Stochastic Computer Model Calibration and Uncertainty Quantification

Arindam Fadikar

Dissertation submitted to the Faculty of the
Virginia Polytechnic Institute and State University
in partial fulfillment of the requirements for the degree of

Doctor of Philosophy
in
Statistics

David Higdon, Chair
Robert Gramacy
Henning Mortveit
Madhav Marathe

June 7, 2019
Blacksburg, Virginia

Keywords: Computer model, gaussian process, sensitivity analysis, epidemiology, bayesian estimation, mcmc

Stochastic Computer Model Calibration and Uncertainty Quantification

Arindam Fadikar

(ABSTRACT)

This dissertation presents novel methodologies in the field of stochastic computer model calibration and uncertainty quantification. Simulation models are widely used in studying physical systems, which are often represented by a set of mathematical equations. Inference on true physical system (unobserved or partially observed) is drawn based on the observations from corresponding computer simulation model. These computer models are calibrated based on limited ground truth observations in order produce realistic predictions and associated uncertainties. Stochastic computer model differs from traditional computer model in the sense that repeated execution results in different outcomes from a stochastic simulation. This additional uncertainty in the simulation model requires to be handled accordingly in any calibration set up.

Gaussian process (GP) emulator replaces the actual computer simulation when it is expensive to run and the budget is limited. However, traditional GP interpolator models the mean and/or variance of the simulation output as function of input. For a simulation where marginal gaussianity assumption is not appropriate, it does not suffice to emulate only the mean and/or variance. We present two different approaches addressing the non-gaussianity behavior of an emulator, by (1) incorporating quantile regression in GP for multivariate output, (2) approximating using finite mixture of gaussians. These emulators are also used to calibrate and make forward predictions in the context of an Agent Based disease model which models the Ebola epidemic outbreak in 2014 in West Africa.

The third approach employs a sequential scheme which periodically updates the uncertainty in the computer model input as data becomes available in an online fashion. Unlike other two methods which use an emulator in place of the actual simulation, the sequential approach relies on repeated run of the actual, potentially expensive simulation.

Stochastic Computer Model Calibration and Uncertainty Quantification

Arindam Fadikar

(GENERAL AUDIENCE ABSTRACT)

Mathematical models are versatile and often provide accurate description of physical events. Scientific models are used to study such events in order to gain understanding of the true underlying system. These models are often complex in nature and requires advance algorithms to solve their governing equations. Outputs from these models depend on external information (also called model input) supplied by the user. Model inputs may or may not have a physical meaning, and can sometimes be only specific to the scientific model. More often than not, optimal values of these inputs are unknown and need to be estimated from few actual observations. This process is known as inverse problem, i.e. inferring the input from the output. The inverse problem becomes challenging when the mathematical model is stochastic in nature, i.e., multiple execution of the model result in different outcome. In this dissertation, three methodologies are proposed that talk about the calibration and prediction of a stochastic disease simulation model which simulates contagion of an infectious disease through human-human contact. The motivating examples are taken from the Ebola epidemic in West Africa in 2014 and seasonal flu in New York City in USA.

Dedication

To all my pet companions.

Acknowledgments

A good number of individuals and animals have been a significant part of my journey during last five years at Virginia Tech, whom I would sincerely like to acknowledge, to the best of my ability.

I feel myself immensely fortunate to have Dave Higdon as my academic advisor. Working and interacting with Dave has always been a fun and learning experience that I can thrive on for many coming years in my life. His kindness, patience and willingness in explaining and answering my numerous questions, on various topics, have cast a significant effect on my working principles; and his guidance will remain an exemplary one to me. On the same note, I would like to express my regards to Henning Mortveit, who has been much more than a mentor, who guided me during my challenging times, helped me in setting up my goals, and perhaps most importantly introduced me to Dave among many other things. I am thankful to Madhav Marathe for providing me this opportunity to work and grow in his lab – NDSSL, which adds a great value to my academic career without any doubt. His leadership, friendly conversations in-spite of being the busiest person in the lab, and useful insights have helped me in ways that have set a standard to me. Although I had limited interactions with Bobby Gramacy, his readiness in helping me with ideas and doubts was hugely beneficial, and I always admired his approach in tackling any task.

I could not have asked for a better cohort at NDSSL, from whom I have received timely advice, and help whenever I needed them. Discussions with Srini, Bryan, Chen, Anil have provided insights into my work, Eric and Maleq helped me with programing aspects, and interactions with fellow students resulted in new ideas. Thanks to Erin as well for making the lab an exciting and memorable workplace.

Debjit and Shibaji have not only been my roommates, they created a homely atmosphere, away from home. I will always value their selfless assistance during my hard days, both in work life and personal life. Speaking of friends, I am indebted to Gurukul for inspiring me everyday in becoming a better person. They are nothing less than a family to me, whom I trust more than myself.

My life in Blacksburg would be incomplete without Salt and Pepper (twin sisters mice) and two gerbils and a hamster. Enjoying the companionship of these wonderful little creatures and challenging myself with new ideas and wood projects for them, kept me well engaged in my personal life.

Last but not least, I would acknowledge them, without whom I would have never come this far, for whom perhaps no word can express my deepest respect and gratitude. My maths teacher during my high school, DC sir, it was him because of whom I decided to pursue maths after school. Saibal sir and Dibyendu sir pushed me into the new subject statistics; and Rahul Roy from Indian Statistical Institute, Delhi, introduced me to Madhav. Above all, it has always been my parents' countless sacrifice, utmost hardship and strong determination to provide me better education and better life that has driven me forward.

Contents

List of Figures **x**

1 Introduction **1**

1.1 Premises . . . 1

1.1.1 System, Model and Computer Simulation . . . 1

1.1.2 Why Simulation? . . . 2

1.1.3 Challenges . . . 3

1.2 Simulation and Statistical Inference . . . 5

1.2.1 Deterministic vs. Stochastic Simulation . . . 5

1.2.2 Verification and Validation . . . 6

1.2.3 Calibration . . . 7

1.2.4 Emulation . . . 8

1.2.5 Prediction and Uncertainty Quantification . . . 9

1.3 Disease Model . . . 10

1.3.1 Compartmental Disease Model . . . 10

1.3.2 Network Based Model . . . 14

2 Emulation and Calibration **23**

2.1 Related Work . . . 23

2.1.1 Bayesian Model Calibration . . . 25
2.1.2 Stochastic Computer Model Emulation . . . 28
2.2 Calibration via Quantile Based Emulation . . . 30
2.2.1 Univariate Response . . . 30
2.2.2 Multivariate Response . . . 34
2.2.3 Calibration . . . 44
2.2.4 Results . . . 47
2.3 Calibration and Emulation via Finite Mixture modeling . . . 54
2.3.1 Univariate Response . . . 55
2.3.2 Multivariate Response . . . 62
2.3.3 Calibration . . . 66

3 Sequential Calibration of Computer Model **73**

3.1 Related Work . . . 73
3.2 Patch Model Revisited . . . 74
3.3 Sequential Calibration (Temporal) . . . 77
3.3.1 Basic SMC on State-Space Models . . . 77
3.3.2 SMC for stochastic computer model . . . 79
3.4 Spatial Calibration . . . 87
3.4.1 Discussion . . . 90

4 Future Direction **93**

4.1 Mixture Density Emulator 93

4.1.1 Related Work 95

4.1.2 Proposed Framework 96

4.2 Discrepancy in Sequential Calibration 98

Bibliography **101**

List of Figures

1.1 The left frame shows the simulated time series of the fraction of total population in each of the three compartments from a simulated deterministic SIR model. The right frame shows the fraction of newly infected individuals at discrete time points. . 12

1.2 The left frame shows the simulated time series of the fraction of total population in each of the three compartments from a simulated deterministic SEIR model. The right frame shows the fraction of newly infected individuals at discrete time points. . 13

1.3 The left frame shows the simulated time series of the fraction of total population in each of the three compartments from a simulated deterministic SIR model, and the right frame shows simulation output from its stochastic counterpart. 14

1.4 The figure shows the simulated disease evolution on a school network [65] using an agent based SEIR model. The node colors green, yellow, red and grey denote susceptible, exposed, infected and recovered state respectively, whereas infection flows through the darker edges. 16

1.5 The figure shows output from a stochastic agent based SEIR simulation, ran 100 times at same configuration. 17

1.6 Datasets used to capture short-range and long-range mobility in United States (image courtesy: Srini Venkatramanan) 18

1.7 Spatio-temporal spread of influenza, for a sample scenario, seeded in Louisiana. Counties where the epidemic emerges by Day 7,30,90 or 180 are are respectively shaded blue, green, purple and orange.(image courtesy: Srini Venkatramanan) 21

1.8 A pictorial description of how synthetic population is generated. The process consists of generating (1) an anonymous population of individuals with detailed demographic and spatial information, (2) a mapping between every individual and their activity locations, and (3) a derived edge-weighted social contact network to be used for disease propagation. (image courtesy: Henning Mortveit) 22

2.1 The figure shows a OA-based LH design of $m = 100$ points in 2d projection. The three points denoted by A, B and C represents three inputs which will be used later for holdout experiment. 32

2.2 The grey lines in each box show the simulated epicurve for 57 weeks at a fixed input setting, where the simulation generates 100 replicates for each input setting from the design in Fig 2.1. The first 20 weeks of epidemic observations are shown in solid black lines – the same black line is shown in all 100 boxes. The three boxes marked by A, B and C correspond to the input locations held out for out of sample experiment. 33

2.3 The figure shows a set of 100 realizations of epidemic ABM simulations each at 5 different input settings. The estimates of the empirical quantiles $Q(\alpha)$ of the logged total epidemic counts at week 20 with $\alpha = .05, .275, .5, .725, .95.$ are shown in red dots. 34

2.4 Posterior mean function of the ABM simulator output conditioned on the univariate response from simulated ABM as shown by then red circles, as a function of input θ and quantile α. These red circles are the same red dots in Figure 2.3 35

2.5 The grey lines are the actual trajectories simulated from the ABM, showing the cumulative number of disease counts for 57 weeks at 3 different input locations $\boldsymbol{\theta}$. The five lines in five colors (magenta, cyan, red, green, blue respectively) show the $.05, .275, .5, .725, .95$ quantiles estimated from the grey lines. 36

2.6 Left: The ensemble of ABM output from the experimental design (Fig 2.1); right: 5 basis functions obtained from the ABM output after centering and standardizing. . . 38

2.7 The top row shows the simulations in the left frame and first and second principal component bases in the right frame, obtained from the simulation output, whereas, the the posterior mean surfaces are for basis loadings $w_i(\boldsymbol{\theta}), i = 1, 2$ with respect to θ_1 and θ_3 in bottom row. Here the other fours parameters were held at their nominal values as θ_1 and θ_3 vary over their predefined range. 41

2.8 The figure shows 2-D posterior mean surface of all 5 bases w_i with respect to two input components θ_1 and θ_3, as obtained from GP emulator. 42

2.9 Posterior distribution of the correlation parameters ρ_{wk} in the covariance function (Eq. 2.8 for 5 bases. 42

2.10 On the left most column, grey lines show the actual simulations and corresponding empirical quantiles at the pre-chosen inputs locations, denoted by the letters A, B and C and the colors from the design in Fig 2.1. The next 5 columns show pointwise 90% credible intervals of the posterior emulator prediction of quantiles at those model inputs. These ABM runs were held out from the training dataset used to train the GP emulator. Each of these five columns compares the empirical quantiles, denoted by different colors, from the actual simulations with the posterior predictions. 43

2.11 Each frame shows the posterior mean predictions (log counts as a function of time) from the GP emulator varying one input across its prior range, while holding others at their posterior mean values. 44

2.12 Discrepancy kernel δ_k between week 1 and week 20 (observation period). 45

2.13 The figure shows estimated posterior distribution of the input parameters $(\theta_1, \cdots, \theta_5, \alpha)$. The diagonal shows the estimated marginal posterior pdf for each parameter; and the off-diagonal images give estimates of bivariate marginals; to contour lines show estimated 90% hpd regions. 48

2.14 The figure shows posterior prediction decomposition into emulator and discrepancy. Left: the posterior 90% uncertainty for the calibrated emulator $\eta(\boldsymbol{\theta})$ prediction is shown. The aggregated uncertainty is due to unknown $\boldsymbol{\theta}$ and the GP emulator of ABM. Middle: posterior 90% uncertainty for δ. Right: posterior 90% intervals for the actual epidemic curve: $\eta(\boldsymbol{\theta}) + \delta$. The blue dots show the log of observed counts, along with 1 standard deviation bars given by the square root of $\text{diag}(\Sigma_y)$. 49

2.15 Left: Posterior and prior realizations of the epidemic curves are shown in grey and cyan respectively. The dashed lines represents the pointwise 90% credible intervals from the posterior predictions. Middle: 40 posterior realizations of the weekly counts over time. The blue dots show the peak weekly count and corresponding time. Right: histogram of posterior peak time realizations. 50

2.16 Top: Posterior realizations (cyan lines) and pointwise 90% credible intervals (dashed lines) for the actual epidemic curves $(\eta(\theta) + \delta)$ for each of the ebola challenge time periods. The gray lines show epidemic curves produced from the initial ABM ensemble. Bottom: The corresponding pointwise 90% credible intervals of the discrepancy (δ) for each of the time periods. The more substantial discrepancy (from the analysis using data up to week 42) adjusts for simulations that produce more late-time cases than the observations suggest 51

2.17 Posterior distributions for three functions of the actual epidemic curve – peak timing (left), peak weakly cases (middle), and total epidemic size (right). Each row shows the posterior density estimates conditional on epidemic data up to 13, 26, 35 and 42 weeks, moving from top to bottom. 52

2.18 The grey lines show the simulated trajectories of the cumulative number of disease incidence for 57 weeks at three input setting. 54

2.19 The figure shows the histogram of total infections at week 57 obtained from simulations at 100 unique input settings, each with 100 replicates. 56

