Study of Parameter Estimation and Model Calibration Using Bayesian Analysis of Noisy Data for a Virus Model

Alejandro Mejia

STUDY OF PARAMETER ESTIMATION AND MODEL CALIBRATION USING BAYESIAN ANALYSIS OF NOISY DATA FOR A VIRUS MODEL

Alejandro Mejia

Thesis submitted to the
Benjamin M. Statler College of Engineering and Mineral Resources at
West Virginia University

in partial fulfillment of the requirements for the degree of

Master of Science
in
Aerospace Engineering

Department of Mechanical and Aerospace Engineering

Morgantown, West Virginia
2017

Keywords: Bayesian analysis, Markov chain Monte-Carlo, virus dynamics, parameter estimation, mathematical modeling

Abstract

Study of Parameter Estimation and Model Calibration using Bayesian Analysis of Noisy Data for a Virus Model

Alejandro Mejia

Numerous engineering problems are concerned with the challenge of representing real life systems through mathematical equations: modeling. Properly generated mathematical models can accurately predict the behavior of natural processes. The key objective of model development is to correctly build a set of equations or expressions that can reproduce the results observed from experimental measurements. By following this methodology, sources of error and uncertainty will arise as most natural processes have random factors that make the results stochastic, and therefore can never be exactly reproduced. A model can try to best approximate the actual outcome, but many times assumptions or simplifications are needed because either the problem becomes mathematically unfeasible, or there is not enough knowledge regarding the process. Additionally, even if models are correctly defined, they may require proper calibration of its parameters to make predictions.

In the study of virology, within host viral infections can be modeled by means of mathematical balances of target cell populations. A virus model will describe how a virus infects healthy cells and spreads by defining a set of depletion/replenishment rate parameters that will depend on each system. The focus of this study is to determine the posterior probability distributions of these parameters that will best approximate a given patient's data, similar to data fitting. Using an inverse modeling approach to generate patient data using known reference values of the virus model parameters and adding random "white" noise, a virus model will be fitted to the generated noisy data using Bayesian methods for parameter estimation.

The main purpose of this study is to validate the use of Bayesian calibration techniques as an alternative to conventional gradient-based parameter estimation methods. The results of a calibrated virus model with a fixed virus generation rate are then used to make model predictions and extrapolate the dynamic behavior to different ranges of the fixed parameter. The results conclude that Bayesian methods can be successfully used for parameter estimation, especially for high-dimensional problems, however the practical identifiability of the parameters is limited by the model's nonlinear terms, the experimental data variance, and the available data measurements. Although the results are encouraging, the excessive computation time needed for obtaining the empirical parameter Posterior distributions limit the practical use of these methods.

Acknowledgements

First, I would like to extend my sincerest gratitude to my research advisor, Dr. David Mebane for his support and guidance during my time in graduate school, as well as giving me the opportunity to be part of his research group. His advice, encouragement, and dedication to my academic success has allowed me to become a better researcher and professional. I would also like to thank my research committee members, Dr. John Christian and Dr. Debangsu Bhattacharyya, for their time, comments, and suggestions that have helped me during my research at West Virginia University. A special thanks to Dr. Pieter Schmal from PSE for providing the necessary input to formulate and design the problem case for my thesis.

I would like to thank my graduate colleagues and friends, Maria Alejandra Torres Arango and Javier Cabrera Barbero, for the academic and personal support they provided during my time at WVU. Also, many thanks to the current and former members of our research group: Brian Logsdon, Keenan Kocan, Kuijun Li, Xiaorui Tong, Nirjhar Alam, Levi Hubbard, and Alex Zurhelle. It was great working with everyone, the relaxing coffee meetings and great moments shared have taught me so much and given me the opportunity to learn from everyone. I wish all of you a great future and successful careers filled with accomplishments.

Finally, I would like to thank my parents for giving me the opportunity to pursue my academic and professional goals, and especially for their loving support during this unforgettable experience at WVU. Many thanks to my wonderful family, my loved ones, Mariana Villegas for providing the best motivation, and my great friends who have always been there to give me their advice and support, and help me be a better person every day. Thank you.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

Mathematical modeling is a useful tool for describing real life processes and systems. If properly defined, it can provide valuable results from precise equations that approximate and predict a system's behavior. Modeling is used in many fields of study and areas of research; thus understanding its fundamentals is an important skill, especially for modern engineering. With the continuing development of computers and more advanced computational software, there is an increasing use of numerical methods to model and analyze progressively more complex systems by taking advantage of better computational speed and power. This has opened a doorway to the ongoing expansion of new methods and techniques of numerically modeling physical systems.

Although modeling can be a powerful tool, it is not without limitations and caveats that must be considered to avoid invalid results and conclusions. Although the study of modeling limitations is not the main interest of this study, one of the challenges that is discussed in this thesis is the issue related to the implementation of a numerical solver for model calibration. Once a mathematical model is properly defined, there is usually a concern regarding model calibration that will allow the properly calibrated model to predict, under given conditions, future system behavior. This ability to make accurate predictions is one of the key advantages of correctly modeling a system.

For the purpose of this research, model calibration refers to the process of determining the unknown parameters of a dynamic mathematical model using experimental data, so-called inverse modeling or data fitting. The calibrated model represents a function that describes the behavior of the system that best approximates the numerical results to the actual measured data, and can be used to make predictions. A wide variety of methods exist for calibrating computer models, however, in recent years Bayesian analysis has gained much interest in its implementation for statistical inference, allowing the making of accurate predictions and informed decisions by employing probability distributions [1].

With this in mind, the focus of this thesis is to develop and analyze the implementation of a computer-based Bayesian calibration routine for an existing mathematical model. The model chosen for testing the calibration code is a simple dynamic virus model that has been used in many studies (Gumel *et al*. [2], Nowak and May [3]) to replicate the initial phase of a virus infection. Although the calibration tool may be adapted to other dynamic systems and engineering problems, this model is used because it is a known existing mathematical model used in the biomedical/pharmaceutical industry and the experimental data set can be obtained from existing patient records if real life implementation is anticipated. This model was also initially provided as a test case from PSE's gPROMS® ModelBuilder platform to determine if this commercial advance process modeling software could potentially take advantage of an existing Bayesian calibration routine through the use of a Foreign Process Interface communication.

The main purpose of implementing a Foreign Process Interface (FPI) is to have gPROMS® be the model solver, which reads a parameter set as an input, and deliver an output containing the model's measured virus which could be calibrated to a single infected patient data-set. The output from gPROMS® would be used by the calibration routine to explore the parameter space and get a posterior distribution. This methodology was only used to test the FPI and the feasibility of implementing a calibration tool because the extensive computation time required for every iteration makes the Bayesian tool impractical. Because of this undesired model evaluation time, an independent C++ version of the virus model solver is also developed to produce the same output and reduce the overall calibration time.

1.1 Background and Previous Research on Virus Modeling

In clinical research, epidemiology refers to the study of how diseases, and their symptoms, can spread within populations of individuals. In this sense, the dynamics of a disease spreading is examined on a large population scale. When studying viruses, research is also aimed at learning how an individual's system behaves after becoming infected. This within host cell-to-cell infectious micro scale spreading phenomenon is of high interest in determining the characteristics of currently known viruses, and help study new ones after they are discovered [3].

Viruses have been studied for many years, and with extensive investigation, some have even been successfully vaccinated against. There is still ongoing research and development in this field. The main goal of understanding virus behavior and its process of replicating is to minimize its ability to spread within susceptible populations and effectively reduce or inhibit its adverse effects inside individual hosts. Once a virus vaccine is successfully developed, it can be used to assist the patient's immunological system in defending itself from future infection by identifying the foreign infectious agent.

One of the viruses of high concern in recent years, since its discovery in 1981, is the Human Immunodeficiency Virus (HIV). The HIV infection attacks the host's immunological system and is characterized by three distinct phases: the primary or acute infection phase, the asymptotic latency phase, and the final progression into acquired immunodeficiency syndrome (AIDS). The primary phase is identified by the early initial period after infection, where a rapid exponential increase in viral load occurs until reaching a maximum peak. The time of HIV primary infection varies greatly from patient to patient, but is usually observed in a period of a few weeks, where symptoms may be present. After the primary phase reaches its peak viral load, the total virus count decreases gradually and settles on a quasi-steady-state viral load that differs for each patient. This phase can last for several years, sometimes without showing symptoms or significant detection, and eventually depletes the host's population of healthy target helper T-cells to a critical level, developing into AIDS.

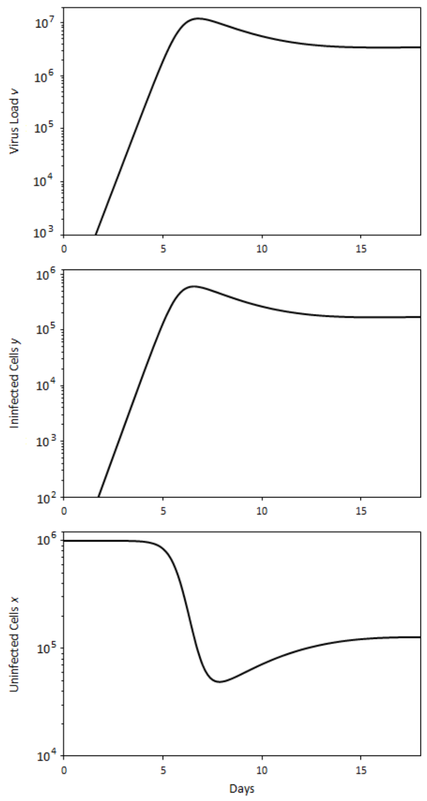

Figure 1-1 HIV primary phase infection progression

Mathematical modeling of HIV virus dynamics has been centered on the primary phase (Murray *et al.* [4], Tuckwell and LeCorfec [5], Stafford *et al.* [6]) and most studies are focused in attempting to reproduce the various patient behavior after initial infection occurs (see Figure 1-1). The target limited virus model used in these studies of primary HIV infection has six rate parameters $[a\ \beta\ d\ k\ \lambda\ u]$ as shown in Figure 1-2, taken from the reference textbook (Nowak and May [3]).

$$\begin{aligned}\dot{x} &= \lambda - dx - \beta x v,\\ \dot{y} &= \beta x v - a y,\\ \dot{v} &= k y - u v.\end{aligned}$$

Figure 1-2 HIV target limited virus model

The currently existing virus models can be used for HIV progression studies by adjusting the different rate parameters to cover a wide range of patient immunological responses. Ciupe *et al.* [7] attempt to estimate the kinetic parameters of a target cell limited model and replicate the oscillating behavior observed in patients during the primary phase, that are not reproducible by the basic model configuration. The exact moment of infection is rarely known, and similarly, the time until the T-cells become activated and respond to the virus infection. In their study, a time delay is added to the model as an additional parameter to account for the host immune system response. Leenheer and Smith [8] use this additional time delay parameter to represent an immune system response that can properly model oscillations that are sometimes observed in patients.

1.2 Virus Model Parameter Identifiability

One of the common challenges found in previous HIV mathematical model studies is the issue related to parameter identifiability. Regardless of the model used, previous authors (Xia and Moog [9], Wentworth *et al.* [10], Wu *et al.* [11]) noted that using measured virus data alone makes the identifiability of all the model parameters mathematically achievable, but difficult in practice due to the patient data being characteristically noisy. Moreover, other studies (Burg *et al.* [12]) concluded that using patient data from the asymptotic phase of infection makes parameter identifiability and estimation impossible. A generally adopted solution for this virus progression measurement constraint is to perturb the host dynamics by introducing HIV viral treatment that inhibits the virus replication and thus generates a disturbance in the quasi-steady-state virus balance found in the asymptotic phase. This approach relies on taking samples after treatment is introduced and finding the parameters from the data obtained.

Sensitivity analyses for parameter identifiability have shown that depending on the experimental data available, some of the parameters are difficult to identify individually. Soetaert and Petzoldt [13] analyze the identifiability of the different combinations of five parameters for a six-parameter HIV virus model, noting in their study that the parameter set $[a\ \beta\ d\ \lambda\ u]$ has the lowest collinearity, and is therefore easiest to identify. Table 1-1 shows the sensitivity analysis results from Stoetaert and Petzoldt for the possible parameter combinations, noting that the identifiability of the parameters is possible using both measurements of virus particles and cell populations.

Table 1-1 Model parameter collinearity

	a	β	d	k	λ	u	N	collinearity
1	1	1	1	1	1	0	5	41
2	1	1	1	1	0	1	5	51
3	1	1	1	0	1	1	5	12
4	1	1	0	1	1	1	5	34
5	1	0	1	1	1	1	5	36
6	0	1	1	1	1	1	5	15
7	1	1	1	1	1	1	6	53

* *Data obtained from Stoetaert and Petzoldt* [13]

Chapter 2

Virus Model

Viruses are understood as infectious, microscopic particles that contain genetic material (RNA or DNA) and need living cells found in organisms in order to "infect" their nucleus and replicate themselves. Because a virus cannot reproduce outside of a host cell it is debated whether it should be considered as a living or non-living organism. Virology, the study of viruses and their life cycles, attempts to understand virus behavior. The theoretical modeling of viral behavior is mathematically analogous to the dynamics and survival of a prey-predator population.

