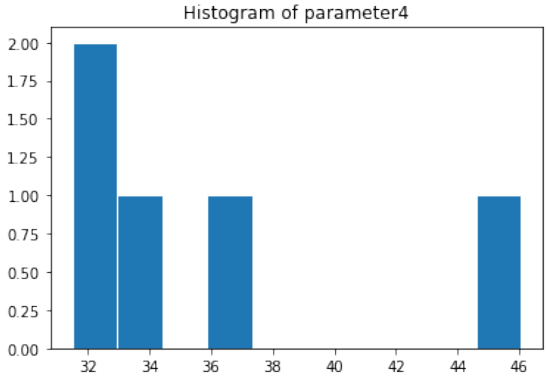
(a) Distribution of k_1 first population



(b) Distribution of k_1 last population



(c) Distribution of β_f first population



(d) Distribution of β_f last population

From the previous results, it can be observed the best approximations obtained are quite accurate, having the approximated or experimental data and the target data a similar behaviour, as the shape of their increasing curve is similar.

Attempt to approximate data of day 9

As we also had access to the real data observed on day 9 of the experiment, some trials were done to see if our model is accurate by using the parameters previously inferred. By fixing parameters according to the values given in table 6.4, and using as initial data the approximation obtained with them (The orange line on figure 6.9 is initial data for the cells population and initial oxygen concentration is given by 6.12). The result obtained is shown in figure 6.13.

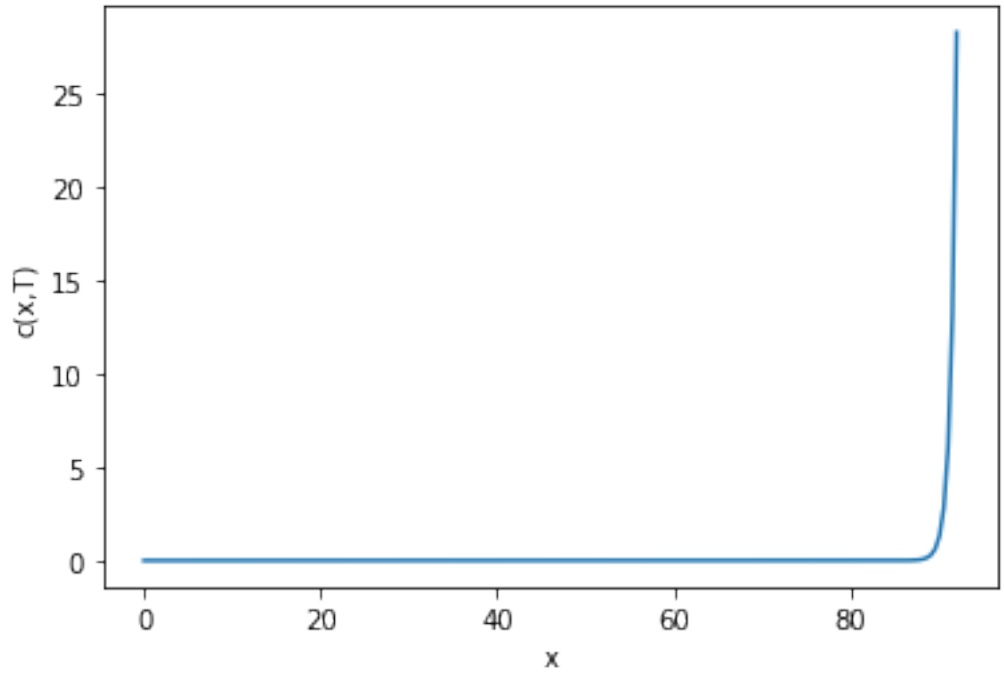


Figure 6.12: Oxygen concentration inferred (day 6)