2.20 The figure shows the grey histograms of total infections at week 57 obtained from simulations at 3 input settings, each with 100 replicates (as used in Fig 2.18), and the estimated densities from gaussian mixture model is shown in red. 57

2.21 The figure shows the kernel density estimate of the all simulation output (same as the histogram in Fig 2.19), along with the actual outputs at the bottom in two colors. The colors characterizes the cluster memberships of the simulation output (high or low regime) 60

2.22 The figure shows true distribution (according to gaussian mixture model) of the simulation at three different input locations (same as Fig 2.18, 2.20) in blue and the predictive distribution using the combined emulator in Eq. 2.13 is shown in red. The gray histrograms in the background are correspond to the actual simulation output. 61

2.23 The figure shows the same set of simulations as in Fig 2.20 classified into two clusters (black and red) using the same gaussian mixture model, and the estimated cluster membership probability $\pi(\boldsymbol{\theta})$ is shown at top left corner in each frame. The unnormalized conditional distribution is also shown vertically in grey dashed line. . . . 63

2.24 The solid lines, black and red respectively, show the in sample mean prediction from two GPs, and dotted lines represent corresponding 90% predicted CI. The fractions at top left of each frame denotes the estimated probability of simulator output being in the cluster denoted by black (higher regime). The opacity is also consistent with this estimated probability. Grey lines in the background are the actual simulation output. 65

2.25 The same emulator prediction as in Fig 2.24, except they are result from leave one out exercise. 66

2.26 Predictive mean simulator predictions (log counts as a function of time) varying one input across its prior range, holding others at their nominal values 67

2.27 Each frame of the figure shows the observed data in blue, and the mean prediction in solid black dots, and 90% confidence interval in dashed black line. 69

2.28 The left figure shows the boxplots of predicted total infection at each of the four scenarios from Fig 2.27. The middle frame corresponds to peak total infections, and the right frame shows peak week. 70

3.1 The left frame shows 5 boroughs that New York city consists of, and the right matrix plot shows proportion of population traveling from a borough on the row to a borough on the column. 75

3.2 An output from flu simulation on New York city patch model. 76

3.3 Simulated output from flu simulation on New York city patch model, with 1-sd error bars. 82

3.4 Histogram of posterior samples of calibration parameter $\boldsymbol{\theta}$ (left), prior and posterior simulation output (in light and dark grey respectively) along with the simulated data. 83

3.5 Posterior distribution of $\boldsymbol{\theta}$ from algorithm 1 (left) and algorithm 1 + 2 (right). . . . 84

3.6 Posterior samples from calibrated simulator from algorithm 1 (left) and algorithm 1 + 2 (right). 85

3.7 Histogram of posterior conditioned on 20 weeks of data (left) and calibrated prediction in green (right). 86

3.8 Boxplot of two posteriors $p(\boldsymbol{\theta}|\boldsymbol{y}^{(1)})$ and $p(\boldsymbol{\theta}|\boldsymbol{y}^{(1)}, \boldsymbol{y}^{(2)})$. Red star shows the actual value of $\boldsymbol{\theta}$. 87

3.9 Patch model output from flu simulation in five boroughs of New York city, with 1-sd error bars. 88

3.10 Posterior samples from calibrated spatial simulator until week 10 from algorithm 1 (left) and algorithm 1 + 2 (right). 89

3.11 The figure shows posterior realizations from multistage calibrated patch model for each borough in NYC. The light grey curves between week 1 and 10 shows prior realizations, whereas light green curves are posterior simulation output from first calibration using week 1-10 data. Dark grey curves between week 10 and 20 are posterior predictions based on previous 10 weeks of data, and dark green curves are posterior realizations using week 10-20 data. 90

3.12 Boxplot of two posteriors $p(\boldsymbol{\theta}|\boldsymbol{y}^{(1)})$ and $p(\boldsymbol{\theta}|\boldsymbol{y}^{(1)}, \boldsymbol{y}^{(2)})$. Red star shows the actual value of $\boldsymbol{\theta}$. 91

Chapter 1

Introduction

1.1 Premises

1.1.1 System, Model and Computer Simulation

> *Computer simulation* is the reproduction of the behavior of a system using a computer to simulate the outcomes of a mathematical model associated with said system.
>
> - Wikipedia

With an exponential increase in processing power of modern computers (as predicted by Gordon E. Moore, the founder of Intel Corporation [1]), and significant advancements in computing algorithms, researches in many scientific disciplines have started relying on simulations, computer experiments, computer models etc. As described in wikipedia (in layman's term), a computer simulation model or simply a computer model can be thought of as representation of one or more mathematical equations describing a *system*. A system may be as simple as an object of fixed mass moving in a straight line in vacuum with fixed acceleration, or as complicated as a space shuttle launched from the earth docking onto the International Space Station orbiting the earth. But both of these processes are governed by some principals depicted in one or more mathematical equations, which allows one to study or obtain the outcome of these processes given the initial conditions, without physically observing or performing the processes or the *experiments*. Some systems are

[1] He predicted that the processing power would be doubled in every two years, and it has been proven true till 2017.

simple enough to be *solved*, hence studied analytically, and some are intractable in which case we seek help of computing machines. *Complex* systems can be complex in nature due to countless interactions in the system or deviation from linearity or numerous other known-unknown factors. However, with the help of computing machines, we can still delve into such systems and hope to gain relevant inferences at least partially if not fully. Such practices are gaining traction ever since the modern computing chips and storage have become affordable and modern computer architectures have emerged, although this does pose challenges to researchers as traditional inference methods become less suitable for such avenue to solve a science problem.

By computer simulation we generally mean solving the governing equations given by the mathematical model by executing one or more algorithms which yields some output representing the behavior of the system at some instances. For example, a computer simulation could be solving a set of ordinary differential equations to determine the number of flu infected people in a region after 10 days of disease emergence, or it could be finding the trajectory of the Mar's rover launched from the earth with a given velocity and acceleration. Although these two systems are different, both of them work in a similar fashion, i.e., they require some input (e.g. initial condition etc.) to run. To solve the ODE system, it has to be supplied with the number of susceptible, exposed and infected people at the start of the disease propagation, and similarly, to find the trajectory of the Mar's rover, we need to input the mass of the object, density of the atmosphere etc. among many other things. These inputs can further be categorize into two major types, one with their values known and one unknown, which will be discussed later in detail. A more comprehensive discussion on the definition of computer simulation can be found in Winsberg [157]

1.1.2 Why Simulation?

Use of simulation in scientific research goes back to as early as 1777 Buffon 'needle experiment' (see [63]), where the goal was to estimate the value of π by throwing needles onto a plane with equally spaced parallel lines. Laplace's contribution is also acknowledged to be important and relevant.

Later on, in nineteenth century during world war II, rapid growth in simulation science was set by Stanislaw Ulam, John von Neumann, and others when they used the Monte Carlo method on electronic computers in order to solve problems in neutron diffusion in the design of the hydrogen bomb, which paved the way to modern age general simulation programs. Goldsman et al. [63] has put together a brief, but fairly detailed history on simulation.

The purpose of simulation can broadly be attributed to few categories, here we mention some of them. Apart from using computer simulation to execute algorithm in order to solve a set of equations, simulation allows one to mimic a real system, to represent the knowledge of the system or to communicate the knowledge to others. The simulation model can be used for further study providing an opportunity to observe the system under different scenarios as if they are occurring in a real setup, which may never be possible otherwise. Simulation based model becomes more appealing when the real system becomes increasingly complex as number of interaction grows rapidly, such that a numerical solution to the mathematical model remains only option to avail. These models also offer an avenue to verify and validate one's understanding of the system by coupling the simulation with real observations. Simulations are also used to make realistic predictions (both interpolatoin and extrapolation) and related *uncertainty* post *validation* and *calibration* steps.

The computational modeling approach thus offers scientist a different but rather intuitive direction to study real-world processes and to answer relevant questions regarding the system. Physics based and engineering models used to be the most popular ones to use such computational approaches, but such practices have since been adopted in a great number of science domains.

1.1.3 Challenges

Although computational models are intuitive and relatively easy to work with, there are number of issues and challenges associated with them which ought to be addressed by respective scientific community. For example, the domain scientist may care about correct representation of theory into the computational model, a computer scientist may want to keep control over the approximation

error, or a data analyst may be interested in validating the output from the model and assessing the reliability and uncertainty in depicting the physics or the reality through computer simulation. Sauro et al. [133] discuss in their paper the challenges of having multi-level computational model in system biology. The scientists often want to build an accurate computational model of the biological systems which operate at cellular level without sacrificing the simplicity of the model, which may seem counter-intuitive. There is also an philosophical side to it, Timm and Lorig [142] put it as a common assumption among scientific community that a valid simulation model can not exist, and only the real world can serve as its only valid model. A group of scientists [88] have gone around that notion and have worked out a way to account for the systematic differences between the simulated world and real world. However the question largely remains relevant in many applications till date.

There are other types of error as well, some of which can be considered as acknowledged error such as physical approximation error, computer rounding-off error, iterative convergence error, discretization error etc.. The meaning of physical approximation error can have an overlap with the previously discussed notion that no simulation model is perfect, and the difference can sometimes be accounted for partially. Rounding off error or convergence error or error due to discretization can broadly be attributed to machine's ability of storing numbers only up to a finite precision, which instigates a different type of error which can sometimes be addressed by improving the computing chips and/or the algorithms used in the simulation, or can be estimated.

While each of these said challenges merits to have their own share of detailed discussion, this thesis will be concerned with a subset of them and those will be addressed in appropriate chapters with sufficient illustrations. The following sections and sub-sections in this chapter will lay the premises of majority of the discussions by providing an overview of the path ahead.

1.2 Simulation and Statistical Inference

A model describes relation between inputs and outputs, where the output can be anything of interest. We stick to the traditional types of model where the output can be quantified; it can be as simple as a scalar or as complex as multidimensional array of arbitrary length, representing the behavior of the system through specific measures. A simulation model is no different than any other scientific model in a sense that user needs to provide an input to the simulation which would then yield some output. Some inputs may be controllable and whose values may be known prior to the execution, and some inputs may have their values unknown, requiring appropriate procedures to infer them. The later is also known as *inverse problem*. When prior knowledge of the input parameters includes uncertainty, then the simulation model provides an obvious way to propagate those uncertainties, yielding uncertainty bounds on the output of interest, also commonly known as *Quantity Of Interest* (a.k.a. QOI).

1.2.1 Deterministic vs. Stochastic Simulation

A broad classification among simulation models is based on output type, viz., deterministic and stochastic. In deterministic setup, repeated runs of the simulation at same input setting yields exactly same output, whereas, a stochastic simulation returns an ensemble when run repeatedly at one input, due to the inherent randomness present in the model. In other words the output of the deterministic simulation is fully determined by the given input. Stochastic models may arise due to the nature of the system that is being modeled, or due to the unobserved random error present in the model. For example, a model on population dynamics can acquire randomness because of discrete nature of different individuals, a weather model may inherit stochasticity due to lack of complete data from the environment, or a cosmological model can possess randomness due to not knowing the true governing model completely. Because of the differences, one can appreciate the need of different set of tools and techniques to handle necessary tasks such as verification and validation, calibration, uncertainty quantification, prediction etc., on deterministic and stochastic

simulation models. The state is further complicated by models of wide variety in nature, which may disable one to use similar tools in every situation.

1.2.2 Verification and Validation

One of the most important and perhaps the first task after building a simulation model is to perform *Verification* and *Validation*, or commonly referred as VV. Although there is a lack of universally accepted definition of these terms, in general the purpose of VV remains same across different groups. We adopt the definition of verification and validation used in the National Research Council report (see [33] for detailed VVUQ discussion).

Verification is referred to the process of determining how accurately a simulation model (a.k.a. computer program) solves the equations of the mathematical model or performs any required algorithms. The process may include checking for any possible bug in the computer code, or in the algorithm, or in the software itself. Code verification relies on the availability of the test problem whose actual solution using the intended algorithm is known beforehand, with which the new solution can be compared. However gathering a complete collection of test problems to check every relevant algorithm in question is not realistically achievable, given the vast and potentially many unknown sources of errors. For the same reason it is impossible to guarantee that a complex simulation code will work without any error outside the domain it has been tested. Solution verification essentially determines the correctness of the algorithm in use, specific to the QOI, and potentially tries to control the error. It is important to acknowledge that the (approximation) error in the solution of one QOI may not be comparable to another QOI, where QOIs are just some functions of full computable solution from the simulation model. Hence the process must be done for each QOI individually.

Validation, on the other hand, is the process of comparing the solution from the simulation model to the actual physical observation from the real world. This process establishes the degree of usefulness of the simulation model in representing the real physical system by comparing the closenessless of

the simulated QOI to the true QOI. Challenges in validation process include but not limited to not knowing the values to the input parameters, having measurement error or approximation error contaminating the solution or the physical measurement etc.. Any validation procedure should also consider the uncertainties present in the simulation input, requiring one to perform uncertainty quantification within the VV setup.

1.2.3 Calibration

Calibration is the process of finding the right setting(s) for the simulation model such that a simulation run at that setting yields output reasonably close to the observed QOI from the real physical system. Calibration process often overlaps with the validation step, where values of unknown input setting is inferred based on true physical observation, such that the simulation output agrees with the true observation. Similar to verification and validation process, calibration should also be executed in the context of pre-specified set of QOIs. A simulation model may be suitable for approximating one QOI, whereas erroneous for other QOIs.

Different set of calibration tools are available among scientific communities, often specific to certain type of simulation models. Calibration via gaussian process emulators in bayesian [11, 75, 76, 88, 120, 152] or frequentist [18] setting are widely accepted and popular techniques among statisticians. The problem of calibration is also known as *inverse problem* (opposite of forward problem where one seeks to predict output observable given the parameters) since it estimates the uncertain parameters from the observations. There are few challenges associated with an inverse problem: (a) the inverse mapping from observations to the parameters may not be one to one, (e.g. when the number of parameters are large or in case of stochastic system), (b) very high sensitivity in few or all parameters due to nonlinear nature of the system, and (c) unavailability of complete knowledge on the true physical system.