2.1 Mathematical Virus Model

In studying the dynamics of a virus, a basic mathematical model is established which has three state variables for each type of population present: the number of uninfected cells (x), the number of infected cells (y), and the measure of free virus particles (v). As previously mentioned, there are six *rate* parameters that are used to help describe the relationship between these state variables and model the time lapse for each variable, once an initial infection has occurred. The following figure shows a diagram depicting the relationship between each of the time dependent variables x, y, and v, with the inbound/outbound arrows representing source and sink terms, respectively.

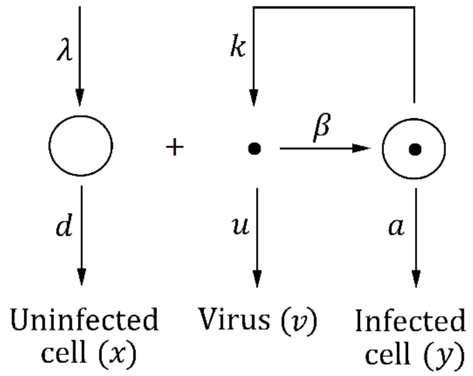

Figure 2-1 Illustration of the dynamics of a basic virus model

Intuitively a virus infection can be regarded as a dynamic process (see Figure 2-1) where an uninfected cell population system, initially at rest, or likewise at equilibrium, is disturbed by an infection from an arbitrary amount of virus particles. Once infected, the system will undergo changes in the infected/uninfected cell populations due to the creation (replication) and destruction of virus particles, also known as virions, and its original cell population dynamics. In other words, the system has an initial uninfected cell population balance, which will be altered once an infection is present. This new infected system will now either return to a state of equilibrium with the infection and death rates leading to the virus particles dying out, or to the continuous growth and eventual overflow of infected cells and virus particles. The outcome of this chain reaction will ultimately depend on the characteristics of the virus which infects the system. For a given virus, the dimensionless ratio known as the reproductive ratio of infection (R_0) will determine how the system will behave in time. A reproductive ration less than one means that, on average, every virus particle produces less than one new virus particle and therefore, in future time generations, the virus count will gradually decrease and be eliminated from the system. If instead R_0 is greater than one, the virus will continuously produce more new virus particles. The reproductive ratio of infection is an important characteristic of infections, and usually of high interest because it gives an idea of how infectious a disease or virus can be.

In an uninfected patient, both y and v are zero. However, when a patient is infected with any amount of virus particles, v_0, the uninfected cells become infected with an infection rate which is proportional to the product of the variables x (number of uninfected cells) and v (number of free viruses): $\boldsymbol{\beta} xv$. Next, the recently infected cells produce new free virus particles at a rate $\boldsymbol{k}y$. The uninfected and infected cells each die with a death rate of $\boldsymbol{d}x$ and $\boldsymbol{a}y$, respectively. Also, the free virus particles die or are removed from the system at a rate $\boldsymbol{u}v$. In modeling an uninfected cell's dynamics, it is assumed that new cells are generated at a constant rate $\boldsymbol{\lambda}$. Alternatively, some authors model this replenishment rate of new cells, λ, as a time-varying parameter. For a constant λ and if no virus infection is present, the differential equation that models the uninfected cell population in a system is given as follows:

$$\frac{dx}{dt} = \dot{x} = \boldsymbol{\lambda} - \boldsymbol{d}x \tag{1}$$

It can be shown from this linear ODE that an uninfected system will equilibrate to the steady state value $x = \lambda/d$ that balances the replication and death of uninfected cells in a system. These relationships, which can be viewed as source and sink terms, provide the basis for the mathematical setup of the virus target cell-limited model as follows:

$$\frac{dx}{dt} = \dot{x} = \boldsymbol{\lambda} - \boldsymbol{d}x - \boldsymbol{\beta}xv \tag{2}$$

$$\frac{dy}{dt} = \dot{y} = \boldsymbol{\beta}xv - \boldsymbol{a}y \tag{3}$$

$$\frac{dv}{dt} = \dot{v} = \boldsymbol{k}y - \boldsymbol{u}v \tag{4}$$

These three equations are used to model the time-dependent interaction of a virus infection in a system. Hundreds of viruses are currently known to exist and infect living organisms such as humans, animals, plants, and bacteria. The Human Immunodeficiency Virus (HIV) is one example of a virus which is modeled using these equations. Table 2-1 provides a summary of the parameters used to model the virus behavior, and a set of reference values for HIV from a virus dynamics reference textbook [3].

Table 2-1 Virus model HIV parameter values

Parameter Description	**Symbol**	**HIV model***	**Bounds**	**S.D.**
Death rate of infected cells	a	0.5	[0.0 1.0]	0.1
Virus infection rate	β	$2*10^{-7}$	[10^{-9} 10^{-3}]	$1*10^{-5}$
Death rate uninfected cells	d	0.1	[0.001 0.8]	0.01
Free virions generation rate	k	100	[0 1000]	1
New cells replenishment rate	λ	$1*10^{5}$	[100 10^{6}]	$5*10^{3}$
Virus death rate	u	5	[1 100]	2.5

** Values obtained for a reference HIV model from Nowak et al* [3]

From these expressions, it is convenient to mathematically define and calculate the reproductive ratio of infection as follows:

$$R_0 = \frac{\lambda \beta k}{dau} \tag{5}$$

For the reference values used in Table 2-1, the reproductive ratio of infection for HIV is computed as $R_0 = 8$. Equations (2), (3), and (4) will be referred to in the remainder of this text, as the Bayesian MCMC calibration tool will be used to estimate the five of the six unknown rate parameters. For a given initial viral load v_0, the system of equations can be numerically solved using a discretization method which will be discussed in the following section.

2.2 Numerical Discretization and Model Solver

The method selected to discretize and numerically approximate (solve) the virus model differential equations is an implicit Crank-Nicolson scheme, which provides a stable second-order convergence in time stencil. In their studies, Ciupe et al. [7] use a modified Runge-Kutta (RK) method with a step size $\Delta t = 0.01$. Alternatively, Gumel et al [2] propose a faster Gauss-Seidel-like method for this same virus model and compare its stability to an RK2 method, showing that this "implicit" discretization scheme can also be used. As will be mentioned later, this first-order accurate solution greatly reduces the computation time needed. Similarly, an Euler method was initially tested in this study, but not used for calibration because of the inherent stability issues of this scheme. The selected method for solving the PDEs is shown, where the superscript *n* indicates the time iteration as follows:

$$\frac{du}{dt} = \frac{u^{n+1} - u^n}{\Delta t} = \frac{1}{2}[F^{n+1} + F^n] \tag{6}$$

A stencil for the Crank-Nicolson discretization scheme is presented in Figure 2-2.

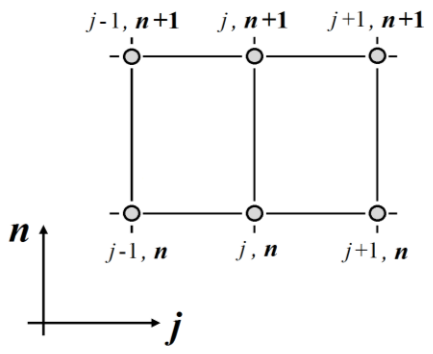

Figure 2-2 Crank-Nicolson discretization method stencil

Applying this stencil to equations (2), (3), and (4) we obtain the following set of discretized equations:

$$\frac{x^{n+1} - x^n}{\Delta t} = \lambda - \frac{d}{2}(x^{n+1} + x^n) - \frac{\beta}{2}(x^{n+1}v^{n+1} + x^n v^n) \tag{7}$$

$$\frac{y^{n+1} - y^n}{\Delta t} = \frac{\beta}{2}(x^{n+1}v^{n+1} + x^n v^n) - \frac{a}{2}(y^{n+1} + y^n) \tag{8}$$

$$\frac{v^{n+1} - v^n}{\Delta t} = \frac{k}{2}(y^{n+1} + y^n) - \frac{u}{2}(v^{n+1} + v^n) \tag{9}$$

To numerically solve these equations, a Newton-Raphson root-finding solver is implemented by moving all the terms to one side and equating each expression to zero. The Newton solver will provide a method of iteratively converging to the roots of these equations for each time step Δt. In order to implement this solver using a computer-based algorithm, the inputs are specified as vectors. Additionally, a Jacobian matrix is required to compute the partial derivatives of the equations with respect to each state variable.

The three discretized equations above are rewritten so that the virus model can be expressed in matrix form:

$$0 = x^n - x^{n+1} + \Delta t\left[\lambda - \frac{d}{2}(x^{n+1} + x^n) - \frac{\beta}{2}(x^{n+1}v^{n+1} + x^n v^n)\right] \tag{10}$$

$$0 = y^n - y^{n+1} + \Delta t\left[\frac{\beta}{2}(x^{n+1}v^{n+1} + x^n v^n) - \frac{a}{2}(y^{n+1} + y^n)\right] \tag{11}$$

$$0 = v^n - v^{n+1} + \Delta t\left[\frac{k}{2}(y^{n+1} + y^n) - \frac{u}{2}(v^{n+1} + v^n)\right] \tag{12}$$

These expressions are now combined as a vector function $F(\bar{x})$, where $\bar{x}$ is a *3x1* column vector and the entries correspond to each state variable at the iteration $n + 1$:

$$F(\bar{x}) = F\begin{pmatrix} x^{n+1} \\ y^{n+1} \\ v^{n+1} \end{pmatrix} \tag{13}$$

The vector function of inputs x, y, and v evaluates the virus model balance equations for a new time iteration $(n + 1)$, knowing the previous state (n), the time step, and the corresponding parameter values:

$$F(\bar{x}) = \begin{pmatrix} x^n - x^{n+1} + \Delta t\left[\lambda - \frac{d}{2}(x^{n+1} + x^n) - \frac{\beta}{2}(x^{n+1}v^{n+1} + x^n v^n)\right] \\ y^n - y^{n+1} + \Delta t\left[\frac{\beta}{2}(x^{n+1}v^{n+1} + x^n v^n) - \frac{a}{2}(y^{n+1} + y^n)\right] \\ v^n - v^{n+1} + \Delta t\left[\frac{k}{2}(y^{n+1} + y^n) - \frac{u}{2}(v^{n+1} + v^n)\right] \end{pmatrix} \tag{14}$$

The Jacobian matrix $F'(\bar{x})$ for this function is computed by taking the derivative of each equation with respect to the state variables at the next iteration *n+1*:

$$F^{'}(\bar{x}) = \begin{pmatrix} \frac{\partial F_1(\bar{x})}{\partial x^{n+1}} & \frac{\partial F_1(\bar{x})}{\partial y^{n+1}} & \frac{\partial F_1(\bar{x})}{\partial v^{n+1}} \\ \frac{\partial F_2(\bar{x})}{\partial x^{n+1}} & \frac{\partial F_2(\bar{x})}{\partial y^{n+1}} & \frac{\partial F_2(\bar{x})}{\partial v^{n+1}} \\ \frac{\partial F_3(\bar{x})}{\partial x^{n+1}} & \frac{\partial F_3(\bar{x})}{\partial y^{n+1}} & \frac{\partial F_3(\bar{x})}{\partial v^{n+1}} \end{pmatrix} \tag{15}$$

Where $F_i(\bar{x})$ is the corresponding row (equation) of the vector function $F(\bar{x})$ for $i = 1, 2, 3$.

$$F^{'}(\bar{x}) = \begin{pmatrix} -1 - \Delta t\left(\frac{d}{2} + \frac{\beta}{2} v^{n+1}\right) & 0 & -\Delta t \frac{\beta}{2} x^{n+1} \\ \Delta t \frac{\beta}{2} v^{n+1} & -1 - \Delta t \frac{a}{2} & \Delta t \frac{\beta}{2} x^{n+1} \\ 0 & \Delta t \frac{k}{2} & -1 - \Delta t \frac{u}{2} \end{pmatrix} \tag{16}$$

The Newton-Raphson method for finding the roots to the vector function $F(\bar{x})$ is defined as follows:

$$x_{k+1}^{n+1} = x_k^{n+1} - \left[F'\left(x_k^{n+1}\right)\right]^{-1} F(x_k^{n+1}) \tag{17}$$

Where the k subscript specifies the Newton iteration, $\left[F'\left(x_k^{n+1}\right)\right]^{-1}$ is the inverse of the Jacobian matrix, and x_k^{n+1} is the current solution for the states, given as a vector.