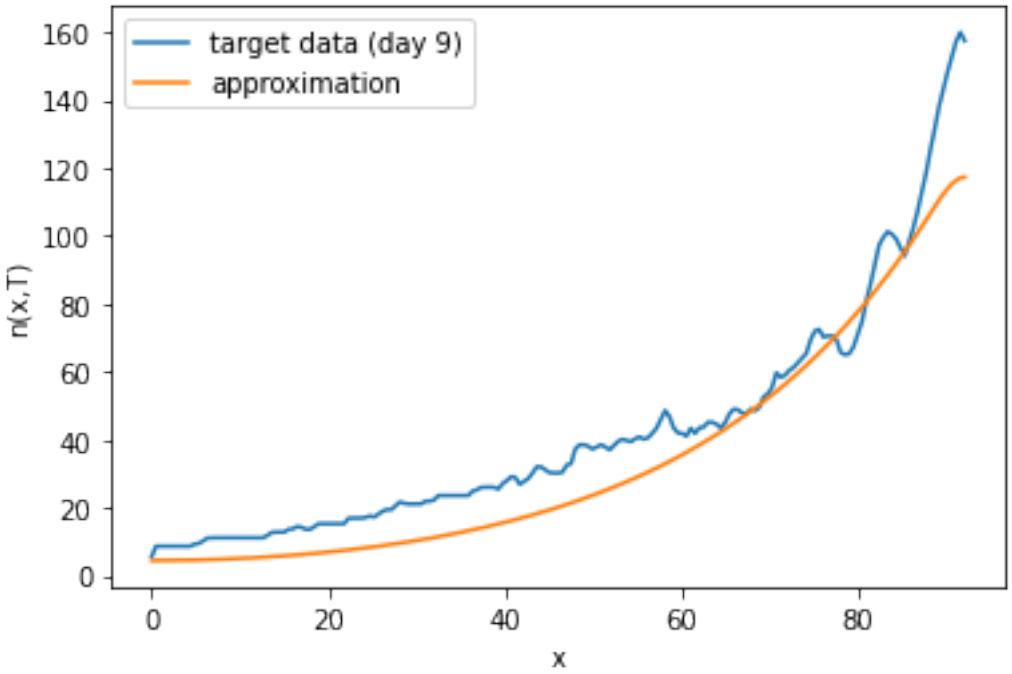


Figure 6.13: Approximation to data of say 9 taking as initial conditions the approximation done with ABC-SMC at day 6.

Even though the approximation obtained is not as accurate as the one obtained to ap-

proximate data on day 6, we can observe a similar behaviour on the curves. The worst approximation is obtained at the right boundary, as we had already observed in figure 6.9. This can be caused by different reasons: the initial conditions may not be well measured and are noisy, moreover, the mathematical model used may have some limitations regarding the representation of the main mechanisms and interactions and the parameters involved are taken over a wide range of values.

Considering a different particle which has also given a good approximation to data on day 6:

$\theta = (0.006066950183550362, 146.79606476000794, 21713.805172516404,$
$2.739918202870899, 24.759673174062115)$

The approximation to day 9 can be seen below on figure 6.14:

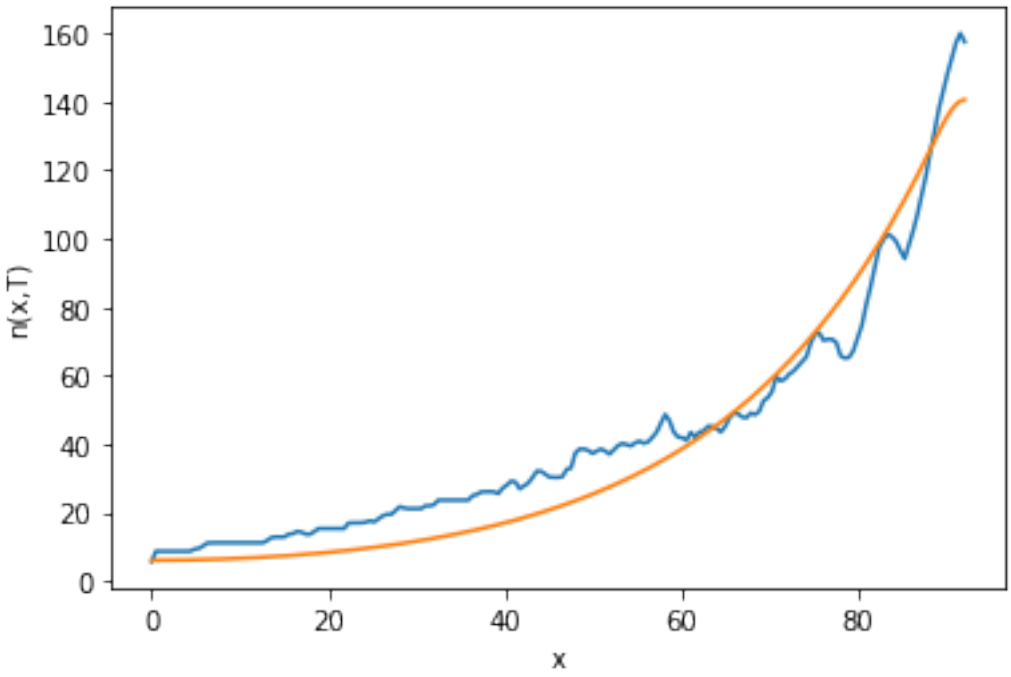


Figure 6.14: Approximation to day 9 using one of the particles inferred.