A typical calibration or inverse problem solution can be summarized by a *best estimate* of the unknown parameters and associated uncertainties. However, in many cases, the *best* estimate

may not be well-defined. Popular approach to obtaining a unique solution to the calibration problem is to treat it as an optimization problem, where the distance between the simulation and observation is minimized pertaining to some regularization, potentially to avoid unwanted complicacy in the parameters. However, a scientist may not be interested just in point estimate of the best parameter, but also in a full description of all parameter values that are consistent with the data. Bayesian techniques provide an easy avenue to obtaining complete uncertainty on the parameter space by reformulating the inverse problem as a statistical inference problem, taking into account the uncertainties in the observations, computer simulation model, and any prior information about the parameters. The solution to this typically consists of full posterior densities of the parameters, describing uncertainties in the model parameters given the true observations.

1.2.4 Emulation

Emulator is a mathematical/statistical model which behaves as an actual computer simulation. It often act as a replacement for computationally expensive simulations in optimization [95], sensitivity analysis [113], calibration and uncertainty quantification [88, 131] routines. An emulator can be thought of as a type of response surface model that aims to estimate the input-output mapping using limited number of runs from expensive computer simulation, often regarded as black-box function. A large number of emulation techniques have been proposed, based on different statistical and mathematical ideas. Gaussian process based model [66, 131] and Lagrange interpolants [99] interpolate between the model runs, where as other techniques which do not interpolate include polynomial regression [21], regression splines [85], projection pursuit [13], radial basis functions [55], support vector machines [30], and neural networks [72] and many more.

An emulator has to be accurate in order to be used in place of the actual computer simulation. The definition of accuracy may vary from one situation to another depending on nature of the simulation, quantity of interests etc.. For example, one may consider only the mean prediction from a gaussian process emulator for a deterministic computer model, but 90% confidence interval

for a stochastic model. Since not all computer simulations share similar characteristics, some emulations techniques work best only on subset of problems. Yet, the saving one makes by using emulators can be huge in terms of both computing resources and time.

However, since emulators estimate the actual simulation output, it is important to quantify the estimation error as well, and include that into subsequent analysis where the emulator replaces computer model. Statistical models sometime put parametric distributional assumption at each estimated output, which serve as full description of the emulator, and easily fit into a uncertainty quantification setup. We would see few illustrations of that later.

1.2.5 Prediction and Uncertainty Quantification

The final objective of a computer model is often to make predictions where the actual physical experiment has not been observed or impossible to do so. After the computer model is calibrated based on limited observations from physical system, either the true physical experiment is run, or a corresponding emulator is used at the setting where prediction is required. A prediction may include prediction error or uncertainties arising from model input, observation error, emulator uncertainty, numerical errors, systematic discrepancy and any other quantifiable errors [88]. In theory one can possibly generate a large number of monte carlo samples from the input space, and quantify the input uncertaity in model output by running and collecting the simulation model at those input settings. However, limited computational budget restricts the possibility of large number of model runs, and the number of sample points in higher dimensional spaces grows exponentially with the number of dimensions. Also, *low-probability high-consequence* events are not well captured in standard Monte Carlo schemes, because such events are rarely generated. An UQ procedure also deals with aggregation of all quantifiable uncertainties from different sources. The uncertainties in the prediction of QOI that are due to not knowing the true input, physical variability, numerical error, and model bias should be combined into a quantitative characterization of the overall uncertainty.

1.3 Disease Model

Mathematical modeling of disease dynamics dates back to 1766 when Daniel Bernoulli [15] first developed a model for analyzing mortality due to smallpox in England. However, most significant advancement in the epidemic modeling has happened in the last century, following some devastating outbreaks such as HIV, influenza, malaria, TB, ebola to name a few. Seminal work by Ross, Kermack and McKendrick [89] in 1927 laid the foundation of deterministic compartmental epidemic modeling, where they modeled the disease transmission at a macro scale by taking into account the number of contacts of an infected individual. Statistical models, unlike mechanistic ones, have also been developed, such as regression based model in Serfling [136], time series analysis based autoregressive models in Choi and Thacker [26], and neural networks in Bai and Jin [7]. Success of many such statistical models purely relies on availability of good quality data. In many cases, where surveillance, early detection of possible outbreaks and predicting the patterns in the disease spread are the primary goals to help eliminating or limiting the spread, collection of relevant data at a timely manner plays a big role in such situations. A review of statistical models for outbreak detection can be found in Unkel et al. [147]. However, such models are often difficult to interpret and relies on strong assumptions which limit their usability in many cases. On the other hand, mechanistic disease models, which we will be discussing in much more details, are realistic, easy to interpret, portable and usually require less expertise to run and analyze, but can also be extended to accommodate more complex diseases. While a complete review of mathematical disease models are out of scope in this report, the readers may find details in Siettos and Russo [138].

1.3.1 Compartmental Disease Model

Here, we discuss about compartmental disease models [89], where the population is divided into finite number of components based on *state* of one's health, namely, susceptible (S), infected (I), recovered (R). Other *states* can be introduced such as exposed (E), vaccinated (V) etc when necessary. The fundamental idea in compartmental model [89] is that the individual travels from one

compartment to another compartment in accordance to a mass-action, where an infected individual infects a susceptible with some probability (sometimes unknown), and recovers with some probability. These simple rules can be depicted in simple differential equations. However, to obtain closed form equations, one would assume the underlying process to be a markovian, satisfying the mean field assumption, which presumes that the population is perfectly mixed and homogeneous and the *transition* (from one state to another) is constant. Such assumptions lead us to the following set of differential equations defining the deterministic compartmental SIR model:

$$\begin{aligned}\frac{dS}{dt} &= \alpha SI,\\ \frac{dI}{dt} &= \alpha SI - \beta I,\\ \frac{dR}{dt} &= \beta I,\end{aligned}$$

where, S, I and R denote the mean number of susceptible, infected and recovered individuals in the population; α denotes the mean transition probability from S to E, and $\frac{1}{\beta}$ represents the mean number of days an infected individual can transmit disease before moving to state R, i.e. recovering. The above model serves as the base for more general models such as SIRS - where an individual who recovers from an infection becomes susceptible again, SEIR - where an individual may become infected but can not transmit the disease initially, which is called an exposed (E) state, and so on. Introducing a new compartment in the model does not require much engineering, except having two new transition probabilities representing the incoming and outgoing rate for that new compartment. An epidemic happens when the rate of change in I is positive and negative for S, i.e. $\frac{dI}{dt} > 0$ and $\frac{dS}{dt} < 0$. By simple calculation, one can see that the number of infections increases for $S > \frac{\alpha}{\beta}$. Hence, the threshold value $R_0 = \frac{\beta}{\alpha}S(t = 0)$ is known as basic reproduction number, and characterizes the disease outbreak. A value of R_0 greater than 1 indicates disease outbreak and at less than 1 the disease dies off. SIR models can also be extended to assimilate demographic distributions, mortality, and to account for spatial effects, migration etc., producing more realistic

models.

Some simulation examples of deterministic SIR and SEIR models are given below. These examples are produced with the help of software implementation of disease models through R [122] package *EpiModel* [83]. The parameters in a typical simulation consists of initial conditions such as number of people in each compartments and the transition probabilities between the compartments. Additional parameters arise as more complicated models are used. For example, two additional parameters would be considered to account for birth-death process in the population. All such simulations are based on discrete time events, i.e., state of an individual changes only at discrete time steps. Fig 1.1 shows output from an SIR model simulation for 500 time steps with infection probability of 0.2 and recovery probability of 0.05. Similarly, Fig 1.2 shows output from an SEIR model, with the additional compartment E. Here we make a note that repeated simulation runs at same initial setting using any of these two models yield same output every time.

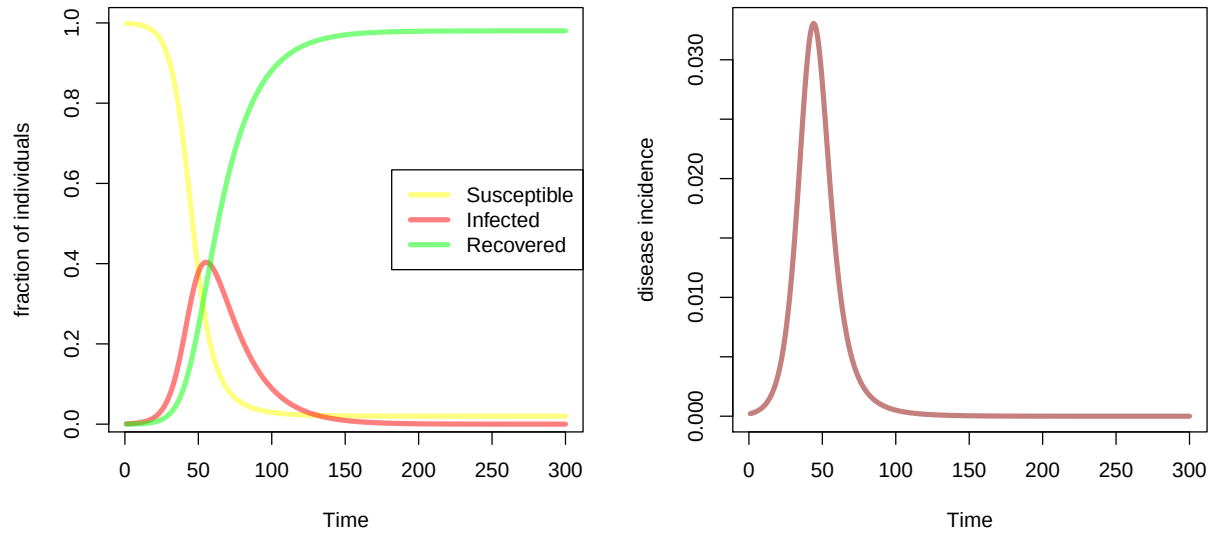


Figure 1.1: The left frame shows the simulated time series of the fraction of total population in each of the three compartments from a simulated deterministic SIR model. The right frame shows the fraction of newly infected individuals at discrete time points.

However, mean field approximation in the deterministic model and the total population being

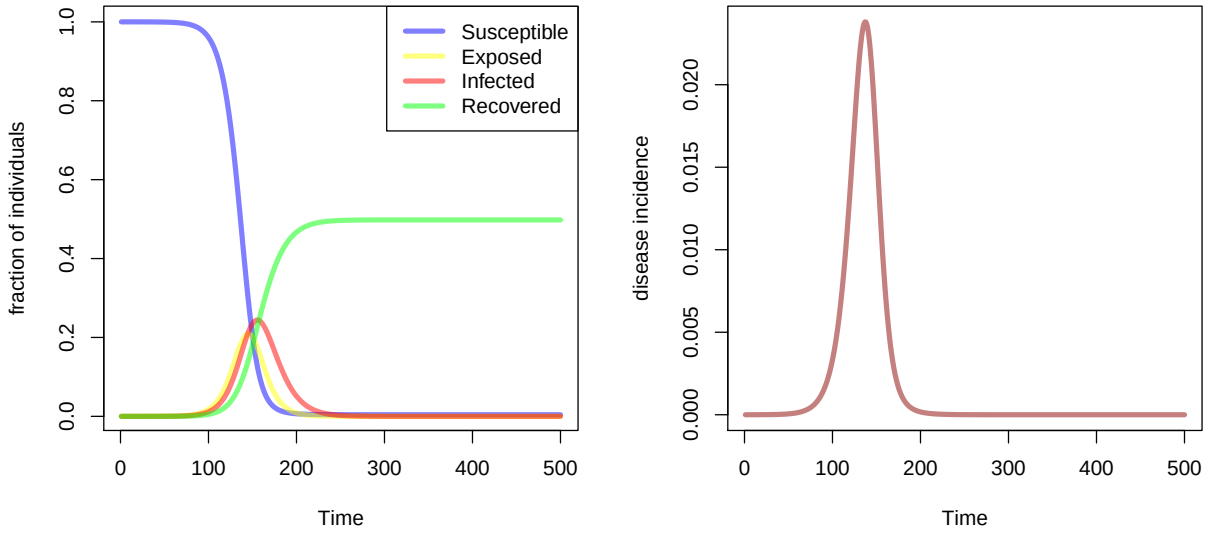


Figure 1.2: The left frame shows the simulated time series of the fraction of total population in each of the three compartments from a simulated deterministic SEIR model. The right frame shows the fraction of newly infected individuals at discrete time points.

infinite are quite strong assumptions, limiting the realism of a mechanistic model. Stochastic models based on discrete and continuous time markov chain relax simple but unrealistic mean field assumption, allowing one to account for individual behaviours. For example, in a discrete time markov chain the state of every individual changes at discrete time steps according to a probability rule involving their own state at present time and the state of *neighbors*. A typical output from such model is stochastic, i.e., at every time step the state of an individual is a realization from a probability distribution. Similar to its deterministic counterpart, often the transition probabilities are assumed to be constant in time, resulting in a homogeneous markov process. In limit with respect to infinite number of individuals and with the assumption that each individual is connected to every other, the stochastic compartmental model reduces to deterministic model. However, the stochastic model is more appropriate when fully connected assumption is not valid in the presence of heterogeneous links between individuals, finite population and additional characteristics (such as demographics, economic etc.) which may affect disease propagation. Some applications of stochastic SIR, SEIR models along with a comparison with deterministic models are found in

[1, 19, 96].