The inverse of the Jacobian must be approximated using some numerical method that is both easy to implement and does not use excessive computation time, as it will need to be computed for every time step iteration. A simple Jacobi method that decomposes the Jacobian matrix into two matrices (diagonal and residual matrices) is used. This numerical method of finding the inverse of a matrix is less computationally expensive than an LU decomposition.

With the matrices and vectors from equations (14) and (16), equation (17) can be evaluated and iteratively solved for a given time step Δt. This process is implemented into C++ as a model evaluator for a given virus parameter set input. The algorithm output can be user-defined to assign a vector that contains the measured virus in a time interval that matches the experimental patient data. This method is tested with a working virus model in the gPROMS ModelBuilder® environment to verify that no coding error exists. The Crank Nicolson (C-N) finite difference numerical solver was implemented for correctly running the C++ based MCMC routine. Afterwards, a Gauss Seidel-like (GSS) method proposed by Gumel *et al*. [2] was found to have fast, stable, and accurate computational results, and eventually used for the final stage of this study, after initial results from the C-N algorithm were obtained. The time derivative is approximated using a first-order forward-difference numerical scheme as follows:

$$\frac{(x^{n+1} - x^n)}{\Delta t} = \lambda - dx^{n+1} - \beta x^{n+1} v^n \tag{18}$$

$$\frac{(y^{n+1} - y^n)}{\Delta t} = \beta x^{n+1} v^n - a y^{n+1} \tag{19}$$

$$\frac{(v^{n+1} - v^n)}{\Delta t} = k y^{n+1} - u v^{n+1} \tag{20}$$

These expressions can be re-arranged to leave the unknown iteration in terms of the known current states to give the following explicit linear algebraic system:

$$x^{n+1} = \frac{x^n + \Delta t \cdot \lambda}{1 + \Delta t(d + \beta v^n)} \tag{21}$$

$$y^{n+1} = \frac{y^n + \Delta t \cdot \beta x^{n+1} v^n}{1 + \Delta t \cdot a} \tag{22}$$

$$v^{n+1} = \frac{v^n + \Delta t \cdot k y^{n+1}}{1 + \Delta t \cdot u} \tag{23}$$

The virus model is solved using both the C-N and GSS methods discussed in this chapter, but the latter provides faster MCMC steps by omitting the need to compute the inverse of the Jacobian and will be used to produce the final results for the Bayesian calibration. The next chapter will discuss the fundamentals of Bayesian analysis needed to understand how the MCMC calibration tool will be used for analyzing a noisy patient data set and estimate the parameters for the HIV virus model.

Chapter 3

Bayesian Model Calibration

When calibrating a mathematical model, the main goal is to determine the unknown parameters in the model equations using measured experimental data, so-called data fitting. The measured data provides the necessary information to compare model output against. Many techniques exist for model calibration, such as Maximum Likelihood Estimation (MLE) and Least Squares (LS) based estimation. An alternative approach to getting model parameter estimations is through the use of Bayesian analysis, where the unknown parameters are defined as random variables and given a probability distribution, which represents degree of belief for the true parameter values. Bayesian model calibration is implemented using Bayes theorem for conditional probabilities.

3.1 Fundamental Statistics and Bayes' Theorem

Statistics is the discipline that focuses on collecting, organizing, presenting, and analyzing data that is generally gathered from experiments. One main branch of statistics is probability theory, which deals with the analysis of random experiments or events. From probability theory, the conditional probability of an event A occurring given that another event B has occurred is stated by:

$$P(A|B) = \frac{P(A \cap B)}{P(B)} \tag{24}$$

Figure 3-1 shows this concept graphically with a Venn diagram. Here, the sample space $\mathcal{C}$ contains all possible outcomes of a random experiment, where events A and B each have a corresponding probability not equal to zero. The conditional probability of an event can be viewed as an event that occurs within a new sample space given by the conditional event. Therefore, the conditional probability of A given B, mathematically expressed as $P(A|B)$, is the darker shaded region, where event B has already occurred and thus becomes the new sample space. Also from this diagram, it can be seen that the probability of event A occurring is equivalently expressed in terms of event B as:

$$P(A) = P(A \cap B) \cup P(A \cap B') \tag{25}$$

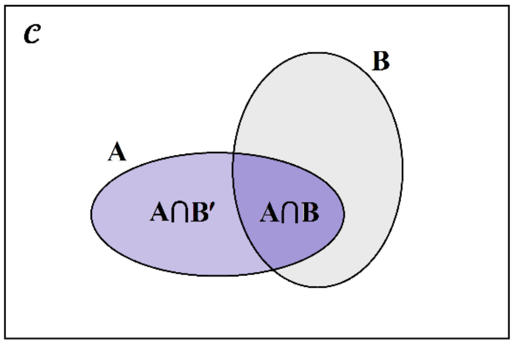

Figure 3-1 Conditional probability of two intersecting sets

We are interested in the events where both A and B have occurred. In other words the intersection of these events. From this notion, we can directly identify the conditional probability. Now, redistributing equation (24), we obtain the Multiplication Rule of Probability, which states that the probability of two events A and B occurring (intersection) is equal to the probability of one event occurring, $P(A)$, times the conditional probability of the other event B occurring given that the first event A occurred, $P(B|A)$, and vice versa:

$$P(A \cap B) = P(A)\, P(B|A) \tag{26}$$

$$P(B \cap A) = P(B)\, P(A|B) \tag{27}$$

Since the intersection of sets has Commutative properties, equations (26) and (27) are equal and can be combined to express the following:

$$P(A)\, P(B|A) = P(B)\, P(A|B) \tag{28}$$

Bayes' theorem immediately follows from this expression and states the probability of an event occurring, given prior information related to the conditional event. Mathematically Bayes' rule is stated as follows:

$$P(A|B) = \frac{P(B|A)\, P(A)}{P(B)} \tag{29}$$

Where A and B are two random events with probabilities $P(A)$ and $P(B) \neq 0$, respectively. $P(A|B)$ is the conditional probability of observing event A given that B has occurred, and similarly $P(B|A)$ is the conditional probability of event B occurring given that event A has occurred.

For the purpose of Bayesian calibration of mathematical models, event A is defined as an arbitrary parameter set θ (usually a vector) and B is the given experimental data set Z. Therefore, the conditional probability $P(A|B)$, or from here on $P(\theta|Z)$, is defined as the probability of a particular parameter set being the true values of the model, given that the specific data set Z is observed. In any multidimensional parameter space, this conditional probability has an unknown distribution and is referred to as the Posterior distribution (Π). The Posterior distribution is the desired result of the Bayesian calibration. Furthermore, the conditional probability $P(Z|\theta)$ is the probability that the experimental data set Y is observed given that the parameter set θ is the true value. This term is also known as the Likelihood ($\mathcal{L}$). Correspondingly, $P(\theta)$, which is the simply the probability of any parameter set being observed from the entire parameter space, is referred to as the Prior probability and it represents the observer's belief of a certain parameter set being true *prior* to any data information being taken into account. Finally, the term $P(Z)$ can be expressed using the Law of Total Probability for a continuous random vector and it is computed as the following integral:

$$P(Z) = \int_{\theta} P(Z|\theta)\, P(\theta)\, d\theta \tag{30}$$

The integral in equation (30), also known as the marginal likelihood, is often very difficult to evaluate, especially if there are many parameters (high-dimensional integration). However, the probability is only dependent on the observed data Z and is the same for every possible parameter set θ. Thus, Bayes' theorem (29) can be used to obtain the Posterior distribution, with the integral being only a normalizing factor or constant, thereby giving a proportional relation as follows:

$$P(\theta|Z) \propto P(Z|\theta)\, P(\theta) \tag{31}$$

Expressing the Posterior distribution as Π, the Likelihood function as $\mathcal{L}$, and including the observational error variance (ψ) as an additional unknown parameter for the calibration, we get the following useful expression:

$$\Pi(\theta, \psi|Z) \propto \mathcal{L}(Z|\theta, \psi)\, P(\theta)\, P(\psi) \tag{32}$$

Using this notation, the data set Z is assumed to have an observational error, and the model is regarded as being a perfect representation of the actual process. This observational error (ϵ) has a normal (Gaussian) distribution with mean zero and variance ψ, also known as "white noise". Thus, the errors can be expressed as the difference between the measured data and the "true" model values:

$$\epsilon_i = z_i - \tilde{z}_i \sim N(0,\psi) \tag{33}$$

Furthermore, the Likelihood can be computed from these independent and identically distributed (*I.I.D.*) observational errors using the joint pdf of n samples from a Normally distributed random variable:

$$\mathcal{L}(Z|\theta,\psi) = \prod_{i=1}^{n} \frac{1}{\sqrt{2\pi}\,\psi^{1/2}} e^{\left[-\frac{1}{2}\frac{(z_i-\tilde{z}_i)^2}{\psi}\right]} \tag{34}$$

The Likelihood cost function can equivalently be rewritten in terms of the sum of squared error (SS_q) as:

$$\mathcal{L}(Z|\theta,\psi) = (2\pi\psi)^{-n/2} \cdot e^{\left[\frac{-SS_q}{2\psi}\right]} \tag{35}$$

$$SS_q = \sum_{i=1}^{n} (z_i - \tilde{z}_i)^2 = \sum_{i=1}^{n} \epsilon_i{}^2 \tag{36}$$

Observing these equations, it is evident that the Likelihood for any proposed model will increase as the sum of squared error term decreases, which is exactly what is anticipated as it should provide a measure of closeness between the proposed model and the measurement data.

Defining the remaining term, $P(\theta)$, is an initial assumed distribution for the parameter set θ. Thus, one of the important aspects of Bayesian calibration is the selection of the prior distribution. Whenever certain information is known beforehand, it can be included in the calibration by means of the prior. If there is no information or knowledge about the parameter values, an uninformative prior is used and recommended, as an incorrect prior can have negative impact on the actual Posterior. This uninformative prior is specified by using a uniform distribution across the continuous parameter space, where every possible outcome is equally likely (see Figure 3-2):

$$P(\theta) \sim U(A,B) \tag{37}$$

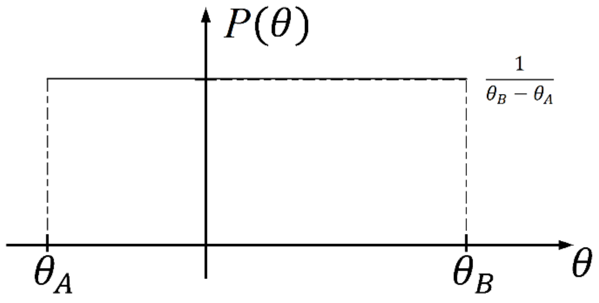

Figure 3-2 Uninformative prior for a parameter with bounds [A,B]

Lastly, the observational error variance ψ has an Inverse Gamma prior distribution, with observational shape (ν) and rate (τ) parameters:

$$P(\psi) \sim \text{Inv-gamma}(\nu, \tau) \tag{38}$$

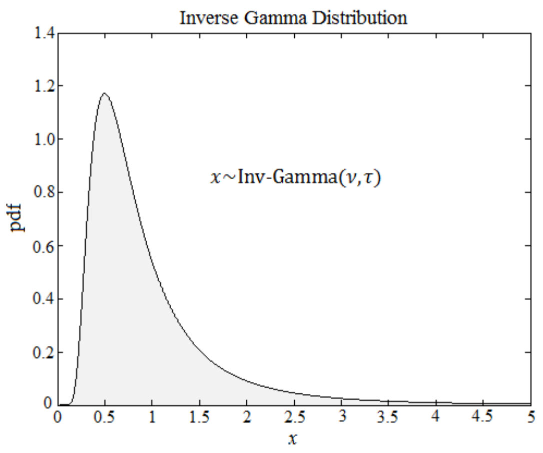

Figure 3-3 Inverse Gamma PDF

The Inverse Gamma distribution is often used for Bayesian inference because of the convenient property of being a conjugate prior distribution (posterior and prior distributions are from the same family) to a Gaussian “white noise” likelihood function. This conjugate property allows for direct sampling from the conditional posterior for the observational error variance without rejection.

Now that Bayes’ theorem provides a method of updating the Posterior distribution for an arbitrary point or parameter set, it is necessary to have a technique that allows for the exploration of the parameter space and after a certain amount of iterations approximate the posterior probability distribution. One sampling method for exploring the parameter space is through the use of a Markov chain Monte Carlo (MCMC) algorithm.