The whole procedure of the algorithm can be seen on the code `https://github.com/vgarcialopezm/ABC-SMC/blob/main/ABCSMCIMPEUL.ipynb`. It is remarkable that the parameter that most influences the accuracy of the model is D_n, small changes in this pa-

rameter give very different results.

Chapter 7

Conclusions

Experimental and theoretical research help each other to study and understand a specific process. To merge these two different points of view allows researchers to bridge over many of their corresponding limitations. To facilitate the integration of experimental data with theoretical studies, we propose a method based on Bayesian optimization.

Sequential Monte Carlo (SMC) methods have become a extremely powerful tool in approximate Bayesian computation (ABC), being one of the most efficient Monte Carlo sampler frameworks. In this project an ABC-SMC algorithm has been developed and applied to estimate model parameters, in this case, to different deterministic models describing biological processes. Its accuracy has been proved as the inference process has reached the parameter values required. In Chapter 4, we have seen the powerful of this method for calibrating in silico models as Lotka Volterra and SIR models.

Regarding the tumor growth model, the great amount of parameters involved in the equations that govern the process as well as the huge variation in the range of many of them found in the literature, make it very difficult to apply the ABC-SMC approach to infer the correct values because the intervals or domains for every parameter cover several orders of magnitude according to previous experiments and results found in the bibliography. This can be a result of the high heterogeneity of the glioblastoma.

The ABC-SMC algorithm has been applied to infer parameters involved in a tumor growth

model. The results have been compared to real data from the experiments done by the Doblaré laboratory, in Zaragoza. This model involves a wide range of parameters and therefore, it is complicated to obtain a good approximation of all of them. Moreover it is hard to identify the precise values of such parameters, due to the diversity of models and experimental conditions. That is why some have been fixed according to information obtained from the literature and the method has been applied to infer five, which is a considerable number regarding the difficulty of the model.

After observing the results obtained, we can conclude that the ABC-SMC method allows to get a good approximation to the target data even though the number of particles at each population must be small because of the computational cost of solving the model,i.e, the approximations could be better or more populations could be considered in case it didn't take so much time to apply the numerical method to solve the equations. Moreover, as it has already been mentioned, the initial data for the cells population may be noisy and the initial values for the oxygen concentration are not known.

Regarding future work we can say that with the ABC-SMC method we have been able to obtain values for five of the most representative parameters involved in the mathematical model: the diffusion coefficients of both the cells population and the oxygen concentration D_n and D_c respectively, the term a_m involved in the calculation of $a_{G1/S}$, the oxygen consumption rate k_1 and the flux of oxygen on the right boundary β_f. With these constants the model describes the evolution of Glioblastoma Multiforme (GBM), the most aggressive and lethal primary brain tumour. And compared to real data, it offers a really good approximation. This result can help in the future to have a better understanding of the cancer evolution, and therefore to find better strategies to eradicate this health problem.