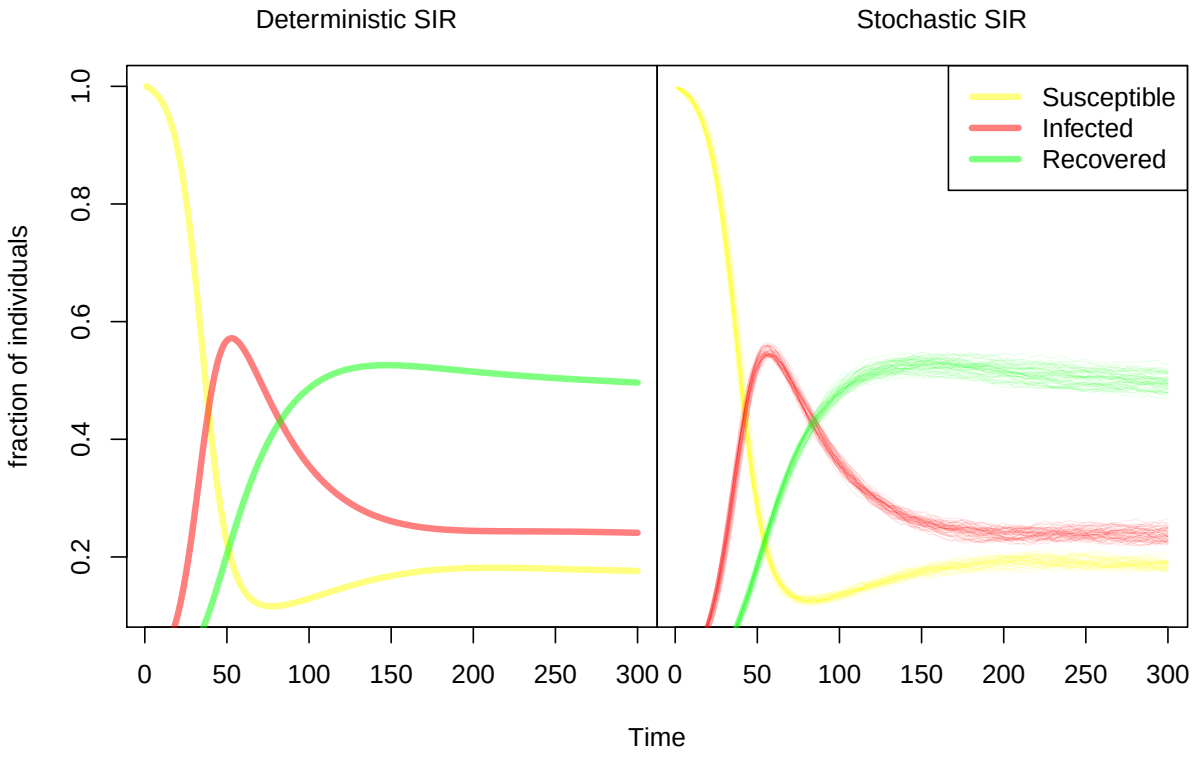


Figure 1.3: The left frame shows the simulated time series of the fraction of total population in each of the three compartments from a simulated deterministic SIR model, and the right frame shows simulation output from its stochastic counterpart.

A comparison between simulation output from a deterministic SIR model and its stochastic version is given in Fig 1.3. The mean response from stochastic simulation is very similar to the deterministic output, however, the extra randomness among the replicates in stochastic model gives an opportunity to incorporate uncertainties from random effects into further analysis.

1.3.2 Network Based Model

Both of the previous models assume homogeneous mixing in the population, and provides very few opportunities to relax the assumption by introducing more factors in the differential equations. Yet, the most critical factor in epidemic spread happens to be the effect of contact network heterogeneity,

and the dynamics. Even the contacts between individuals evolve over time, possibly due to various reasons (complex biological or socioeconomic reasons etc.), which directly or indirectly affects the disease dynamics. For example, a certain pathogen may have high interaction with particular type of hosts, making a group of population more vulnerable than others. Or in other case, a well informed individual may opt for preventive actions to help containing the infectious virus within a boundary. Understanding and considering this complex interaction between the evolving disease characteristics as well as the changing network topology in a mechanistic model may certainly help in making more accurate predictions. Most recent disease modeling efforts have been attributed to these type of modeling approaches. A flu outbreak at Harvard University in 2009 was studied in [28] using en empirically generated contact network from the students, whereas similar high-resolution human contact network based disease transmission model is developed in [132] to study outbreak at an American high school. In [128], spatio-temporal network extracted from Brazilian internet community along with SI and SIR mechanism is used to model sexually transmitted diseases.

Complex network based disease models not only captures properties of different elements of the whole system, it also facilitates flexibility in implementation and interpretation. In modern mathematical epidemiology, agent-based modeling (also referred as individual based model, or multi agent system) is the state-of-the-art technique in the arsenal for simulating any complex system. Details such as transportation infrastructures, population mobility, demographics, host-pathogen interaction, and the evolution of contact network can be embedded in an agent based system, where the simulation happens at very micro-level taking into account every possible micro or macro interactions affecting disease dynamics. These increasingly realistic simulations not only helps in predicting the disease characteristics in early periods of the epidemic, but are also incredibly useful in testing many what-if scenarios without compromising the safety of people at risk. This has gained significant popularity among public health scientists and policy makers due to its appropriateness and cost effective alternative ways for experimenting new policies or drugs or strategies. EpiSims [49] is one of such highly detailed agent based disease simulator, which couples data from micro mobility simulation, and traditional compartmental style disease model. Demographic information

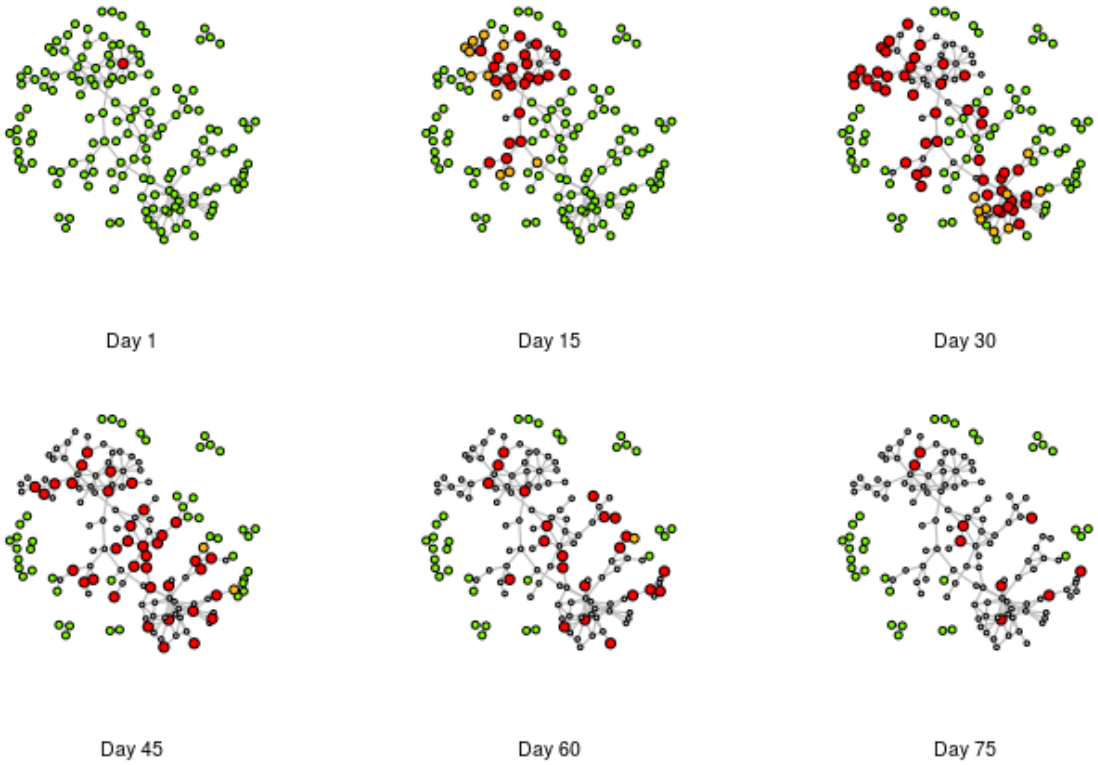


Figure 1.4: The figure shows the simulated disease evolution on a school network [65] using an agent based SEIR model. The node colors green, yellow, red and grey denote susceptible, exposed, infected and recovered state respectively, whereas infection flows through the darker edges.

including age, gender, household description, geographic information having locations for different activity types (e.g. school, work place, shopping place), and typical daily schedules of individuals are combined to create a synthetic network of anonymized set of individuals representing a real world, which acts as the based of agent based simulation. In this report, we would work with two specific stochastic disease simulation models as examples to demonstrate statistical techniques for calibration and uncertainty quantification. In the next section we describe those two models in details.

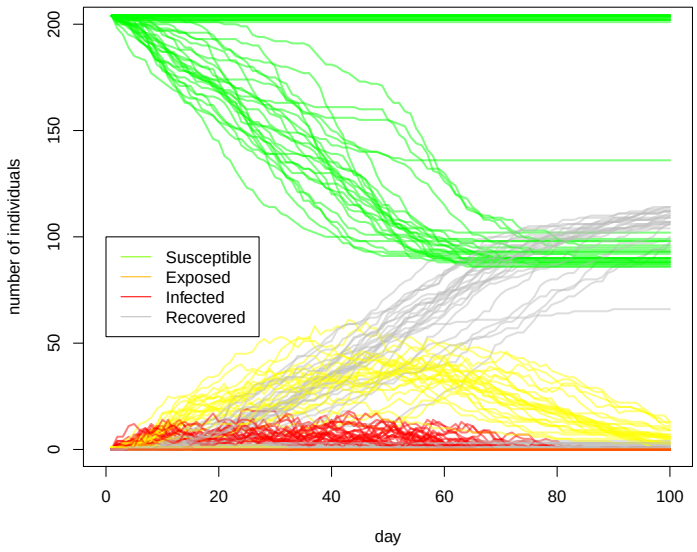
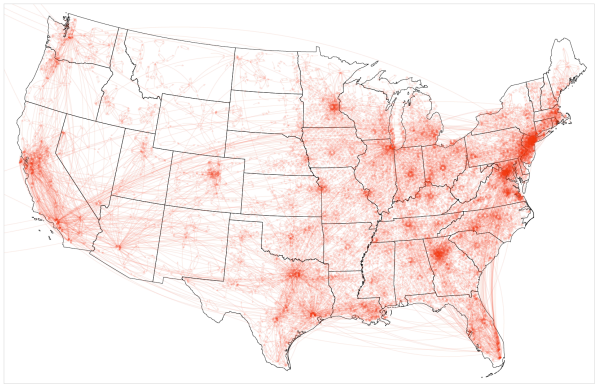


Figure 1.5: The figure shows output from a stochastic agent based SEIR simulation, ran 100 times at same configuration.

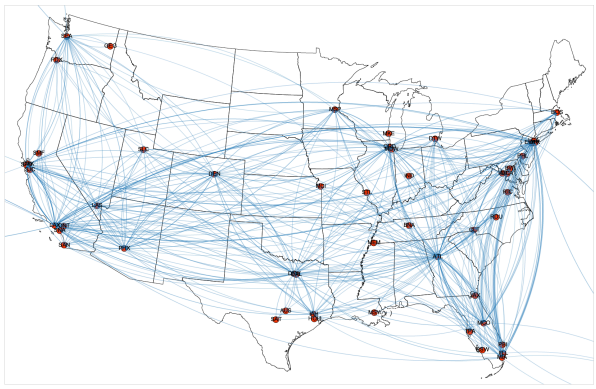
Travel based multi-population SEIR model

In this section, we describe a spatial compartmental model (also referred to as a patch model) [148] which is primarily developed to model the spread of Influenza in contiguous US. The idea is to define simple compartmental SEIR models for each spatial region (counties, or states of HHS regions) having homogeneous mixing within that region (aka patch), and a spatial travel network among the regions adding heterogeneity across the patches. Such model is simplistic in nature, yet preferred in situations where very limited or no data is available on social contact network and human mobility to construct a highly detailed agent based type models. The patch model consists of two main components - travel module and disease dynamics module.

Travel module defines the temporal flow of people from one patch to another patch, denoted by the travel matrix Θ. Each entry Θ_{ij}^{t} of the travel matrix represents the fraction of individuals in patch i spending their time in patch j on day t. Available commuter data from American Community



(a) Commuter flow among counties in contiguous United States (excluding Alaska and Hawaii)



(b) Domestic airline flow among select major airports in United States

Figure 1.6: Datasets used to capture short-range and long-range mobility in United States (image courtesy: Srini Venkatramanan)

Survey (ACS) [2] and long distance air travel from the Bureau of Transportation Statistics (BTS) [3] as seen in Fig 1.6 creates the entries in travel matrix Θ. By denoting the effective number of individuals from patch i who are in patch j on day t is given by O_{ij}^t, we define $\Theta_{ij}^t = \frac{O_{ij}^m}{P_i}$, where P_i is the population in patch i.

Within each patch, a simple SEIR model is defined as follows, where each individual can be in any one of the following states: Susceptible (S), Exposed (E), Infected (I), Recovered or Removed (R), Vaccinated (V). $S_i(t), E_i(t), I_i(t), R_i(t)$ denote the number of individuals in the above states, respectively, at time t. Initially, the entire population is assumed to be susceptible, except for a few initial infections possibly due to external contact.

[2] https://www.census.gov/programs-surveys/acs/
[3] https://www.rita.dot.gov/bts/data_and_statistics/

$$\begin{aligned}
\frac{dS_i}{dt} &= -V_i(t) - \beta S_i(t)\frac{I_i(t)}{N_i} \\
\frac{dE_i}{dt} &= \beta S_i(t)\frac{I_i(t)}{N_i} - \alpha E_i(t) \\
\frac{dI_i}{dt} &= \alpha E_i(t) - \gamma I_i(t) \\
\frac{dR_i}{dt} &= \gamma I_i(t),
\end{aligned}$$

where β denotes the transmissibility (which quantifies the rate at which an infected individual spreads the infection to each susceptible contact), and α and γ denote the incubation and recovery rate, respectively. Individuals who become recovered do not get infected again.