3.2 Markov chain Monte Carlo

In Bayesian calibration, a Markov chain Monte Carlo is a method of iteratively sampling from a probability distribution, and constructing a sample distribution that will represent the target posterior distribution. It provides an organized procedure for searching the parameter space for regions of higher Likelihood. The idea is that the Markov process will reach an equilibrium distribution that is a sample drawn from a simulation of the posterior distribution. The routine will begin at an initial parameter space starting point from which it will propose new parameter sets and evaluate their likelihood. If a proposed parameter set (θ_1) has a higher likelihood than the current parameter point (θ_0), then the chain will move to this new region or point in the parameter space. If the proposed point has a lower likelihood, then the algorithm will either reject the new set or accept it, with a certain probability, compared to a random draw from a Normal(0,1) distribution. This acceptance criteria is defined using the following acceptance ratio, which is computed every MCMC iteration:

$$r(\theta_1|\theta_0) = \frac{\mathcal{L}(Y|\theta_1,\psi_1)\ P(\theta_1)\ P(\psi_1)}{\mathcal{L}(Y|\theta_0,\psi_0)\ P(\theta_0)\ P(\psi_0)} \tag{39}$$

$$\theta = \begin{cases} \theta_1, & \text{with probability } p = \min(1,r) \\ \theta_0, & \text{else} \end{cases} \tag{40}$$

The Likelihood ratio is usually calculated in logarithmic scale, as the terms can quite often be very small. Additionally, the MCMC routine has an adaptive proposal option, which uses an empirical covariance matrix for the proposal distribution that is calculated from previously accepted states. Using only a certain number of prior states to calculate the covariance matrix, a fixed window is obtained. This adaptive proposal tool is subsequently used to increase/decrease the proposal step size in order to achieve a desired target acceptance rate. The specification of a target acceptance rate allows for the MCMC to avoid high acceptance/rejection rates that can lead to local convergence or limited parameter space exploration.

The statistical acceptance of points with lower likelihood allows the MCMC algorithm to step out of regions of local maxima/minima. Conceptually a Markov Chain with an infinite number of Monte Carlo steps will exactly match the desired Posterior distribution. However, this is clearly not possible to implement in real life applications and we can only run the algorithm for a certain amount of time. Nevertheless, if the MCMC routine runs for a sufficient number of steps, then an empirical probability density function (pdf) can be obtained that will represent a sample from the Posterior. Because the algorithm can endlessly continue to take MCMC steps, it is important to specify a convergence criteria that can indicate when the routine has obtained sufficient samples. Evidently, there is no single argument to specify when the MCMC has taken a “sufficient” number of samples, but a proposed batch means convergence, discussed in the next section, can be used to identify when the chain has reached a statistically significant convergence of the parameter means.

3.3 Batch Means Convergence Criteria

As mentioned in the previous section, an MCMC generates a random walk through the parameter space and will continuously take more steps until the algorithm is stopped or it reaches a predetermined number of steps. There are two important calibration properties that need to be considered for deciding when to stop the algorithm: burn-in and convergence. Burn-in refers to the initial region (arbitrary n number of steps) of the MCMC where the chains will be initialized at a selected starting point, which can potentially be random, and begin exploring the parameter space. These first n samples need to be discarded from the final empirical density distribution. After these n steps are taken, the chains will have ideally reached a parameter space region that contains the Posterior density and begin taking useful samples. Here an additional convergence condition is needed.

A statistical convergence criterion that can be quickly used to identify when the algorithm has successfully sampled the posterior distribution is through a batch means test. This convergence criterion will take the remaining $N - n$ samples, after burn-in, and create equal size bins, or batches. Next, each bin’s population mean is computed. The batch means test will determine how these estimated means vary and through a Student’s t-test give an indication of average batch means convergence with a confidence interval $\pm p\%$. A confidence interval of $\pm 1\%$ is ideal, but $\pm 5\%$ is also acceptable to reduce the overall simulation time. An arbitrary parameter’s (μ) batch means confidence interval is estimated as follows:

$$\mu = \bar{x} \pm \left[\left(\frac{t(0.975, N-1) * \bar{\sigma}}{\sqrt{N}} \right) / \bar{x} \right] * 100\% \tag{41}$$

Where $t(\cdot)$ is a t-statistic function that tests whether the means of two populations are close within a given confidence interval. As the confidence interval becomes smaller, the batch means are consequently closer and the chain samples can be regarded as converged.

Chapter 4

Parameter Estimation Methodology

This chapter outlines the steps involved in performing a parameter estimation through a Bayesian analysis approach algorithm using a MCMC to obtain the parameter posterior distributions. For this virus model case, a set of generated experimental values are needed for parameter calibration. The experimental data set represents measurements taken from an infected patient (usually number of virus particles per milliliter of blood) for a useful time interval normally given in a time-lapse of either days or months.

4.1 Experimental Data Generation

Actual patient data is not immediately available for testing the virus model and a technique for approximating real life data is desired. Hence, a method of generating the calibration experimental data was used. The generated data was obtained by modeling the virus function using a set of standard reference HIV parameters (Table 2-1). Additionally, the generated data has added noise to represent actual measurement error and variations, typically found in real life measurements, by using the following expression:

$$\log_{10} Z_{exp} = \log_{10} Z_{model} + 0.5 * N(0,1) \tag{42}$$

This process of adding noise relies on randomly generated values from a Normal distribution. Although it is not actual measured data, it has been shown that this type of noisy data can closely match actual data obtained from groups of similar patients. Figure 4-1 shows a plot of the model data and the “experimental” noisy data.

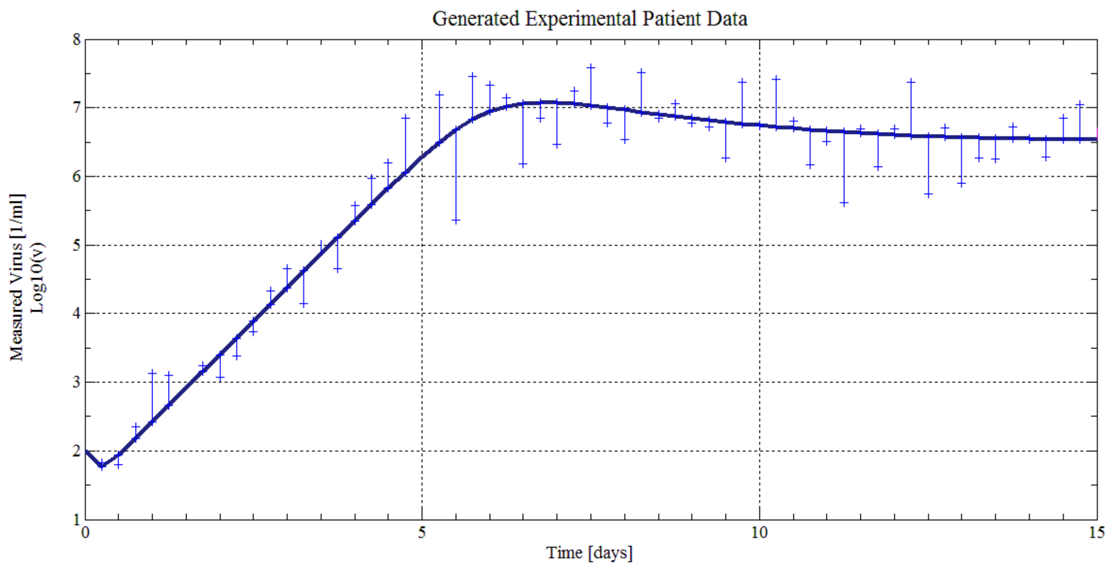

Figure 4-1 Patient data generated using HIV model parameters with added noise

The randomly generated noisy data has been shown to be a realistic representation of the variability seen in practice. Some studies have used similar methods of generating noisy virus data, with more or less variation.

4.2 Genetic Algorithm for Optimizing MCMC Starting Point

The MCMC code can be configured to start a run from any point within the parameter space bounds; however, random starting points can potentially take too long to properly explore the entire space and/or quickly reach a local minimum, especially for multidimensional problems. It is therefore desirable to have a means of selecting starting points that can reduce the unwanted effects of using a randomly generated starting point.

A basic genetic algorithm implementation, inspired on the natural evolution of species, consists of iteratively exposing a group or population of potential solutions (parameter sets), which represent individuals, to a user-defined cost or fitness function that represents the environment. The main idea is that only the fittest individuals (potential solutions) will “survive” in the specific environment and consequently be more likely to reproduce and create new offspring that inherit the parent’s good traits through a crossover operator that can randomly combine the parents’ chromosomes. In a computer-based GA, a chromosome contains the individual’s “genetic” information, or for parameter optimization problems, the actual values for each of the parameters (also referred to as genes). The new populations are also exposed to a mutation operator that ensures more randomness in the process by changing one or several genes. After iteratively exposing each new generation of individuals to the fitness function, the algorithm will ideally have explored the parameter space in search of better fit individuals and eventually reach a population that contains many good solutions and no apparent improvement in performance index.

A simple genetic, or evolutionary, algorithm (GA) for minimizing the experimental and model data error is implemented to find a starting point that has a dynamic behavior that can closely represent the experimental data. The starting point for the MCMC can be an individual selected from the final genetic algorithm population based on a performance index provided by the algorithm's fitness function. The performance index selected is a weighted average of four different error measurements: maximum error, root mean square deviation (RMSD), sum of squared error, and total error. The performance index is assigned in a [0 1] interval, with 1 being a perfect match of the model with the data and not actually expected.

The GA routine will generate a user-defined number of random individuals, which represent the initial population. As mentioned previously, an individual represents a potential solution, or set of model parameters. After an individual is simulated using the virus model numerical solver, a fitness function will assign a performance index (PI) to that corresponding set of parameters that will provide a measure of the accuracy between the individual's model output and the experimental data. The randomness used in this technique means that a different potential solution will be obtained each time the algorithm is ran. Having an initial random seeding spread out across the parameter space and a crossover/mutation rate for the population in every generation also gives this method the useful algorithm characteristic of potentially avoiding local minima. Ciupe et al. used a similar search technique in their estimation of the kinetic parameters [7].

4.3 MCMC Setup

The methodology applied in this study uses an adaptive sampler to account for high/low acceptance rates of proposed parameters. The adaptive sampler allows the MCMC to explore more locally if the acceptance rate is too low and inversely widen the proposal window if the acceptance of new parameter sets is too high. Therefore, the user can specify the bounds on the adaptive sampler as well as assign a switch to stop the adaptive steps after burn-in. Additionally, the MCMC allows for several other user inputs that can improve the overall results and/or calibration time.

Some of the MCMC parameters that are user-defined include the model parameter starting points, parameter bounds, covariance matrix recalculation rate, adaptive sampling switch, observational error shape (ν) and rate (τ) parameters, and number of desired MCMC steps, which can also be replaced with a convergence criteria if available.

This user-defined configuration implies that some knowledge or experience with Bayesian calibrating is necessary to correctly tune and obtain good results. An alternative to having expert knowledge is to run the MCMC with a predetermined setting to get preliminary results and then retune the algorithm configuration parameters appropriately. This is particularly useful for determining chain burn-in and identifying when to break the adaptive sampler. Additionally, the MCMC can be restarted from the last chain point by inputting the Covariance matrix. This is also useful for resetting the Covariance matrix when the chains seem to "flatline" and show slight movement, which is visually identified when the chains remain in a narrow range, as shown in Figure 4-2.

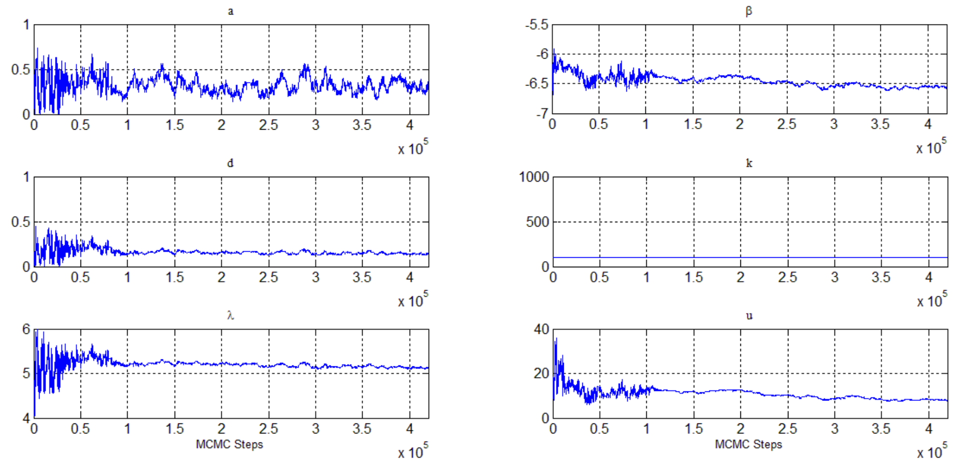

Figure 4-2 Flatlined parameter chains

The following table provides a layout of the Virus Model MCMC user-defined configuration file used for calibrating the model parameters:

Table 4-1 MCMC configuration

MCMC user parameters	**Default Configuration**
Number of Parameters	6*
Inv-Gamma Shape Parameters (ν)	4
Inv-Gamma Rate Parameters (τ)	5
Number of MCMC Steps	420,000
Proposal Type	Block
Covariance Matrix Recalculation	Every 4,000 steps
Dynamic Step Recalculation Rate	Every 250 steps
Adaptive Switch Off	After 120,000 steps
Acceptance Rate Bounds	[1 10]
Log Scale Switch	[0 1 0 0 1 0]
Parameter Bounds/S.D.	(see Table 2-1)

* *Five calibration parameters, one fixed*

The parameter proposal type is a convenient user-defined configuration option. The currently used “BLOCK” proposal option allows for the MCMC to propose all of the calibration parameters at once, as opposed to the “SINGLE” proposal option that only proposes the parameters one at a time. Both options have their respective advantages and disadvantages: while the block proposal configuration allows the MCMC to take potentially larger steps in the parameter space and reduce the total number of model solver iterations, the single proposal will be more likely to produce accepted parameter sets at the cost of multiple model evaluations and increased computation time.