Appendices

Appendix A

ABC-SMC code

```
# -*- coding: utf-8 -*-
"""ABC_SMC_ALGORITHM.ipynb

Automatically generated by Colaboratory.

Original file is located at
    https://colab.research.google.com/github/ugarcialopezm/ABC-SMC/
        blob/main/ABC_SMC_ALGORITHM.ipynb

# ABC SMC

This notebook contains all the necessary functions to include in
    the ABC SMC algorithm.
"""

import numpy as np
from scipy.stats import norm, uniform, multivariate_normal
from scipy.optimize import minimize
from scipy.special import logsumexp
import sys,ast
from math import exp
from math import log
from random import choices,seed,random
```

```
from tqdm import tqdm
#import p_tqdm
from functools import partial
import os
import matplotlib.pyplot as plt
from numpy import exp

"""The function *euc_dist* calculates the SSE (sum of squared
   errors) betweeen two data sets. The distance between them is
   compared to the tolerance fixed to decide if the sample is
   accepted or not."""

def euc_dist(data1, data2):
    if np.shape(data1) != np.shape(data2):
        print ("\n the dimensions of the datasets are different (%s
           v.s. %s)\n" % (len(data1), len(data2)))
        sys.exit()
    else:
        z=np.array((data1 - data2)**2)
        distance = np.sum(z)
        #print('dist',data1 - data2)

    if distance < 0:
        return [None]
    elif np.isnan(distance):
        distance=100000
        return distance
    else:
        return distance

"""The function *prior* generates a random parameter inside the
   limits stablished for each of them according to a Uniform
   distribution. It is used at first step of ABC SMC (time t=0)."""

def prior():
### Generate a random parameter inside the limits stablished. The
```

```
    shape of the distribution can be changed if required
    prior = []
    for ipar,par in enumerate(params_tumor):
        prior.append(uniform.rvs(loc = par['lower_limit'],
                                 scale = par['upper_limit']-par['
                                     lower_limit'])) #par['
                                     upper_limit']))


    return prior

"""*evaluate_prev_pru* : function that given the values of the
    parameters of a particle, obtains the joint probability density
    function."""

def evaluate_prev_pru(params):
    print('parameters',params)
    l=len(params)
    prior = 1
    for ipar,par in enumerate(params_tumor):
    #for i in range(l):
        prior *= uniform.pdf(params[ipar],loc = par['lower_limit'],
                                 scale = par['upper_limit']-par['
                                     lower_limit'])
        if prior==0:
            break
      #  print('params i', params[i])
       # print('prior',prior)
    return prior

"""*perturb*: function that, given a list of parameters sampled (a
    particle), perturbs it by applying a multivariate normal kernel
    """

#function that, given a list of parameters sampled, perturbs it by
    applying a multivariate normal kernel
```

```
def perturb(listaprev,s):
    #print(listaprev)
    lista=np.asarray(listaprev) #.tolist()
    #mean_vec=np.mean(lista)
    cov_matrix=2.0*np.cov(lista.T)  #the covariance matrix for the
       multivariate normal perturbation kernel is given by this
       expression
    kernel=multivariate_normal(cov=cov_matrix)
    pert=s+kernel.rvs() # here we obtain the list of perturbed
       parameters
    pertur=pert.tolist()
    return pertur

"""*weighting*: function that gives the denominator used to
   calculate the weights of every particle"""

#function that gives the denominator used to calculate the weights
   of every particle.
def weighting(i,j,N,sam,wei,sampre):
     denom=0
     #ker=1
     samprev=np.asarray(sampre)
     cov_matrix=2.0*np.cov(samprev.T)
     kernel=multivariate_normal(cov=cov_matrix)
     for k in range(N):
            sampre[k]=np.array(sampre[k])

            ker=kernel.pdf(sam[j]-sampre[k])
            #print('ker',ker)
            #kerne=np.prod(ker)  #here we are obtaining the joint
               probability of the parameter vector obtained when
               applying the kernel
            denom+=wei[k]*ker #kerne
     #print('den',denom)
     return denom
```

```
#function used to normalize the weights
def normalize(wei):
    #normalized=wei/np.linalg.norm(wei)
    normalized=wei/np.sum(wei)
    return normalized

"""# ABC-SMC algorithm

The principal function uses all the previous functions in order to
   generate the ABC SMC algorithm as it has been stablished in the
   project. The goal is to obtain a sampled list of parameters that
    gives the best approximation to the target data.
"""

def principal(epsilons,listaparametros,N,data1,t):
    T=len(epsilons)
    weight=np.zeros((T,N),float)
    dist=np.zeros((T,N),float)
    sample=np.zeros((T,N),list)
    X0=[1.0,0.5]
    #t=np.linspace(0.,10,10)
    for i in range(T):
        count=0
        counti=0
        label=i
        print("SMC step with target distance: {}".format(epsilons[i
           ]))
        if i==0:  #first time step. The particle is sampled from
           the prior distribution.
            for j in range (N):
                dist[i,j]=epsilons[i]+1
                while dist[i,j]>epsilons[i]:
                    sample[i,j]=prior()
                    #sample[i,j]=np.array(prior())
                    sample[i,j]=np.asarray(sample[i,j])
                    data2= rk4(lotka_volterra,X0,t,sample[i,j]) #
```

```
                    this function changes depending on the model
                     to solve as well as the number of
                    parameters
                #print('data2',data2)
                #data2=np.array(data2, dtype=np.float64)
                dist[i,j]=euc_disti(data1,data2)  #distance
                    between target data and the new data
                    obtained to decide if accepting or not the
                    particle.
                #print('distcondata2',dist[i,j])
            count+=1
            print(count)

    else:  #procedure followed after the first time step. The
        particles are sampled from the previous one

        for j in range (N):
            dist[i,j]=epsilons[i]+1
            while dist[i,j]>epsilons[i]:
                seed()
                np.random.seed()
                choose = choices(sample[i-1,:], weights =
                    weight[i-1,:],k=1)[0] # select a point from
                    the previous sample
                sample[i,j]=choose
                #print("before perturb",type(sample[i,j]))
                #print("before perturb",list(sample[i-1,:]))
                sample[i,j] = perturb(list(sample[i-1,:]),
                    sample[i,j]) # and perturb it
                #print("after perturb", sample[i,j])
                #print("after perturb", type(sample[i,j]))
                evaluation=evaluate_prev_pru(sample[i,j])
                if evaluation>0:
                    data2=rk4(lotka_volterra,X0,t,sample[i,j])
                        #solve the model with the parameters
                        inferred
```

```
                        data2=np.array(data2)
                        #print('data2',data2)
                        dist[i,j]=euc_disti(data1,data2) #calculate
                            distance to target data
                        print('distendata2',dist[i,j])
                counti+=1
                print(counti)
        for j in range(N):
            if i==0:
                weight[i,j]=1
               # print(weight[i,j])
            else:
                denom=weighting(i,j,N,sample[i,:],weight[i-1,:],
                    list(sample[i-1,:]))
                weight[i,j]=evaluate_prev_pru(sample[i,j])/denom
        #print('weight[i,:]',weight[i,:])
        if i!=0:
           weight[i,:]=normalize(weight[i,:])
    return sample, weight, dist,data2
```