The above set of equations are extended for multiple patches such that susceptible individuals can be infected by infectious individuals from other patches. This depends on the fraction of individuals moving from county i to county j on any given day. The temporal travel matrix as obtained earlier is used to calculate the *force of infection* among patches, without actually moving infected individuals around (i.e., virtual dispersal). Let $I_j^{\mathtt{eff}}$ and $N_j^{\mathtt{eff}}$ be the effective number of infected individuals and effective population of patch j after travel on a given day.

$$I_j^{\mathtt{eff}} = \sum_i \Theta_{ij} I_i;\ N_j^{\mathtt{eff}} = \sum_i \Theta_{ij} N_i$$

One can then write the conditional force of infection for an individual in patch j as $\beta_j^{\mathtt{eff}} := \beta \frac{I_j^{\mathtt{eff}}}{N_j^{\mathtt{eff}}}$. $\beta_j^{\mathtt{eff}}$ is *conditional* because it is conditioned on the observation that the individual is in patch j. We can then calculate the unconditioned force of infection on a susceptible individual from patch i, as a component-wise product of (a) the probability of the individual being in a patch j (Θ_{ij}^t) and (b) the conditional force of infection in patch j ($\beta_j^{\mathtt{eff}}$).

For county i, the evolution of $\mathcal{Z}_i = [S_i, E_i, I_i, R_i, V_i]$ from t to $t+1$ can be defined as follows:

$$\mathcal{Z}_i(t+1) = \mathcal{Z}_i(t) + \Delta\mathcal{Z}_i(t)$$

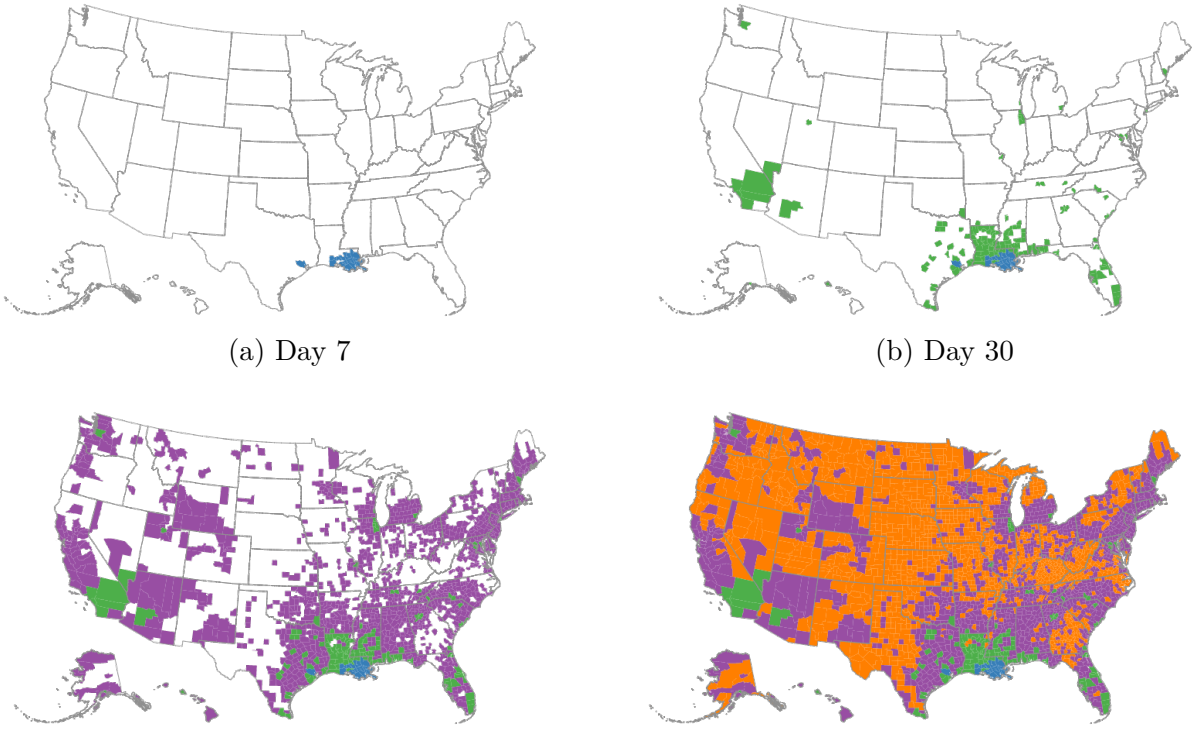
where,

$$\begin{aligned}
\Delta S_i &= -\Delta V_i - \sum_{j=1}^{K} \Theta_{ij} \beta_j^{\texttt{eff}} S_i \\
\Delta E_i &= \sum_{j=1}^{K} \Theta_{ij} \beta_j^{\texttt{eff}} S_i - \alpha E_i \\
\Delta I_i &= \alpha E_i - \gamma I_i \\
\Delta R_i &= \gamma I_i
\end{aligned}$$

and ΔV_i is the number of newly vaccinated nodes (at time t). Note that, this patch model can be operated as a deterministic simulation using mean field approximation, or can be regarded as stochastic by using random draws from probability distributions in the simulation. An illustration of the simulation output is shown in Fig 1.7.

Agent based model using synthetic population

The other simulation model that we will be using in later chapter is an agent based model as proposed in [49]. This particular version of the model is used to describe the 2014 Ebola epidemic [139] in Liberia. Much like patch model, agent based network model for disease dynamics relies on few key components, a realistic synthetic population, social contact network and a disease model. First, a synthetic population for Liberia is created using demographic and land use data from various sources such as census, landscan and surveys. This includes marginal demographic information of individuals as well as population densities, geo-location of workplaces, households and schools of different sizes and types. This coarse data is synthesized to fill in spatial cells resulting in an anonymized synthetic population, a detailed description of this process is given in [70]. Daytime location of each individual is also estimated according to the person's daily assigned activity and the activity location. A social contact network is then obtained by observing the proximity of any individual to any other given their geo-locations. This forms the basis for an agent based simulation to simulate diseaes propagation through contacts. Fig 1.8 gives an overview



(a) Day 7

(b) Day 30

(c) Day 90

(d) Day 180

Figure 1.7: Spatio-temporal spread of influenza, for a sample scenario, seeded in Louisiana. Counties where the epidemic emerges by Day 7,30,90 or 180 are are respectively shaded blue, green, purple and orange.(image courtesy: Srini Venkatramanan)

of the complete process.

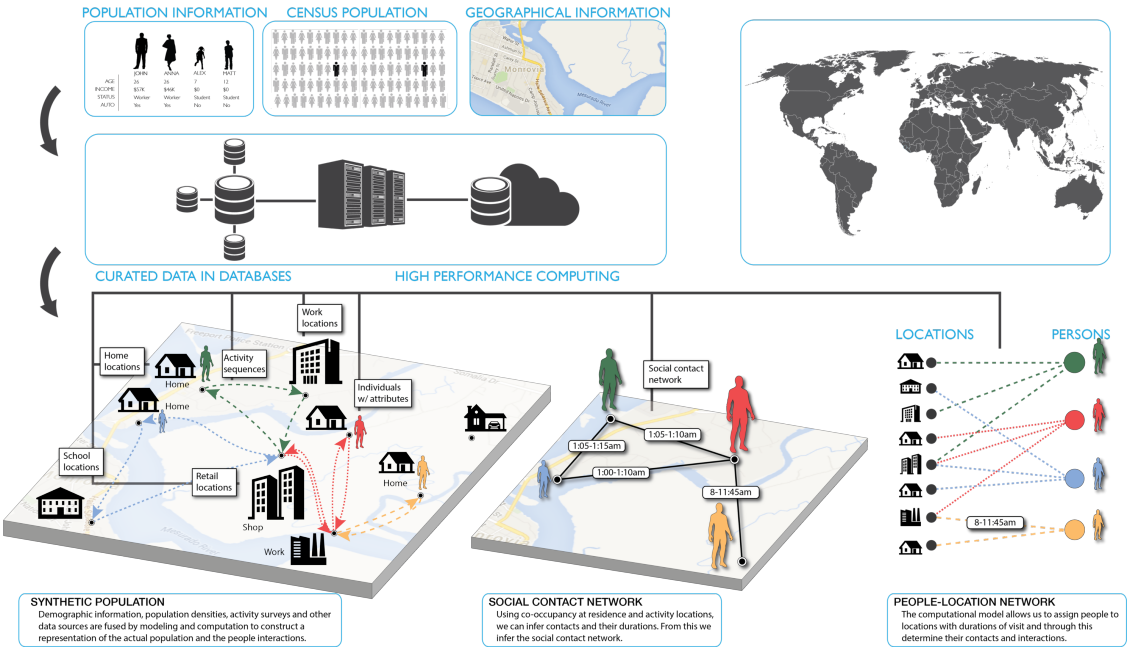


Figure 1.8: A pictorial description of how synthetic population is generated. The process consists of generating (1) an anonymous population of individuals with detailed demographic and spatial information, (2) a mapping between every individual and their activity locations, and (3) a derived edge-weighted social contact network to be used for disease propagation. (image courtesy: Henning Mortveit)

The ebola agent based model requires some specific parameters apart from usual disease parameters, namely, number of initially infected people, infectious period distribution, incubation period distribution, transmissibility. Parameters mimicking the effect of hospitalization and social isolation are taken into consideration for modeling ebola. Some of these parameters are considered to be calibration parameters, i.e., their values has to be estimated given data. It is important to note that, unlike patch model there is no deterministic counterpart of this agent based model. Hence, a simulation run would not match with any other run, even at same input settings (see Fig 2.2).

Chapter 2

Emulation and Calibration

2.1 Related Work

Most of the calibration procedures requires of a computer model to be run numerous times at different input settings, in order to find the 'best' or the plausible setting(s) where the model outputs are close to the observations [2, 73, 149]. Simulations, which are expensive to run, add challenges when one can afford only a finite number of runs under limited budget. Sacks et al. [131] suggested use of less expensive surrogate model or emulator as replacement for the simulator in the optimization/calibration loop. Since then, emulation has become an integral part of any computer model calibration framework, where handling expensive simulator is of priority. Gaussian process model [18, 34, 35, 75, 76, 124, 131, 154] in it's various forms has been the most popular emulation techniques among the statisticians and mathematicians due to it's nice properties, ease of applicability, and ability to model a wide variety of dependencies in a non-parametric way. To build a GP emulator, one needs to run the actual simulation at a few carefully chosen input settings [131] in the given input space. Based on those few simulations, one then carry out the inference on the unknown parameters governing the GP model using any preferred inferential procedures. This trained emulator then works as a surrogate to the actual simulation and provides an estimate of the simulator output along with prediction uncertainties at input settings where the actual simulation is not available.

A number of different calibration techniques have been in use recently, some of which use surrogate models. For example, Kennedy and O'Hagan [88] (KOH) laid the foundation of Bayesian calibration

using GP and Higdon et al. [75, 76] proposed a fully Bayesian framework by combining observations with the simulation in the calibration, whereas Bayes linear history matching was developed in [64], Vernon et al. [150], bypassing the full Bayesian approach due to impracticality. Binois et al. [18] developed a computationally efficient heteroskedastic GP to model input dependent noise. While GP emulators are mostly preferred, in some situations, the computational complexity of training a GP model becomes large enough making it unsuitable for practical purposes, when the input dimension becomes too large. Andrianakis et al. [2] uses a simple linear regression emulator with history matching to avoid the computation burden while proposing an efficient sampling algorithm to sample from the input space. Gaussian process is also used as a surrogate to the likelihood function as opposed to the simulator to carry out calibration in Oakley and Youngman [114].

It is important to acknowledge and account for simulator discrepancy in the context of calibration. An estimate of the tuning parameters may result in a biased one, specially when the parameters have physical meaning and can be observed directly [22]. Furthermore, the interplay between the tuning parameter and the discrepancy may rise the indentifiability issue which are not uncommon in many situations, and have been acknowledged by many practitioners [88]. Prior specification of the input parameters as well as the choice of discrepancy model play a significant role in getting a true and meaningful calibration output. Wong et al. [158] suggested a frequentist approach addressing the indentifiability issue between the computer model parameters and discrepancy function by offering a new parametrization of the calibration problem, with good convergence properties. Similarly, Tuo and Jeff Wu [145], Tuo et al. [146] came up with L_2 calibration method arguing the L_2 inconsistency of KOH method. Handling of stochastic simulators, where repeated runs at an input yield a distribution of output, requires slightly different set of tools. An extension to the existing methodologies would be to emulate the mean and covariance of the simulator. Henderson et al. [73] opted for emulating the probabilities for binary data, whereas Plumlee and Tuo [121] proposed quantile kriging approach to model the replicate variability. Other methods including optimization via importance sampling Asmussen and Rubinstein [6], Rubinstein [130], and derivative-free search based algorithms Venkatramanan et al. [149] have also been used as alternative to KOH in different

applications.

2.1.1 Bayesian Model Calibration

In this section, we lay foundation of the Bayesian Computer Model (a.k.a. BMC) calibration set up as described by Kennedy and O'Hagan [88] in their seminal paper. Based on the gaussian process [124] regression for computer model [131], this procedure models the physical process as a noisy version of the computer model plus model discrepancy, and uses bayesian statistical tools to infer the unknowns. For rest of the document, Kennedy O'Hagan framework will be referred as KOH in short. In following paragraphs, notations along with the specific terminologies are introduced, and a general overview of the KOH for the univariate output setting is given. Some details are skipped to avoid repetition in following sections.

The central idea in KOH framework is to replace the expensive computer model by a cheap statistical model, commonly referred as *emulator*, which is then used to find the right setting at which the computer model agrees with the limited physical observation available, along with estimates of model bias or model discrepancy. An obvious choice for cheap and accurate emulator is gaussian process regression [124, 131], which provides an interpolator where the actual computer simulation has been run, and predicts the simulation output at other location, with an estimate of prediction error. In any calibration framework, the method consists of running the computer model or the emulator multiple times at different input settings and comparing the output with the physical observation, until the plausible input settings given the observations are found.