From the model described in the previous chapter, the calibration parameter set θ for the virus is given by the following vector:

$$\theta = [a, \beta, d, k, \lambda, u]^{\mathrm{T}} \tag{43}$$

Additionally, if any parameter is fixed for calibration purposes, then θ will correspond to the remaining unknown parameters. The experimental patient data is also defined as a vector that contains the measured virus particles in the patient at certain time intervals:

$$Z = [z_1,\ z_2, \dots,\ z_n]^{\mathrm{T}} \tag{44}$$

The sampling rate used for running the MCMC is every 0.25 time units. Because the model does not have any specific defining time scale, this sampling rate can be scaled to any time interval as needed by appropriately adjusting the rate parameter time units.

Lastly, the initial conditions are input as a vector containing the values for the three state variables:

$$\bar{x}(t = 0) = [x(0) \;\; y(0) \;\; v(0)]^{\mathrm{T}} \tag{45}$$

In this thesis, constant initial conditions are assumed for the study of one data set, but alternatively these conditions can be included as unknown parameters for the MCMC to estimate. The initial conditions used in this case are as follows:

$$\bar{x}(t = 0) = [10^6 \;\; 0 \;\; 10^2]^{\mathrm{T}} \tag{46}$$

With the parameter vector, model data vector, and initial conditions vector now properly defined, the MCMC routine can be setup to run a parameter estimation using the virus model numerical solver to evaluate the proposed parameter sets. The following block diagram in Figure 4-3 shows the general layout of the MCMC algorithm:

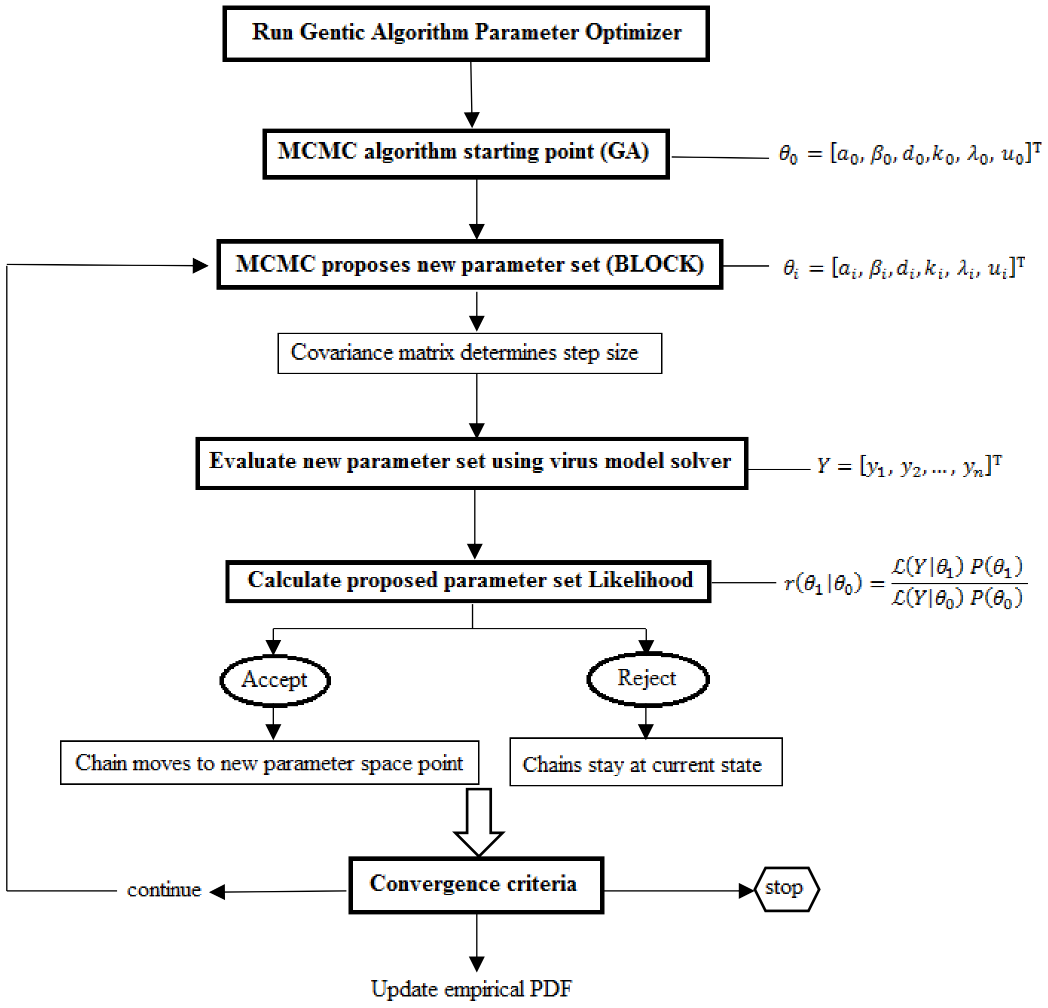
Run Gentic Algorithm Parameter Optimizer
MCMC algorithm starting point (GA)
$\theta_0 = [a_0, \beta_0, d_0, k_0, \lambda_0, u_0]^T$
MCMC proposes new parameter set (BLOCK)
$\theta_i = [a_i, \beta_i, d_i, k_i, \lambda_i, u_i]^T$
Covariance matrix determines step size
Evaluate new parameter set using virus model solver
$Y = [y_1, y_2, \dots, y_n]^T$
Calculate proposed parameter set Likelihood
$r(\theta_1|\theta_0) = \frac{\mathcal{L}(Y|\theta_1)\, P(\theta_1)}{\mathcal{L}(Y|\theta_0)\, P(\theta_0)}$
Accept
Reject
Chain moves to new parameter space point
Chains stay at current state
Convergence criteria
continue
stop
Update empirical PDF

Figure 4-3 MCMC block diagram

Chapter 5

Results and Discussion

The focus of this study was to test the results of a Bayesian calibration tool against a maximum likelihood analysis currently used in PSE's gPROMS ModelBuilder® platform. The virus model was first tested using this parameter optimization option within the gPROMS interface to compare the results obtained from a point estimate technique.

5.1 gPROMS ModelBuilder® Parameter Estimation

PSE's advanced process modeling tool gPROMS is a commercially available software commonly used to model industrial/chemical processes. Within the gPROMS language environment, the user has the option of running a gradient-based parameter estimation for a given mathematical model with experimental data. This approach uses a maximum likelihood estimation to provide a point estimate that minimizes the observation error (see Figure 5-1). However, this method of estimating model parameters has the disadvantage of easily converging to a local minimum, even if a relatively good starting point or guess is used. The results from a gPROMS virus model file are shown in Table 5-1 for estimating all six rate parameters. Notice that some parameters are hitting their bounds. These results are not representative of the final MCMC calibration model as this was an initial test of a gPROMS parameter estimation capability for the virus model.

Table 5-1 gPROMS ModelBuilder® parameter estimation results

Parameter	Initial Guess	gPROMS MLE	True Values
a	0.20	0.19	0.50
$\log_{10}\beta$	-7	-6.48	-6.7
d	0.05	0.046	0.10
k	80	1000	100
$\log_{10}\lambda$	4	2	5
u	5	100	5

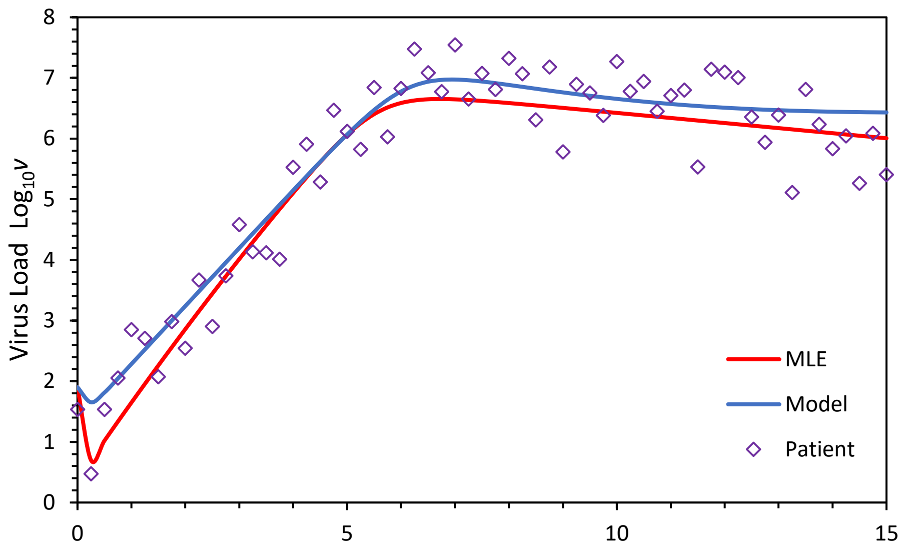

Figure 5-1 gPROMS MLE parameter result

5.2 Genetic Algorithm Starting Points

The procedure followed in this study for starting the Bayesian calibration is to initially run an optimizing genetic algorithm such that a relatively good first set of parameters is found for starting the MCMC. The algorithm is designed to give a set of model parameters that provide a dynamic virus behavior that closely matches the noisy experimental data. The randomness implied in this process and the high dimensionality of the problem mean that each GA optimization produces different starting points every time it is executed. The results obtained clearly show the complexity of this six-parameter model and how several combinations of the input parameters can produce similar dynamic response.

Figure 5-2 shows the evolution of a GA run with the blue line representing the performance index (PI) of the best individual in the current population while the dashed purple line represents the average performance index of the population. By using an elitist approach in the algorithm, the best solution in each iteration is guaranteed to survive and continue for at least the next future generation, until a better solution is found. This is observed in the performance index GA plot, as the best individual performance index is always increasing. For this particular run, the algorithm found a solution with a performance index close to 0.9 that indicates that the error fitness function is relatively close to the maximum optimal value of one. As a result, the closeness of the best individual to the noisy data is fairly decent.

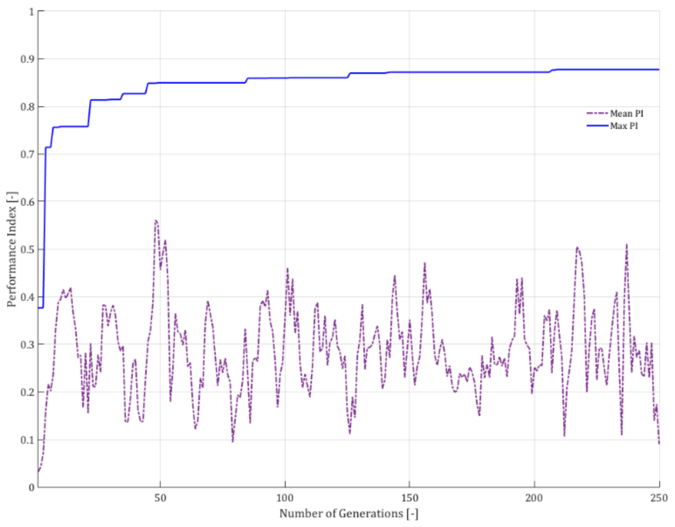

Figure 5-2 GA performance index for each generation

The algorithm's randomly drawn initial population has few individuals that obtain a good performance index, Figure 5-3(a), but as the generations evolve, through crossover and mutation operations better solutions are found, Figure 5-3(b), until reaching a final population, Figure 5-3(c), that should contain at least one individual with a relatively high performance index.