Appendix B

Numerical methods

B.1 Newton Raphson algorithm

The Newton-Raphson algorithm is a technique used to locate good approximations of the zeros of a continuous and differentiable function f.

$$\text{Let } f : \mathbb{R}_n \to \mathbb{R}_n \text{ have a zero at } x^*, \text{ that is, } f(x^*) = 0.$$

The Newton-Raphson algorithm calculates x^* by iterating from an initial guess x_0 using the relation:

$$x_{(i+1)} = x_{(i)} - \frac{f(x_{(i)})}{f'(x_{(i)})}$$

where the subscripts indicate the iteration count.

The convergence of Newton Method is guaranteed, i.e offers good results, if the initial guest is close to x^*, then x_0 must be carefully chosen.

B.1.1 Newton Raphson code used to solve $\lambda(c)$

```
def newton_raps(c,am,ccr,nu,beta):
    #print('c',c)
    if c>ccr:
      aG1S=am*(c/ccr-1)**(-beta)
    else:
      aG1S=10000000000
    #print(aG1S)
    #print('ag1s',aG1S)
    #Expression of the function to obtain the value of lambda
    f=lambda lam:log(taup)-log(2)+lam*aG1S+nu*aG1S+log(lam+nu+1/
       taup)

    #Fist derivative of th previos function
    Df=lambda lam: aG1S+1/(lam+nu+1/taup)
    #tabla = []
    section = abs(2*tolerance)
    xi = 0
    while (section>=tolerance):
        xnew = xi - f(xi)/Df(xi)
        section  = abs(xnew-xi)
        #tabla.append([xi,xnew,section])
        xi = xnew
    return xi
```

B.2 Runge Kutta method order 4

Runge–Kutta method is an effective and widely used method for solving the initial-value problems of differential equations. Runge–Kutta method can be used to construct high order accurate numerical method by functions' self without needing the high order derivatives of functions.

```
def rk4(f,in_c,t,params):
    #params=[a,b,c,d]
    #h=t[1]-t[0]
    n=len(t)
    X  = np.zeros([n,len(in_c)],dtype=np.float64)
    X[0]=in_c
    for i in range(n-1):
      h=t[i+1]-t[i]
      k1=f(t[i],X[i],*params)
      k2=f(t[i]+h/2.,X[i]+k1*h/2.,*params)
      k3=f(t[i]+h/2,X[i]+k2*h/2.,*params)
      k4=f(t[i]+h,X[i]+k3*h,*params)

      X[i+1]=X[i]+h*(k1/6.+k2/3.+k3/3.+k4/6.)

    return X
```