Let us denote a physical process by $\zeta(\boldsymbol{x})$, where $\boldsymbol{x}$ denotes the system input (which can be vector-valued), and the computer model or the simulator by $\eta(\boldsymbol{x}, \boldsymbol{\theta})$, with $\boldsymbol{\theta}$ being the input parameter needed to run the simulation, whose optimal value is often unknown, along with system input $\boldsymbol{x}$. In general bold face letters should denote a vector-valued object unless otherwise mentioned. Noisy observation of the physical system ζ at $\boldsymbol{x}$ is denoted by $y(\boldsymbol{x})$. The statistical model connecting the

physical process to the its observed counterpart is given by

$$y(\boldsymbol{x}_i) = \zeta(\boldsymbol{x}_i) + \epsilon(\boldsymbol{x}_i), \quad i = 1, \cdots, n,$$

where, $\epsilon(\boldsymbol{x}_i)$ denote the observation error. In the disease model case, $\zeta(\boldsymbol{x}_i)$ may represent the number of infected people as a function of $\boldsymbol{x}_i$, which may encode time and location. Often times the distribution of the observational error $\epsilon(\boldsymbol{x}_i)$ is treated as known, whenever prior knowledge is available and suitable assumptions are met. We take $\boldsymbol{y}$ to be the vector of n physical observations $y_1, \cdots, y_n$, at $\boldsymbol{x}_1, \cdots, \boldsymbol{x}_n$.

In the next step the unknown physical process $\zeta(\boldsymbol{x})$ is replaced by the computer simulator $\eta(\boldsymbol{x}, \boldsymbol{\theta})$ at the best calibration input setting $\boldsymbol{\theta}$ given by,

$$y(\boldsymbol{x}_i) = \eta(\boldsymbol{x}_i, \hat{\boldsymbol{\theta}}) + \delta(\boldsymbol{x}_i) + \epsilon(\boldsymbol{x}_i) \;\; i = 1, \cdots, n,$$

where the stochastic term $\delta(\boldsymbol{x}_i)$ accounts for the model bias or discrepancy between the simulator $\eta(\boldsymbol{x}_i, \boldsymbol{\theta})$ and reality $\zeta(\boldsymbol{x}_i)$, and $\hat{\boldsymbol{\theta}}$ denotes the best plausible, but unknown setting for the calibration input $\boldsymbol{\theta}$.

In many situations, where budget is limited and the computational demand for computer model is huge, only a handful number of simulation can be run. Those limited number of simulations are carried out at pre-specified input locations in order to build a training data set based on which a cheap but accurate emulator is trained, in order to predict the simulation output where the actual simulation has not been run. A gaussian process (GP) model [115, 124, 131] is then specified for the unknown function $\eta(\cdot, \cdot)$ which maps $R^{p_x + p_\theta}$ to R, where p_x and p_θ represent the dimension of $\boldsymbol{x}$ and $\boldsymbol{\theta}$ respectively. A GP model is fully characterized by its mean function $\mu(\boldsymbol{x}, \boldsymbol{\theta})$ and covariance function $\Sigma((\boldsymbol{x}, \boldsymbol{\theta}), (\boldsymbol{x}', \boldsymbol{\theta}'))$. Standard practices include scaling all inputs to unit hypercube, and treating the GP mean as a constant function of the inputs. However, non-constant GP mean models such as linear regression or splines are also proven to be relevant in some cases.

We specify a product covariance function here for this illustration and later use, although many other choices viz., exponential, matern covariance functions are available which are widely popular among modelers.

$$\begin{aligned}\Sigma((\boldsymbol{x},\boldsymbol{\theta}),(\boldsymbol{x}',\boldsymbol{\theta}')) &= \frac{1}{\lambda_\eta}\prod_{k=1}^{p_x}\rho_{\eta k}^{4(x_k-x_k')^2}\prod_{k=1}^{p_\theta}\rho_{\eta,p_x+k}^{4(\theta_k-\theta_k')^2} \\ &= \frac{1}{\lambda_\eta}R((\boldsymbol{x},\boldsymbol{\theta}),(\boldsymbol{x}',\boldsymbol{\theta}');\boldsymbol{\rho}_\eta),\end{aligned} \tag{2.1}$$

where λ_η is the marginal precision of simulator η and the dependence strength is embeded in the $(p_x + p_\theta)$ vector $\boldsymbol{\rho}_\eta$, in each dimensional direction of $\boldsymbol{x}$ and $\boldsymbol{\theta}$. The choice of this particular covariance structure produces a smooth, infinitely differentiable representation of the simulator η. The parameter $\rho_{\eta k}$ denotes the correlation between outputs at inputs that vary in only the k-th dimension by half of their domain. The GP description is completed by the prior specification for the parameters. We will discuss the prior specifications in section 2.2.2.

The discrepancy or the model bias term $\delta(\boldsymbol{x})$ accounts for the impurity in the computer model. Such impurities can often be attributed to lack of complete knowledge of the reality, the approximation error in computer algorithm etc. and few more, leading to systematic differences between the actual physical process $\zeta(\boldsymbol{x})$ and the calibrated model $\eta(\boldsymbol{x},\boldsymbol{\theta})$. A GP model is also used for the discrepancy term $\delta(\boldsymbol{x})$ with zero mean and a product covariance function of the following form,

$$\begin{aligned}\Sigma(\boldsymbol{x},\boldsymbol{x}') &= \frac{1}{\lambda_\delta}\prod_{k=1}^{p_\delta}\rho_{\delta k}^{4(x_k-x_k')^2} \\ &= \frac{1}{\lambda_\delta}R(\boldsymbol{x},\boldsymbol{x}'),\end{aligned} \tag{2.2}$$

Having the model well-defined, we then combine the field observation with the simulation outcomes to construct the likelihood. We define $\boldsymbol{\eta} = (\eta(\boldsymbol{x}_1^*,\boldsymbol{\theta}_1^*),\cdots,\eta(\boldsymbol{x}_n^*,\boldsymbol{\theta}_n^*))^T$ to be the vector of simulator response from the experimental design and the joint $(n+m)d$ data vector $D = (\boldsymbol{y}^T,\boldsymbol{\eta}^T)^T$ with corresponding input values $(\boldsymbol{x}_1,\hat{\boldsymbol{\theta}}),\cdots,(\boldsymbol{x}_n,\hat{\boldsymbol{\theta}}),(\boldsymbol{x}_1^*,\boldsymbol{\theta}_1^*),\cdots,(\boldsymbol{x}_m^*,\boldsymbol{\theta}_m^*)$. The likelihood for the

observed data D is written as,

$$L(D|\theta,\mu,\lambda_\eta,\boldsymbol{\rho}_\eta,\lambda_\delta,\boldsymbol{\rho}_\delta,\Sigma_y) \propto |\Sigma_D|^{-\frac{1}{2}} \exp\left\{ -\frac{1}{2}(D-\mu\mathbf{1}_{m+n})^T\Sigma_D^{-1}(D-\mu\mathbf{1}_{m+n})\right\}, \tag{2.3}$$

where,

$$\Sigma_D = \Sigma_\eta + \begin{pmatrix} \Sigma_y+\Sigma_\delta & 0 \\ 0 & 0 \end{pmatrix},$$

Σ_y is the $n\times n$ covariance matrix corrensponding to the observations, Σ_η is the covariance matrix obtained by applying (2.1) to each of the $n+m$ input points corresponding to D, and Σ_δ is obtained by (2.2) on $\boldsymbol{x}_1,\cdots,\boldsymbol{x}_n$ which corresponds to the field observation $\boldsymbol{y}$. The likelihood (2.3) along with appropriate priors π for the parameters constitutes the resulting posterior density of the following form,

$$\pi(\boldsymbol{\theta},\mu,\lambda_\eta,\boldsymbol{\rho}_\eta,\lambda_\delta,\boldsymbol{\rho}_\delta|D) \propto L(D|\theta,\mu,\lambda_\eta,\boldsymbol{\rho}_\eta,\lambda_\delta,\boldsymbol{\rho}_\delta,\Sigma_y)\times\pi(\mu)\times\pi(\lambda_\eta)\times\pi(\boldsymbol{\rho}_\eta)\times\pi(\lambda_\delta)\times\pi(\boldsymbol{\rho}_\delta)$$

This posterior is then explored using Markov chain Monte Carlo (MCMC) [16, 107], and inference can be drawn based on the posterior samples. The multivariate extension is described in section 2.2.2 in conjunction with the quantile based emulation for a stochastic computer model.

2.1.2 Stochastic Computer Model Emulation

A stochastic computer model produces realizations from the conditional distribution of it's output when run multiple time at a fixed input, condition being on that input setting, as opposed to same output in case of a deterministic computer model. Stochastic computer models are being used in many scientific domains, to represents dynamical processes more appropriately by introducing randomness into the mathematical model. The stochasticity may arise from the unpredictability of an individual's action in an Agent Based Model (ABM) [51, 117, 140], or it may account for the uncertainty due to the fact that the mathematical model is a close approximation only upto a

factor. For example, a simulator consisting of a set of partial differential equations does not have a closed form solution, enforcing one to discretize the problem and approximate numerically [5, 90]. Statistical methods for computer model emulation and calibration have largely been focused on deterministic computer models, however, many of them can readily be extended to a stochastic set up [92]. However, due to broad range of unalike computer models, no one particular technique has been proven to be applicable to every occasion.

In the context of computer model emulation, a rather straight forward extension of existing GP emulation is used where one models the mean response as function of input parameters [2, 93, 94]. Such emulators can be used in a calibration framework based on genetic algorithms or stochastic approximation Yuan et al. [160]. But in order to do UQ, one must account for the variability in the model output, which can be achieved by modeling both the mean and variance as function of input parameters, assuming the output distribution to be gaussian. A GP model can be used for this purpose [73, 103]. Other strategies to deal with non-gaussian output include use of mixture of normals [126], or directly modeling the Gaussian uncertainty in the error term of the GP [18] allowing for heteroskedastic variation about the mean of the GP.

Assuming an emulator is necessary for a stochastic computer model, one can have various choices in terms of what an emulator should emulate. Unlike emulating mean and variance for a normal distribution, Plumlee and Tuo [121] vouched for quantile emulation to empirically build the output distribution without any parametric assumption on the distribution. In other cases, Henderson et al. [73] and Oakley and Youngman [114] constructed emulators for the likelihood, rather than the output itself in order to avoid additional complexity due to multivariate output in their application, similar to modeling of probabilities by Diggle et al. [39].

While the above methods target different aspect of a stochastic computer model, it would rather be an unrealistic expectation to have a one size fits all emulation technique. Depending on the purpose of the emulator and the specific type of computer model in hand, one may redefine the objective appropriately, and can only hope to serve a broad class of similar problems. In the next few sections

we lay down our proposed emuation framework based on gaussian process model, describing it's applicability and usefulness through rigorous illustration.

2.2 Calibration via Quantile Based Emulation

Quantile emulation targets to construct the simulator distribution at any arbitrary input setting by emulating the quantiles as function of the input. We define $\eta_\alpha(\boldsymbol{x}, \boldsymbol{\theta})$ to be the α quantile of the distribution of the output from our stochastic simulator η at input setting $(\boldsymbol{x}, \boldsymbol{\theta})$, such that, $\eta_\alpha(\boldsymbol{x}, \boldsymbol{\theta}) = \inf\{u : P(\eta(\boldsymbol{x}, \boldsymbol{\theta}) \leq u) \geq \alpha\}$. By treating $\eta_\alpha(\boldsymbol{x}, \boldsymbol{\theta})$ as a function of $(\boldsymbol{x}, \boldsymbol{\theta})$, we can apply known modeling techniques such as Gaussian process to estimate $\eta_\alpha(\boldsymbol{x}^*, \boldsymbol{\theta}^*)$ at any arbitrary $(\boldsymbol{x}^*, \boldsymbol{\theta}^*)$, assuming such quantile functions are continuous. The real advantage of such framework lies in the fact that we can still reuse known statistical modeling techniques, and can construct an emulative distribution without any prior assumption on the distribution of $\eta(\boldsymbol{x}, \boldsymbol{\theta})$ as opposed to mean and covariance emulation [160] of Gaussian distribution. However, this does require the access to the quantiles. Given we observe α quantile $\eta_\alpha(\boldsymbol{x}, \boldsymbol{\theta})$ for each $(\boldsymbol{x}, \boldsymbol{\theta})$ in the input space, a simple GP emulator like one in sec 2.1.1 gives an estimate of the α quantile at any other $(\boldsymbol{x}', \boldsymbol{\theta}')$. Seldom is the case where such quantiles are available, but one can run the stochastic simulator multiple times and estimate the empirical quantile $\hat{\eta}_\alpha(\boldsymbol{x}, \boldsymbol{\theta})$. Exploiting the continuity property of the quantile functions, one can treat quantile as an input to the stochastic emulator which predicts any quantile at any input setting, hence gives access to an estimate to the emulative distribution.

2.2.1 Univariate Response

In this section, we see a simple illustration of quantile emulation on univariate response model taken from the Ebola application described in chapter 1.3. We continue to denote the computer model or the simulator, in this case the ABM disease model in chapter 1.3, by η, and the system input by $\boldsymbol{x}$ and calibration input by $\boldsymbol{\theta}$. We will drop the system input $\boldsymbol{x}$ in the equations where applicable,

noting that the general recipe remains the same even with a system input. The ABM needs an input $\boldsymbol{\theta}$ of dimension $p = 5$, $\boldsymbol{\theta} = (\theta_1, \cdots, \theta_5)$ (Table 2.1) to run. We start with an experimental design of $m = 100$ input parameter settings $\boldsymbol{\theta}_1^*, \ldots, \boldsymbol{\theta}_m^*$, random realizations from a space-filling, symmetric Latin hypercube design [159]. The 2-d projection of this $m \times p$ design are shown in Figure 2.1. However, for this univaraite illustration we fix 4 input dimensions $\theta_2, \theta_3, \theta_4, \theta_5$ at their nominal settings, while letting θ_1 vary in its range, i.e. essentially treating $\boldsymbol{\theta} = \theta_1$.