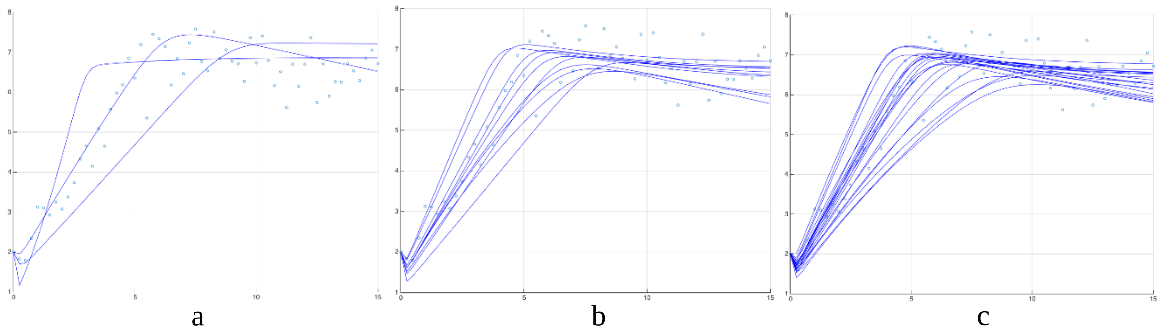

Figure 5-3 GA best individual solutions for initial, intermediate, and final populations

The following table gives the parameter set results for five selected GA simulations that are consequently used as starting points for the calibration. Each GA parameter optimization run uses a randomly seeded preliminary population with equal probability across the parameter bounds. Since the true values used to generate the model are know, it is important to note that the GA provides parameter optimization solutions that are relatively close to the desired values. However, as will be seen after the Bayesian calibration, this point estimate results do not guarantee that the MCMC will remain within these parameter regions. Instead, the MCMC will explore neighboring points, and occasionally generate posterior densities in regions different from the initial starting points.

Table 5-2 Selected GA parameter starting points

Parameter	True Value	$GA_{\#1}$	$GA_{\#2}$	$GA_{\#3}$	$GA_{\#4}$	$GA_{\#5}$	Mean	S.D.
a	0.50	0.20	0.42	0.322	0.19	0.44	0.31	0.11
$\log_{10}\beta$	-6.70	-7	-6.53	-6.55	-6.41	-6.61	-6.62	0.20
d	0.10	0.05	0.204	0.065	0.27	0.17	0.152	0.08
k*	100	100	100	100	100	100	100	0
$\log_{10}\lambda$	5	4	5.15	4.65	4.99	5	4.76	0.41
u	5	15	7.05	7.9	8.77	5.35	8.81	3.01

** The virus generation rate k is fixed at its actual value for calibration*

The parameter sets obtained from the GA show the variation of the potential solutions. Although the algorithm converges to values that are within one order of magnitude of the true values, the noise in the data allows for many possible combinations of the parameters that will have similar fitness function values, and thus similar calculated overall error.

5.3 MCMC Results

The virus model calibration was initially designed to have the gPROMS ModelBuilder® as a model solver and use a Foreign Process Interface (FPI) to communicate with the C++ based MCMC developed tool using input/output text files. Although this configuration was successfully setup, the repetitive use of the virus model solver for every Monte Carlo step made this interface very inefficient for calibration purposes. The main disadvantage of this setup was the required authentication of a software license request to execute the ModelBuilder® solver file using an in-built gPROMS command: gO:RUN.

The initial FPI results are not shown in this study due to the limited sample sizes that were successfully obtained. Instead, the C++ Crank-Nicholson numerical solver was subsequently implemented as mentioned in Chapter 2. This new solver implementation reduced the MCMC computation time in half, from roughly 1 hour for every one thousand steps with gPROMS to approximately 30 minutes. Afterwards, the Gauss-Seidel algorithm further reduced the computation time by 1/10 and ultimately provided 1000 MCMC steps every 3 minutes. This faster model computation resulted in the MCMC routine being able to obtain an empirical Posterior distribution with 300,000 steps (excluding burn-in) in less than one day.

After running many calibrations with varied starting points, the results obtained indicated multiple MCMC outcomes. The following plots show the parameter chain results for some of the selected starting points in Table 5-2 that prove the different parameter coverage.

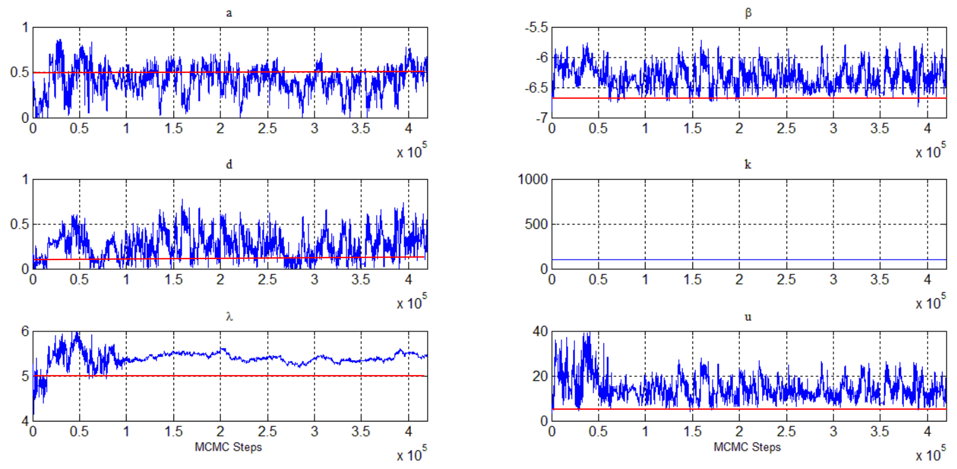

Figure 5-4 MCMC parameter chains for GA$_{\#5}$

Figure 5-4 shows the result for the parameter chains (blue) for the five calibrated parameters; the virus reproduction rate (k) is a fixed parameter, at its reference value. This is observed by the fixed line which shows no change in the parameter value as the MCMC progresses. The red line corresponds to the actual parameter values. For this particular calibration result, the uninfected cell generation rate λ is the only parameter that is not covered by the chain after burn-in. As was mentioned by Stafford *et al.* [6], this parameter is not possible to estimate when only using measured virus data for parameter estimation and is usually expressed in terms of the initial cell population count x_0 and the uninfected cell death rate d.

$$x_0 = \frac{\lambda}{d} \tag{47}$$

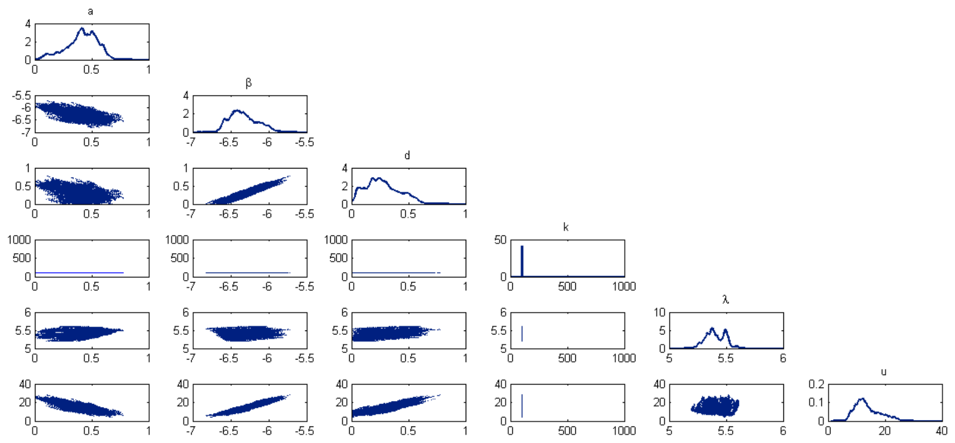

Figure 5-5 MCMC variate plots with parameter PDF for $GA_{\#5}$

As previously discussed, the batch means test for chain convergence was used as a criterion to statistically indicate when the MCMC had obtained sufficient samples from the Posterior distribution. Table 5-3 shows the batch means test results for the MCMC starting points selected from the GA optimization routine given in Table 5-2.

Table 5-3 Parameter batch means for converged calibrations

Mean (μ)	**$GA_{\#1}$**	**$GA_{\#2}$**	**$GA_{\#3}$**	**$GA_{\#4}$**	**$GA_{\#5}$**	**Mean**	**S.D.**
μ_a	0.801	0.696	0.529	0.107	0.414	0.509	0.24
μ_β	-6.56	-6.42	-6.46	-6.57	-6.33	-6.47	0.09
μ_d	0.417	0.332	0.178	0.109	0.272	0.262	0.11
μ_k*	100	100	100	100	100	100	0
μ_λ	5.30	5.56	5.40	3.32	5.41	4.998	0.84
μ_u	4.59	10.56	11.55	6.99	13.72	9.48	3.27

** The virus generation rate k is fixed at its actual value for calibration*

The batch means values (μ) are a measure of each parameter's population mean for the selected number of bins or batches from the total number of MCMC step samples. Each of the parameters in Table 5-3 had a confidence interval of 95% or higher.

Lastly, contour plots (see Figure 5-6) are also plotted for the five calibrated parameters using a heatmap color bar to reveal the 2D bivariate regions of higher density, that are not easily identified from the monochromatic scatter plots.

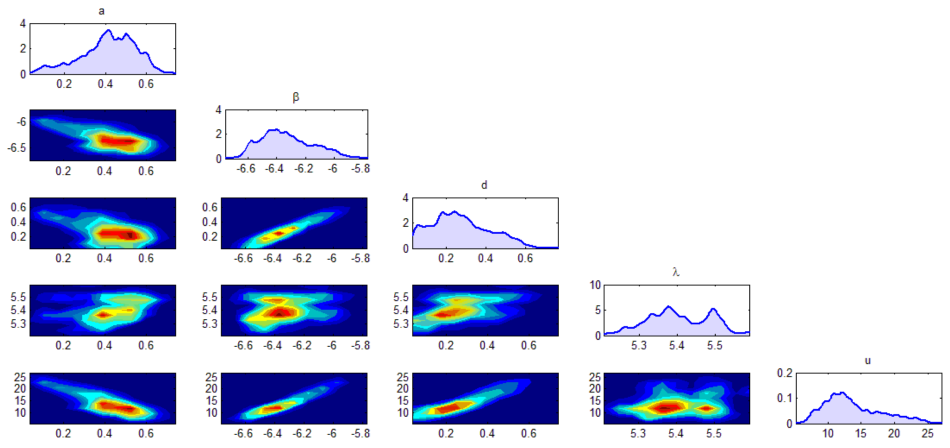

Figure 5-6 Bivariate contour plots for GA$_{\#5}$

5.4 Model Calibration and Predictions

In the previous section, the marginal probabilities were plotted to show the empirical Posterior densities. These PDFs are samples from the Posterior distribution and as such are used to randomly draw sets of parameters that show the coverage of the calibrated model compared to the noisy experimental data. Figure 5-7 shows the coverage of the GA$_{\#5}$ data set and the actual model used to generate the data that represents what the calibration should obtain when no noise is present.

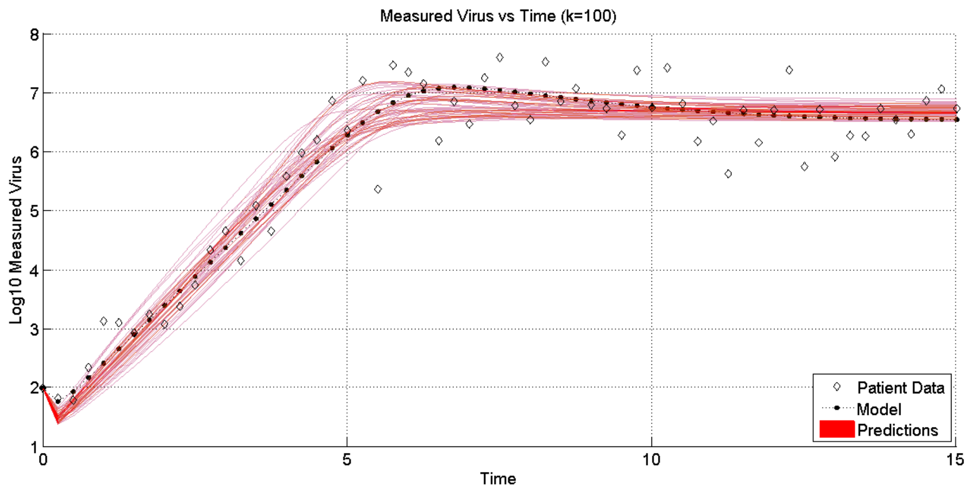

Figure 5-7 Calibrated model coverage over the experimental data

Figure 5-7 shows 50 randomly selected parameter sets from the calibration Posterior. This calibrated model is compared to the noisy patient data and additionally shows the actual model used to generate the data. In order to test the calibrated model prediction capabilities for different virus generation rates k, these randomly picked parameter sets are ran through the solver for 2 different user-input values of fixed k, larger and smaller than the true value used for calibration, to analyze the extrapolation capabilities.