Appendix C

Extraction of data techniques of the experiment

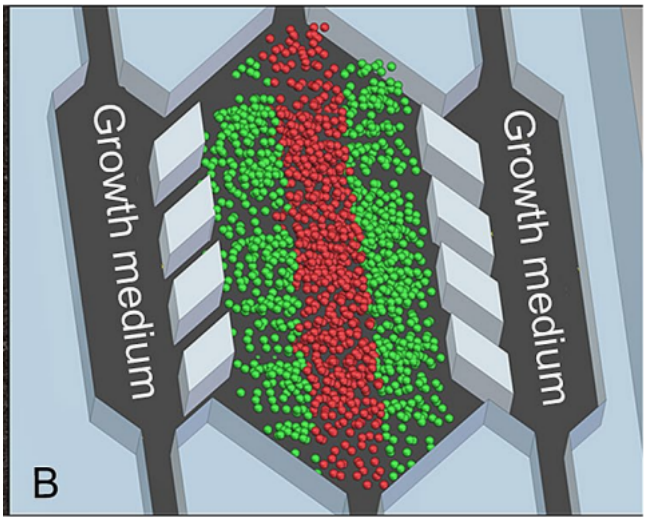


Figure C.1: Schematic view of the central region of the polystyrene/COP microdevice and necrotic core formation.Extracted from **doblare**.

In order to form a 3D structure, oxygen impermeable microfuidic devices (BEOnChip Ltd.) consisting of a central chamber and two lateral microchannels were used (C.1). They had diferent dimensions and were made of $SU-8$, polystyrene or cyclic olefn polymer (COP), using different fabrication processes. 3D distribution of cells was achieved

within the central chamber, using collagen hydrogel, which was well resuspended and injected into the central chamber of the microfuidic device using a micropipette. The hydrogel droplet was placed on the top of the inlet to prevent evaporation. The devices were placed into an incubator $(37C, 5\%CO_2)$ for 15 min to promote collagen gel polymerization. Aferwards, pre-warmed growth medium was perfused through the lateral channels, mimicking blood vessels, and refreshed every 24 hours (**doblare**).

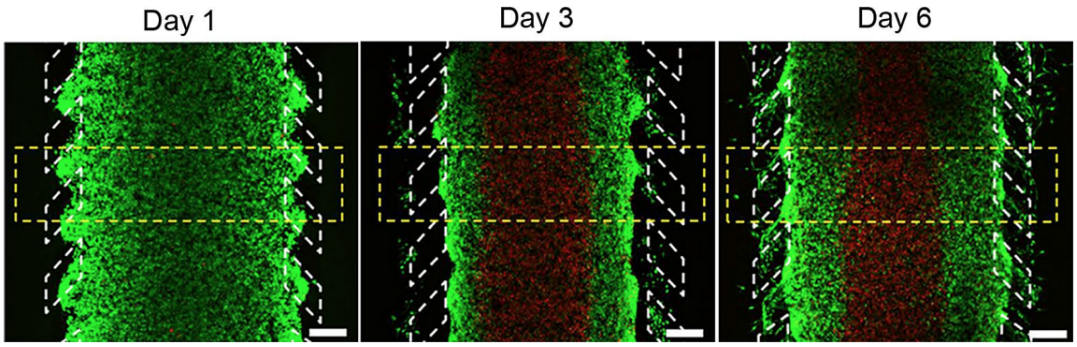


Figure C.2: Necrotic core formation. Viable cells are stained green with calcein AM and dead cells are labelled red with propidium iodide.Extracted from **doblare**.

In order to produce the necrotic core formation, a high density of cells was embedded in the collagen hydrogel and injected within the central microchamber. Growth medium was refreshed every day and the culture was maintained for 6 days. Oxygen are not able to reach the central part of the device due to cell consumption close to the microchannels, thus causing cell death in the central region appearing an autoinduced necrotic core. See figure C.2), mimicking the parts of the tumour far from functional blood vessels **doblare**

.