The ABM is run $r = 100$ times at each of the $m = 100$ input settings, producing the ensemble of disease outcome shown in Figure 2.2. We consider log of the weekly cumulative number of disease incidence as the simulator output, yielding a vector of length 57 in each replicate of the simulation run. For the univariate setting we define our QOI as the total number of disease incidence after week 20. Hence, we acquire a dataset of size $100 \times 100 = 10,000$ from all the runs.

As the first step we pre-process the data and compute the α-quantiles $\{\eta_1(\boldsymbol{\theta}), \ldots, \eta_q(\boldsymbol{\theta})\}$ from a corps of model output $\{\eta^{(1)}(\boldsymbol{\theta}), \ldots, \eta^{(r)}(\boldsymbol{\theta})\}$. This pre-processing is repeated for each input setting $\boldsymbol{\theta}$ in the design. Next, we use the same GP formulation described in sec 2.1.1 to build an emulator whose input is the tuple $(\boldsymbol{\theta}, \alpha)$ and output $\eta_\alpha(\boldsymbol{\theta})$.

We consider the ensemble of ABM simulation run in Figure 2.3 derived from Figure 2.2. $N = 100$ simulations are considered at 5 different settings for the standardized parameter with values $\boldsymbol{\theta} = 0, 0.25, 0.5, 0.75, 1$ (the remaining ABM parameters are fixed at their nominal values). The log of total epidemic counts at 20 weeks is considered to be the univariate model response $\eta(\boldsymbol{\theta})$ and our QOI.

Table 2.1: Disease ABM parameters and their ranges.

parameter	description	lower	upper
θ_1	transmissibility	3×10^{-5}	8×10^{-5}
θ_2	initial infected number	1	20
θ_3	hospital intervention delay	2	10
θ_4	hospital intervention efficacy	0.1	0.8
θ_5	intervention travel reduction	0	2

Next, for each setting of $\boldsymbol{\theta}$, $q = 5$ values of the quantile η_α are estimated, with $\alpha = .05, .275, .5, .725, .95$, using the $N = 100$ model responses, producing an effective simulation output $\eta(\boldsymbol{\theta}, \alpha) = \eta_\alpha(\boldsymbol{\theta})$ for each tuple $(\boldsymbol{\theta}, \alpha)$ in the ensemble. Red circles in Figure 2.4 shows the univariate response at each of the 5 parameter settings.

Using the Bayesian GP emulator formulation, this data is modeled as a realization from a stationary GP, producing estimated posterior mean surface shown in Figure 2.4. These empirical quantiles are estimated based on $N = 100$ realizations. Hence an error term (or nugget) would allow one to account for the uncertainty in estimating the unknown quantiles from few realizations as suggested in [121]. Consequently, the posterior mean surface from fitted GP should not interpolate these

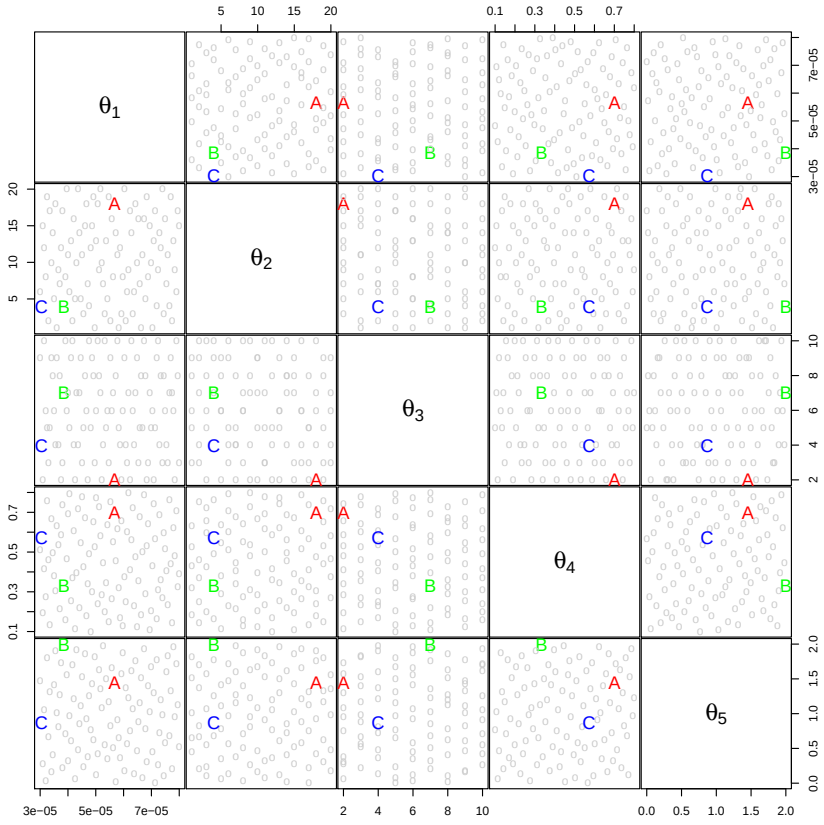


Figure 2.1: The figure shows a OA-based LH design of $m = 100$ points in 2d projection. The three points denoted by A, B and C represents three inputs which will be used later for holdout experiment.

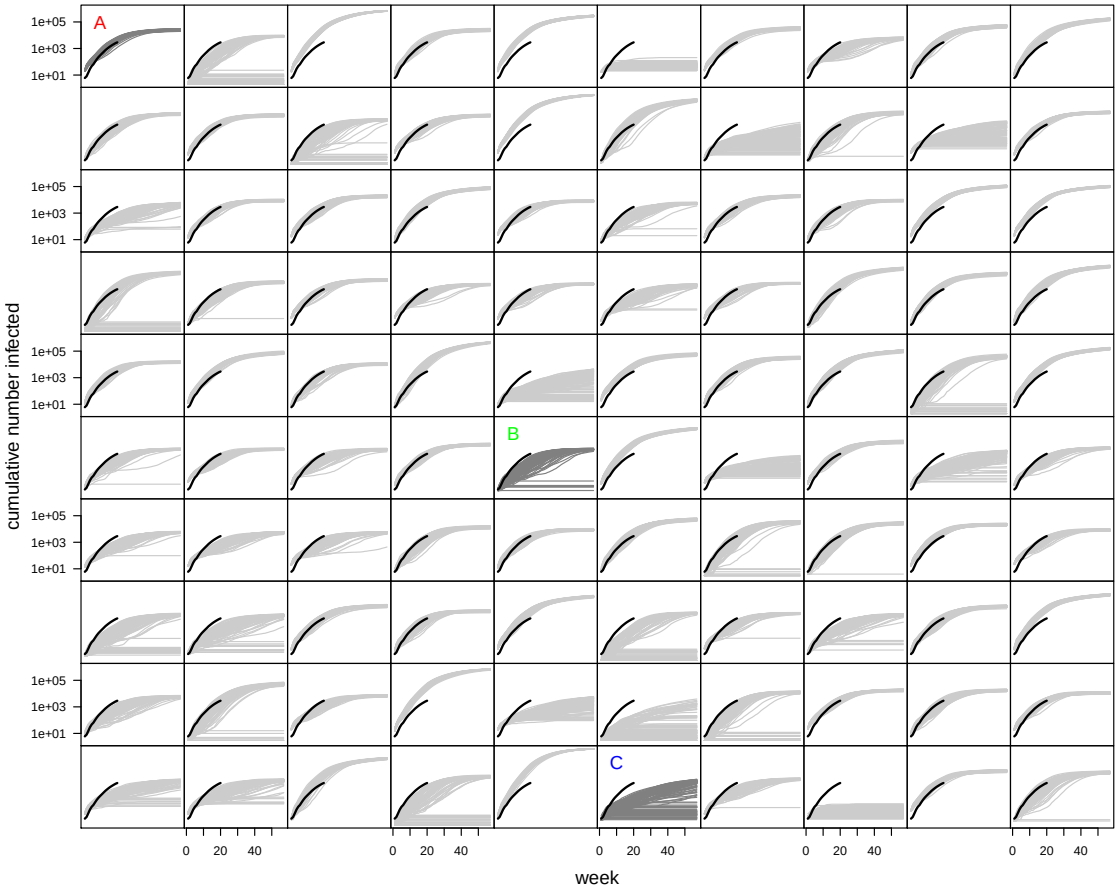


Figure 2.2: The grey lines in each box show the simulated epicurve for 57 weeks at a fixed input setting, where the simulation generates 100 replicates for each input setting from the design in Fig 2.1. The first 20 weeks of epidemic observations are shown in solid black lines – the same black line is shown in all 100 boxes. The three boxes marked by A, B and C correspond to the input locations held out for out of sample experiment.

quantile estimates. Note that in this example our choice of number of quantiles has been $q = 5$ for training data; but other choices could be made as deemed appropriate for the number of quantiles. Next, we extend this approach to account for multivariate output for the same application by embedding this model within an encompassing BMC framework.

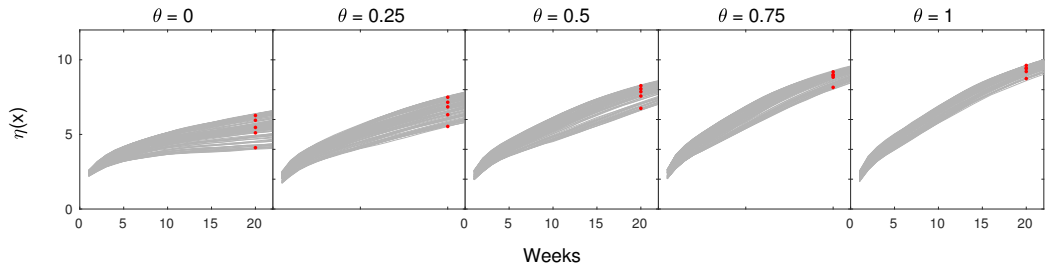


Figure 2.3: The figure shows a set of 100 realizations of epidemic ABM simulations each at 5 different input settings. The estimates of the empirical quantiles $Q(\alpha)$ of the logged total epidemic counts at week 20 with $\alpha = .05, .275, .5, .725, .95.$ are shown in red dots.

2.2.2 Multivariate Response

In the actual Ebola application, we consider the QOI to be the full time series of length 57 of the log of weekly cumulative disease incidence counts. The full epidemic simulator is used here with an input dimension of 5 (see Table 2.1) and output dimension of 57. We also continue to work with the previously mentioned experimental design of size $m = 100$ drawn from a space-filling, symmetric latin hypercube design (see Fig 2.1), corresponding to the full simulation output in Fig 2.2 yielded by running the simulator 100 times at each of the 100 input setting. This produces an ensemble of 10,000 simulated time series of logged counts. The reported counts till week 20 is considered as the observation to calibrate the ABM and make future predictions. These observed cases are shown by the black line in each box in Figure 2.2.

It is important to note here that for different input parameter locations, the ensemble of replicated simulations can show very different epidemic behavior. In Fig 2.2, some frames shows the grey lines being tightly concentrated around their mean line, and in some other frames, the distribution of those are not unimodal. Some of those simulations suggest no epidemic, and some of them shows ever-increasing throughout the year. Since the replicate distribution varies as a function of input $\boldsymbol{\theta}$, it restricts one to model such distribution as $N(\mu(\boldsymbol{\theta}), \Sigma(\boldsymbol{\theta}))$.

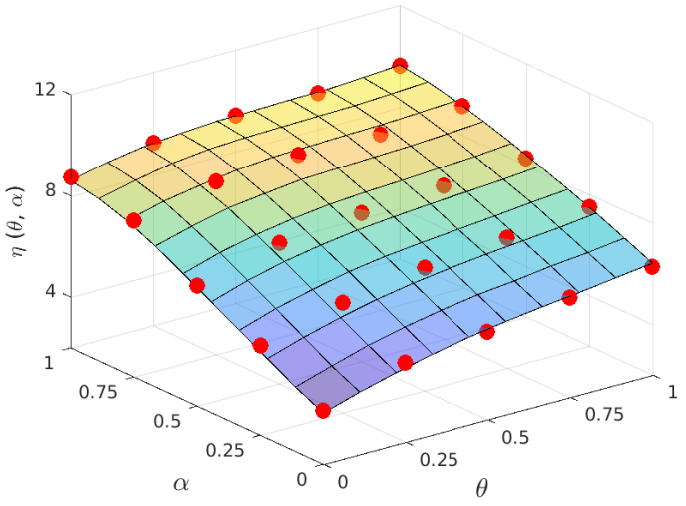


Figure 2.4: Posterior mean function of the ABM simulator output conditioned on the univariate response from simulated ABM as shown by then red circles, as a function of input θ and quantile α. These red circles are the same red dots in Figure 2.3

Pre-processing

It is apparent for the Figure 2.2 that not all simulations are consistent with the observation, in fact, not all replicates of some simulations are consistent to the observations. In some frames, only a few replicates can be seen agreeing to the ground truth, and in other a good number of replicates seems reasonable. To get an accurate prediction of the epidemic after 20 weeks, it is important to collect all the input settings as well as the combination of replicates that are plausible given the observation until week 20. Hence, we require to index the epidemic curves by replicate, and the input parameter $\boldsymbol{\theta}$. We adopt the quantile kriging [121] approach to do so.