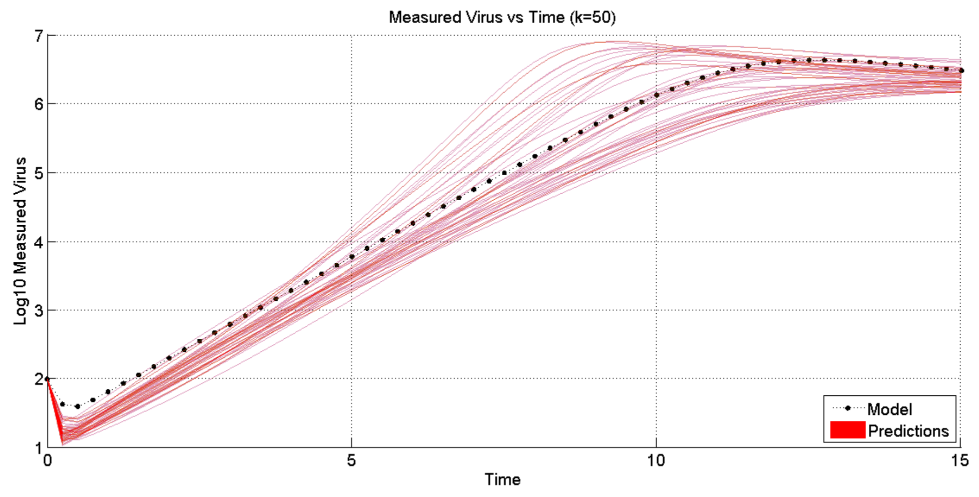

Figure 5-8 Calibrated model predictions for $k = 50$

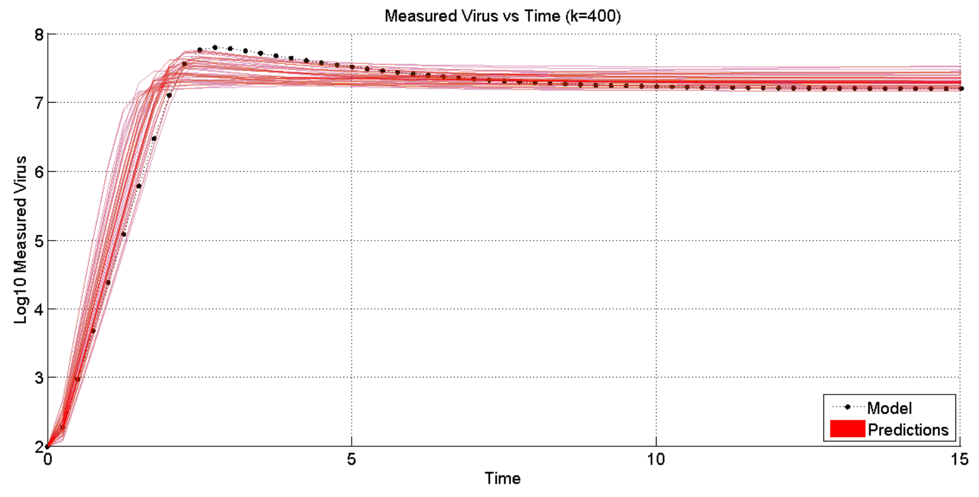

Figure 5-9 Calibrated model predictions for $k = 400$

Figure 5-8 and Figure 5-9 illustrate the coverage obtained from the calibrated model's predictions for lower and higher values of the virus generation rate parameter, respectively. It is seen that by decreasing the virus generation rate, the prediction plots have higher uncertainty, while the increased parameter plots show narrower variance with some missed points at the peak. In addition, continuously reducing the virus generation rate led to unstable predictions (see Figure 5-10), as the virus balance resulted in an unrealistic suppression of the virus that is not expected unless some form of drug therapy is used, and this would require a different model. These results show that the calibrated model can only be used to make predictions within a certain range, which can be defined by the reproductive ratio of infection as $R_0 > 1$. For the case of HIV, as mentioned in Chapter 2, $R_0 = 8$ for the reference parameter values used.

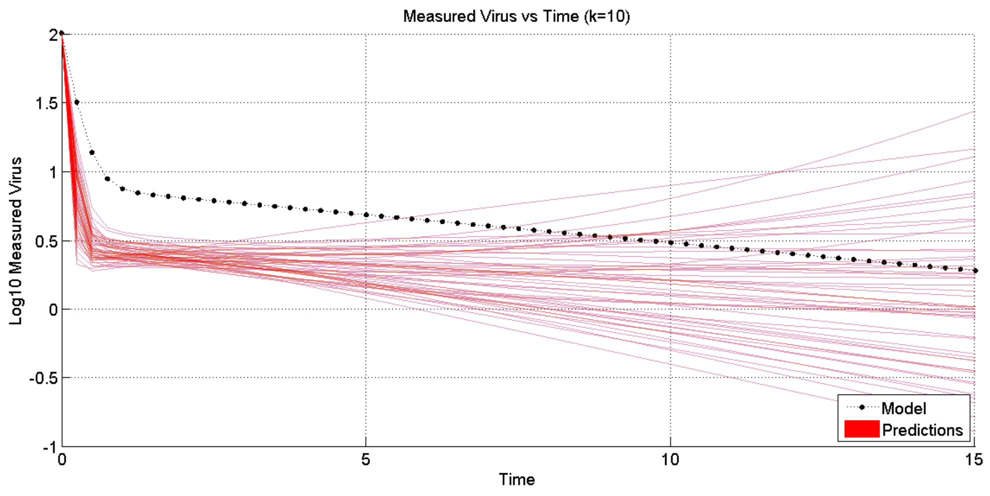

Figure 5-10 Unstable MCMC calibration predictions for $k = 10$

The parameter bivariate contour plots (Figure 5-11) are generated for two additional starting points from Table 5-2 (GA$_{\#1}$ and GA$_{\#2}$) to determine the parameter coverage and calibration/predictions of different MCMC runs. These results show how the batch means criteria converged calibrations reach different Posterior distributions.

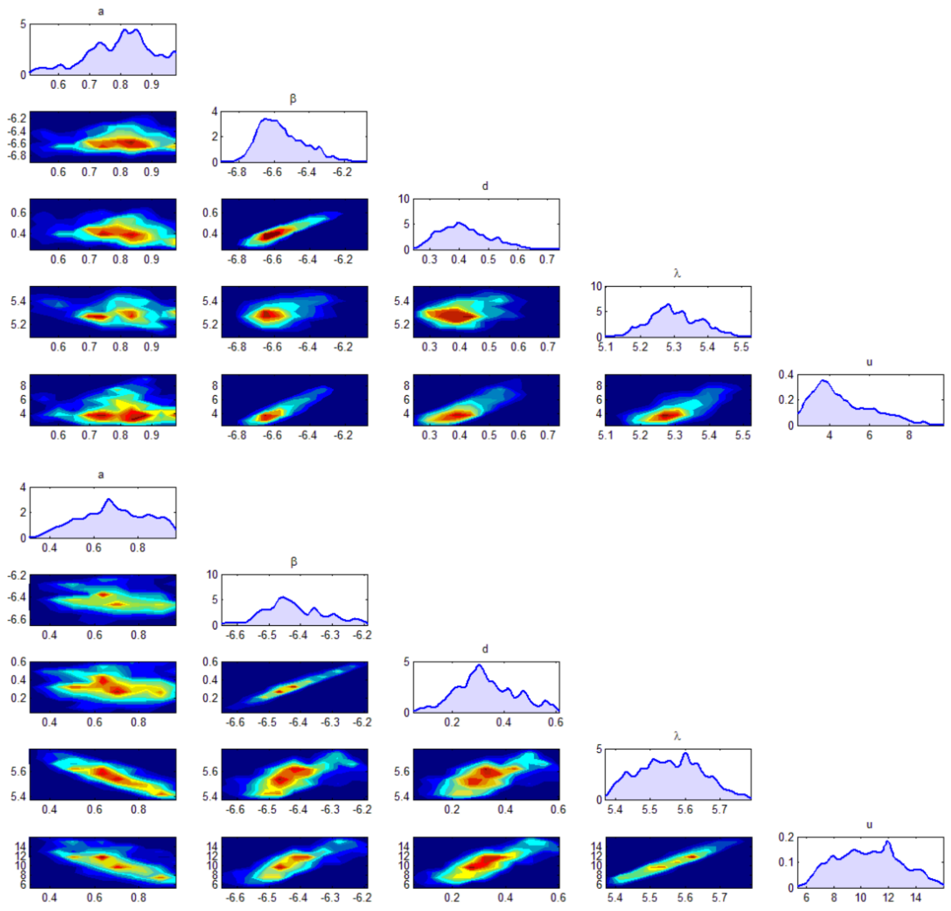

Figure 5-11 Bivariate contour plots for $GA_{\#1}$ (top) and $GA_{\#2}$ (bottom)

The previous plots show that each MCMC run reaches a significantly different empirical posterior pdf. Although this is not the desired convergence behavior that was initially anticipated, the following calibration (Figure 5-12) and prediction (Figure 5-13 and Figure 5-14) plots demonstrate that the distributions obtained are nonetheless useful for this virus model problem.

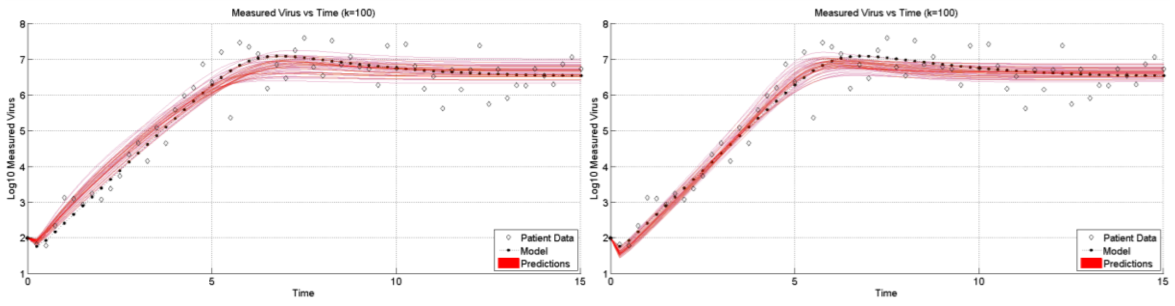

Figure 5-12 MCMC calibrated model coverage for $GA_{\#1}$ (left) and $GA_{\#2}$ (right)

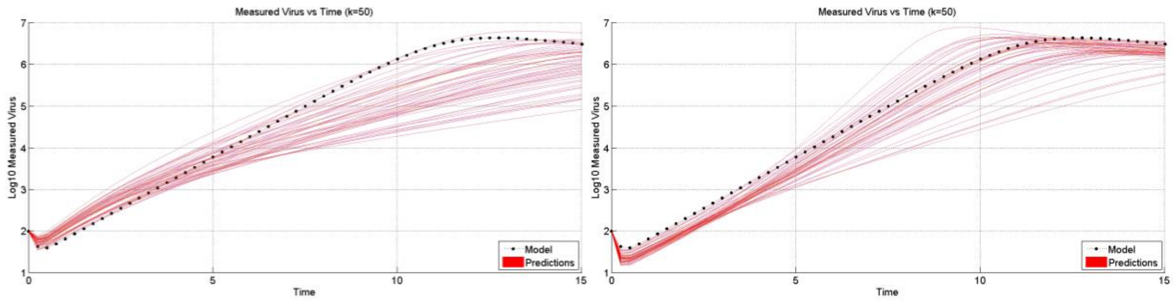

Figure 5-13 Calibrated model predictions for GA$_{\#1}$ (left) and GA$_{\#2}$ (right) with $k = 50$

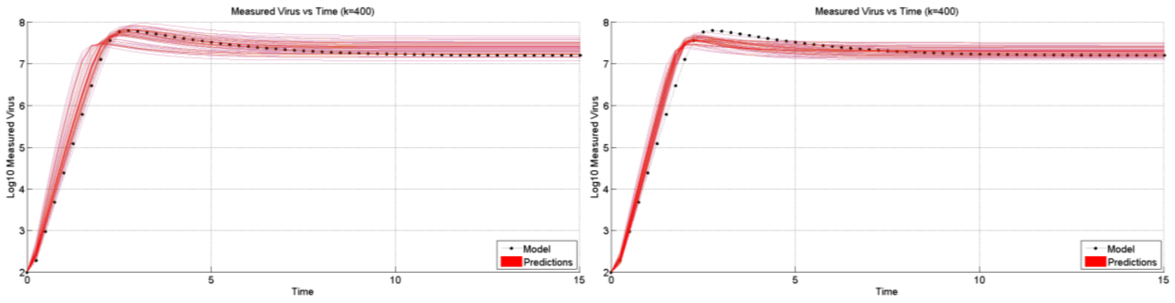

Figure 5-14 Calibrated model predictions for GA$_{\#1}$ (left) and GA$_{\#2}$ (right) with $k = 400$

The results for repeatedly different MCMC runs indicate that obtaining a statistically comparable (converged) empirical Posterior for every run was not possible with the configuration used. Figure 5-15 has the different Posterior PDFs obtained from the previous results, overlapped to show the areas where each MCMC run covered the same parameter space. The noise variance, parameter identifiability, and limited experimental data-type used (only measured virus, as opposed to including infected/uninfected cell counts) can be potential reasons for not being able to always obtain a fully converged Posterior. These results open the doorway to new opportunities to test the MCMC, using different configurations that may result in better Posterior convergence. Nonetheless, the calibrated model predictions obtained in this study show that Bayesian analysis can be effectively used to study multi-parameter dynamic models, such as this primary HIV model, and obtain useful predictions.

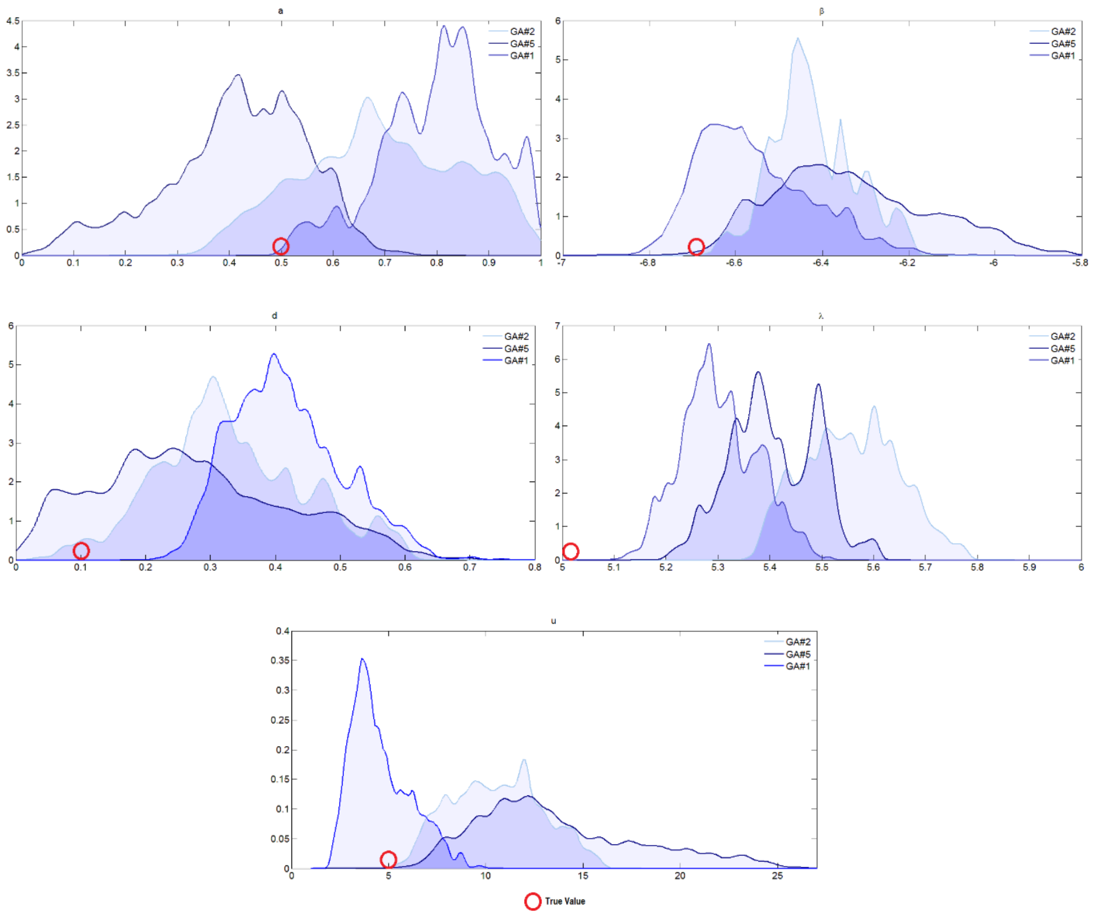

Figure 5-15 Overlapped parameter Posterior PDFs for GA$_{\#1}$, GA$_{\#2}$, and GA$_{\#5}$

Chapter 6

Conclusions

The MCMC calibration results demonstrate that Bayesian analysis is a promising alternative to conventional parameter estimation techniques that can be structured for multidimensional problems. Nevertheless, the results also indicate that the MCMC Bayesian calibration tool is difficult to completely automate and, in general, requires some level of adequate user input and experience to properly tune the simulation parameters.

As was mentioned in earlier studies, the virus model’s parameter identifiability, although mathematically possible, is not guaranteed when only measured virus data is used. This statement was confirmed in this study, as coverage of all parameters in a single MCMC run was rarely obtained, even when using starting points close to the true values. Imposing certain parameter restrictions and assumptions such as a time invariant uninfected cell replenishment rate can lead to fixed parameters for this problem and a potentially simpler parameter estimation problem. This assumption is not easily implemented as only measured virus data is used. An alternative solution is to possibly repeat a virus model MCMC calibration run using additional data from the uninfected CD4 cell population.

Although the true parameter values do not fall into regions of high marginal probability posterior density, the predictions from the calibrated model show coverage of the actual dynamic behavior for ranges outside of the calibration data, as was observed for a fixed virus generation rate parameter. Another problem occasionally encountered during the different calibrations was poor chain movement/coverage, most probably caused by a narrow parameter proposal distribution from a fixed covariance matrix. This issue could be solved by resetting the covariance matrix at the current MCMC step, but this has the unwanted consequence of requiring the user to closely monitor the chain’s progression to detect chain stagnation or flatline behavior.

From this study, it can be concluded that Bayesian techniques can be successfully and usefully applied for modeling problems concerned with analyzing and calibrating noisy data sets. Additionally, Bayesian techniques offer an alternative solution to conventional gradient-based estimation methods that cannot accurately fit noisy data, at the cost of more computational calibration time. Because computation time is a main concern and usually a key disadvantage of Bayesian calibration techniques, some possible solutions include MCMC parallelization and multiple moving chain algorithms that can simultaneously make several random walks from different starting points.

From the results obtained in this thesis, some final comments and suggestions for future research are: as mentioned in other studies, using not only measured virus data, but also target cell population counts allows for better identification of the model parameters. A comparison of calibration results using both data sets may allow the MCMC to get better coverage of the true parameter values. Additionally, varying the measurement time sampling can account for more realistic infected patient HIV data. In actual patient measurements, the measured virus is sampled more frequently during primary infection to cover the infection's characteristic peak behavior. Conversely, the initial data is rarely sampled and similarly the asymptotic phase offers almost no useful parameter identifiability. An interesting comparison is to rerun the MCMC with patient data only for the measured virus peak with a higher sampling frequency and also changing the variance in the added noise to have higher/lower experimental measurement variance.

References

[1] M. C. Kennedy and A. O'Hagan, "Bayesian calibration of computer models," *Journal of the Royal Statistical Society,* vol. 63, no. 3, pp. 425-464, 2001.

[2] A. B. Gumel, P. N. Shivakumar and B. M. Sahai, "A mathematical model for the dynamics of HIV-1 during the typical course of infection," *Nonlinear Analysis,* vol. 47, no. 1, pp. 1773-1783, 2001.

[3] M. A. Nowak and R. M. May, Virus Dynamics: Mathematical Principles of Immunology and Virology, UK: Oxford University Press, Inc, 2000.

[4] J. M. Murray, G. Kaufmann, A. D. Kelleher and D. A. Cooper, "A model of primary HIV-1 infection," *Mathematical Biosciences,* vol. 154, no. 1, pp. 57-85, 1998.

[5] H. C. Tuckwell and E. Le Corfec, "A Stoshastic Model for Early HIV-1 Population Dynamics," *J. of Theoretical Biology,* vol. 195, no. 1, pp. 451-463, 1998.

[6] M. A. Stafford, L. Corey, Y. Cao, E. S. Daar, D. D. Ho and A. S. Perelson, "Modeling Plasma Virus Concentration during Primary HIV Infection," *Journal of Theoretical Biology,* vol. 203, no. 3, pp. 285-301, 200.

[7] M. Ciupe, B. Bivort, D. Bortz and P. Nelson, "Estimating kinetic parameters from HIV primary infection data through the eyes of three different mathematical models," *Mathematical Biosciences,* vol. 200, no. 1, pp. 1-27, 2006.

[8] P. D. Leenheer and H. L. Smith, "Virus Dynamics: A Global Analysis," *SIAM Journal on Applied Mathematics,* vol. 63, no. 4, pp. 1313-1327, 2003.

[9] X. Xia y C. H. Moog, «Identifiability of Nonlinear Systems With Application to HIV/AIDS Models,» *IEEE Transactions on Automatic Control,* vol. 48, nº 2, pp. 330-336, 2003.

[10] M. T. Wentworth, R. C. Smith and H. T. Banks, "Parameter Selection and Verification Techniques Based on Global Sensitivity Analysis Illustrated for an HIV Model," *SIAM/ASA Journal on Uncertainty Quantification,* vol. 4, no. 1, pp. 266-297, 2016.

[11] H. Wu, H. Zhu, H. Miao and A. S. Perelson, "Parameter Identifiability and Estimation of HIV/AIDS Dynamic Models," *Bulletin of Mathematical Biology,* vol. 70, no. 3, pp. 785-799, 2008.

[12] D. Burg, L. Rong, A. U. Neumann and H. Dahari, "Mathematical modeling of viral kinetics under immune control during primary HIV-1 infection," *Journal of Theoretical Biology,* vol. 259, pp. 751-759, 2009.

[13] K. Soetaert and T. Petzoldt, "Inverse Modelling, Sensitivity and Monte Carlo Analysis in R Using Package FME," *Journal of Statistical Software,* vol. 33, no. 3, 2010.

[14] R. Hogg and J. McKean, Introduction to Mathematical Statistics, New York: Pearson, 2005.

[15] A. Gelman, J. B. Carlin, H. S. Stern and D. B. Rubin, Bayesian Data Analysis, Chapman & Hall/CRC, 2003.

[16] R. L. Burden and J. D. Faires, Numerical Analysis, Boston: Cengage Learning, Inc, 2001.

[17] A. S. Perelson, "Modelling viral and immune system dynamics," *Nature Reviews Immunology,* vol. 2, pp. 28-36, 2002.

[18] K. A. Pawelek, S. Liu, F. Pahlevani and L. Rong, "A model of HIV-1 infection with two time delays: Mathematical analysis and comparison with patient data," *Mathematical Biosciences,* vol. 235, no. 1, pp. 98-109, 2012.

[19] M. A. Nowak and C. R. M. Bangham, "Population Dynamics of Immune Responses to Persistent Viruses," *JSTOR,* vol. 272, no. 5258, pp. 74-79, 1996.

[20] A. S. Perelson, D. E. Kirschner and R. De Boer, "Dynamics of HIV infection of CD4+ T cells," *Mathematical Biosciences,* vol. 114, no. 1, pp. 81-125, 1993.

[21] N. Lange, B. P. Carlin and A. E. Gelfand, "Hierarchical Bayes Models for the Progression of HIV Infection Using Longitudinal CD4 T-Cell Numbers," *Journal of the American Statistical Association,* vol. 87, no. 419, pp. 615-626, 1992.

[22] R. Luo, M. J. Piovoso, J. Martinez-Picado and R. Zurakowski, "HIV Model Parameter Estimates from Interruption Trial Data including Drug Efficacy and Reservoir Dynamics," *PLoS ONE,* vol. 7, no. 7, 2012.

[23] H. Miao, X. Xia, A. S. Perelson and H. Wu, "On Identifiability of Nonlinear ODE Models and Applications in Viral Dynamics," *SIAM Review,* vol. 53, no. 1, pp. 3-39, 2011.

[24] X. Xia, "Estimation of HIV/AIDS parameters," *Automatica,* vol. 39, no. 11, pp. 1983-1988, 2003.

[25] D. E. Kirschner, "Using Mathematics to Understand HIV Immune Dynamics," *Notices of the American Mathematical Society,* vol. 43, no. 2, 1996.

[26] A. M. Jeffrey and X. Xia, "Identifiability of HIV/AIDS Models," in *Deterministic and Stochastic Models of AIDS Epidemics and HIV Infections*, World Scientific, 2005, pp. 255-286.

[27] M. T. Wentworth, "Verification Techniques for Parameter Selection and Bayesian Model Calibration Presented for an HIV Model," ProQuest, Ann Arbor, 2017.

[28] A. F. M. Smith and G. O. Roberts, "Bayesian Computation Via the Gibbs Sampler and Related Markov Chain Monte Carlo Methods," *Journal of the Royal Statistical Society,* vol. 55, no. 1, pp. 3-23, 1993.

[29] H. Miao, C. Dykes, L. M. Demeter, J. Cavenaugh, S. Y. Park, A. S. Perelson and H. Wu, "Modeling and Estimation of Kinetic Parameters and Replicative Fitness of HIV-1 from Flow-Cytometry-Based Growth Competition Experiments," *Bulletin of Mathematical Biology,* vol. 70, no. 1, pp. 1749-1771, 2008.