Unlike in the univariate set-up, defining the empirical quantiles is tricky for multivariate output, and often is not well defined. One strategy could be to order the replicates with respect to the cumulative case count until week 20 (since we observe data until week 20), and then find the n_α qunatiles, in which case one can not guarantee the ordering would be preserved for the entire time period. Another strategy one may think of is to order and obtain the quantiles at each time point, and later join them to produce the quantile curves. In the later case, although the quantile curves

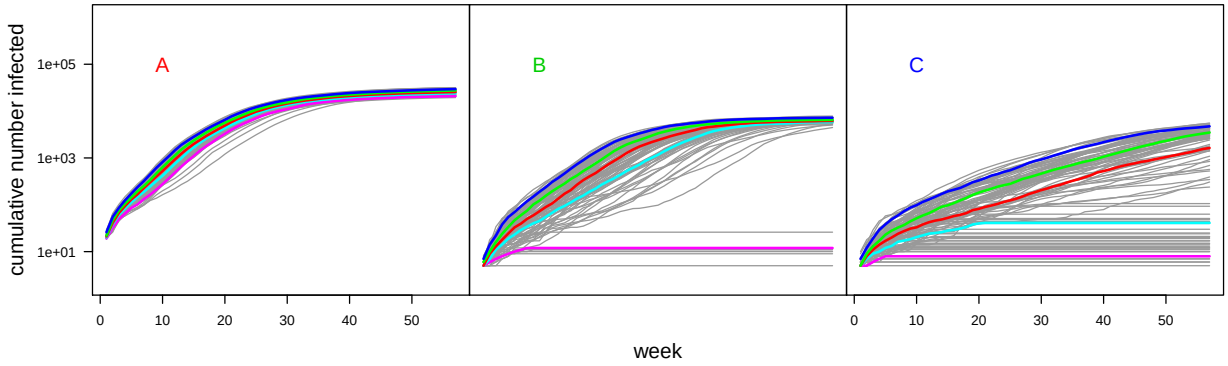


Figure 2.5: The grey lines are the actual trajectories simulated from the ABM, showing the cumulative number of disease counts for 57 weeks at 3 different input locations $\boldsymbol{\theta}$. The five lines in five colors (magenta, cyan, red, green, blue respectively) show the $.05, .275, .5, .725, .95$ quantiles estimated from the grey lines.

may not and will not certainly correspond to any of the actual simulation replicates, they do look like similar and are comparable to simulated epidemics. The pointwise $\alpha = (.05, .275, .5, .725, .95)$-quantiles estimated from 100 replicates are shown in Figure 2.5 for 3 different values of $\boldsymbol{\theta}$.

The simulation runs consisting of $n_r = 100$ replicates for each of $m = 100$ parameter locations is now reduced to an ensemble of $n_\alpha = 5$ quantile trajectories for each of $m = 100$ settings, effectively making our dataset smaller in size. This new ensemble is then indexed by an augmented input parameter ($\boldsymbol{\theta}$,α) of dimension $p + 1$ (see Eq.(2.4)).

$$\begin{pmatrix} \theta^*_{11} & \cdots & \theta^*_{1p} & \alpha_1 \\ \vdots & \ddots & \vdots & \vdots \\ \theta^*_{11} & \cdots & \theta^*_{1p} & \alpha_{n_\alpha} \\ \vdots & \ddots & \vdots & \vdots \\ \theta^*_{m1} & \cdots & \theta^*_{mp} & \alpha_1 \\ \vdots & \ddots & \vdots & \vdots \\ \theta^*_{m1} & \cdots & \theta^*_{mp} & \alpha_{n_\alpha} \end{pmatrix}. \tag{2.4}$$

The choice of $n_\alpha = 5$ is decided after exploring different scenarios using a variety of choices for n_α.

A small number of quantiles certainly lessen the computational burden of fitting a GP emulator and subsequent calibration formulation; a large number of quantiles suggests that the emulator need not interpolate over larger distances in the α-component of the input space. Our prior experiments indicated that the prediction outcome do not change significantly for $n_\alpha \geq 5$. Hence, we made a choice considering the computational cost.

Given the required ingredients, there are different number of modeling framework available to perform a calibration and prediction analysis (see [11, 42, 76, 118]). The ingredients include 1) a $(m \cdot n_\alpha) \times (p + 1)$ experimental design of input parameters, 2) an ensemble of ABM response (57-vector of weekly logged cumulative cases) corresponding to the design, and 3) an observation giving the log cumulative cases until the first 20 weeks of the epidemic. We adopt the GPMSA [57] as our choice of calibration framework.

Model Formulation

The basic idea in GPMSA [57] is based on the KOH set up where the vector-valued observation $\boldsymbol{y}$ is modeled as the sum of the computational model $\eta(\hat{\boldsymbol{\theta}}, \hat{\alpha})$ at the best setting $(\hat{\boldsymbol{\theta}}, \hat{\alpha})$, a systematic model discrepancy term δ, and observation error $\boldsymbol{\epsilon}$:

$$\boldsymbol{y} = \eta(\hat{\boldsymbol{\theta}}, \hat{\alpha}) + \delta + \boldsymbol{\epsilon}, \tag{2.5}$$

where each of $\eta(\boldsymbol{\theta}, \alpha)$, δ and ϵ are vectors of length 57, and $\boldsymbol{y}$ is a vector of observations of length 20. The uncertainty in the prediction from the model arises due to lack of knowledge of the best parameter values $(\hat{\boldsymbol{\theta}}, \hat{\alpha})$, having a limited number of ABM runs, requiring one to estimate $\eta(\boldsymbol{\theta}, \alpha)$ for input settings not used in creating the corps of runs, unknown hyperparameters in the statistical model, systematic bias δ between the computational model and the actual epidemic process ζ, and observational error which is modeled by $\boldsymbol{\epsilon}$.

A GP emulator for η is trained in order to account for the uncertainty in the prediction of η at

untried settings $(\boldsymbol{\theta}', \alpha')$ in the analysis. One can incorporate the appropriate estimation error due to interpolation and prediction by using an GP prior for such emulator. To handle the multivariate nature of the model response and the data, a basis decomposition of η [76] is used:

$$\eta(\boldsymbol{\theta}, \alpha) = \phi_0 + \sum_{k=1}^{p_\eta} \phi_k w_k(\boldsymbol{\theta}, \alpha) + \epsilon_{w0}, \tag{2.6}$$

where p_η denotes the number of basis functions used in the representation and ϵ_{w0} accounts for error due to finite number of basis functions used, ϕ_0 denotes the overall mean, and ϕ_ks are the basis functions. Here we consider $p_\eta = 5$ for the analysis. The basis functions are obtained from the eigenvectors (or empirical orthogonal functions [151]) from the ensemble of epidemic trajectories (Fig 2.6). For the sake of simplicity, we would consider α a part of the parameter vector $\boldsymbol{\theta}$ going

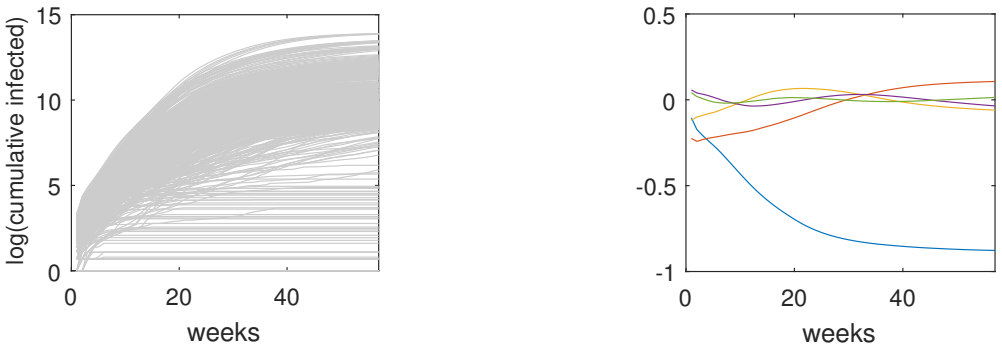


Figure 2.6: Left: The ensemble of ABM output from the experimental design (Fig 2.1); right: 5 basis functions obtained from the ABM output after centering and standardizing.

forward, and hence we redefine the $\boldsymbol{\theta}$ vector as $\boldsymbol{\theta} = (\theta_1, \cdots, \theta_5, \alpha)$. The error vector ϵ_{w0} is given a $N(0, \lambda_{w0}^{-1} I)$ prior distribution. Each of the 5 basis weights $w_i(\boldsymbol{\theta})$, $i = 1, \ldots, p_\eta$, is then modeled as a GP with mean 0:

$$w_i(\boldsymbol{\theta}) \sim \text{GP}(0, \lambda_{wi}^{-1} R(\boldsymbol{\theta}, \boldsymbol{\theta}'; \rho_{wi})), \tag{2.7}$$

where λ_{wi} controls the marginal precision of the process and the correlation function is given by

$$R(\boldsymbol{\theta}, \boldsymbol{\theta}'; \rho_{wi}) = \prod_{k=1}^{p_\theta} \rho_{wik}^{4(\theta_k - \theta_k')^2}. \tag{2.8}$$

Each ϕ_i is scaled in such a way so that the marginal variance of $w_i(\cdot)$ should become close to 1. The priors for the λ_{wi}s hence are set to be $\Gamma(5, 5)$, to emphasize the mass being centered around 1. Under this parameterization of the Gaussian covariance, ρ_{wik} gives the correlation between $w_i(\boldsymbol{\theta})$ and $w_i(\boldsymbol{\theta}')$ when the input settings $\boldsymbol{\theta}$ and $\boldsymbol{\theta}'$ are identical, except for a difference of half the prior range of the kth component. More details about the covariance function can be found in [57].

Restricting ourselves to the m input settings used for the initial model runs we can define,

$$w_i = (w_i(\boldsymbol{\theta}_1^*), \ldots, w_i(\boldsymbol{\theta}_m^*))', \quad i = 1, \ldots, p_\eta.$$

Then, $w = (w_1', \ldots, w_{p_\eta}')'$ then has prior distribution

$$\begin{pmatrix} w_1 \\ \vdots \\ w_{p_\eta} \end{pmatrix} \sim N\left(\begin{pmatrix} 0 \\ \vdots \\ 0 \end{pmatrix}, \Sigma_w = \begin{pmatrix} \lambda_{w1}^{-1} R(\boldsymbol{\theta}^*; \rho_{w1}) & 0 & 0 \\ 0 & \ddots & 0 \\ 0 & 0 & \lambda_{wp_\eta}^{-1} R(\boldsymbol{\theta}^*; \rho_{wp_\eta}) \end{pmatrix} \right), \tag{2.9}$$

where $R(\boldsymbol{\theta}^*; \rho_{wi})$ is obtained by applying the Gaussian covariance function Eq.2.8 on the design matrix in (2.4). The ABM output $\eta(\boldsymbol{\theta}_j^*)$ is projected (via dot product) onto each basis vector ϕ_i, giving transformed output $w_i^* = (w_i^*(\boldsymbol{\theta}_1^*), \ldots, w_i^*(\boldsymbol{\theta}_m^*))$. These transformed simulations are then modeled as independent normal perturbations from the w_i vectors:

$$w_i^* \sim N(w_i, \lambda_{w\epsilon i} I_m), i = 1, \ldots, p_\eta. \tag{2.10}$$

In usual settings, the "nugget" precisions $\lambda_{w\epsilon i}$ are typically set to be quite large, so that the posterior produced $w_i(\cdot)$s are very close to the projected simulations w_i^*. Since the quantiles are empirically estimated from limited model replicates, the posterior values for $\lambda_{w\epsilon i}$ are at a level where they

don't enforce exact interpolation.

For the rest of the parameters, we specify independent, diffuse $\Gamma(1, .0001)$ priors for $\lambda_{w\epsilon i}$ and λ_{w0}, allowing the information in the data and simulations to constrain their posterior values. The independent $\Gamma(5, 5)$ priors for each λ_{wi} puts prior mass near 1 which is consistent with the scaling used in the basis representation of equation (2.6). And the independent beta($a_{\rho_w} = 1, b_{\rho_w} = 0.1$) priors for the ρ_{wik}'s encourage effect sparsity since prior mass is concentrated at 1 [76].

$$\begin{aligned}
\pi(\lambda_{w0}) &\propto \lambda_w^{a_{w0}-1} e^{-b_{w0}\lambda_{w0}}, \\
\pi(\lambda_{w\epsilon i}) &\propto \lambda_{w\epsilon i}^{a_{w\epsilon}-1} e^{-b_{w\epsilon}\lambda_{w\epsilon i}}, \quad i = 1, \ldots, p_\eta, \\
\pi(\lambda_{wi}) &\propto \lambda_{wi}^{a_w-1} e^{-b_w\lambda_{wi}}, \quad i = 1, \ldots, p_\eta, \\
\pi(\rho_{wik}) &\propto \rho_{wik}^{a_{\rho_w}-1}(1-\rho_{wik})^{b_{\rho_w}-1}, \quad i = 1, \ldots, p_\eta, \; k = 1, \ldots, p_\theta.
\end{aligned}$$

Fig 2.7 and 2.8 show the 2 dimensional mean posterior response surface based on the GP prior. In particular, Fig 2.7 describes the decomposition of the multivariate output in terms of the principal components ϕ_k and the basis loadings $w_k(\boldsymbol{\theta})$.

Once, the GP emulator is trained, we can assess the accuracy of this new quantile based emulator by predicting the model output for three holdout desing points show in Figures 2.1 and 2.2 by three colors. Fig 2.10 shows the resulting simulated and predicted model outputs at three holdout input parameter settings denoted by A, B and C in Fig 2.1. Each row in this figure correponds to one of the holdout design points. The first column shows all the replicates from the simulation run along with the 5 estimated empirical quantiles. The next 5 columns shows the pointwise 90% credible intervals for each of the 5 quantiles, along with the estimated quantiles from actual simulation runs.

One can also look into the sensitivity of the simulator response with respect to it's inputs. By holding all but one input at a nominal setting, the emulator is run at various settings of that particular input and Fig 2.11 shows posterior means for the ABM response $\eta(\cdot)$ for each of the input parameters. This exercise relays an idea about the simulator as in how the multivariate

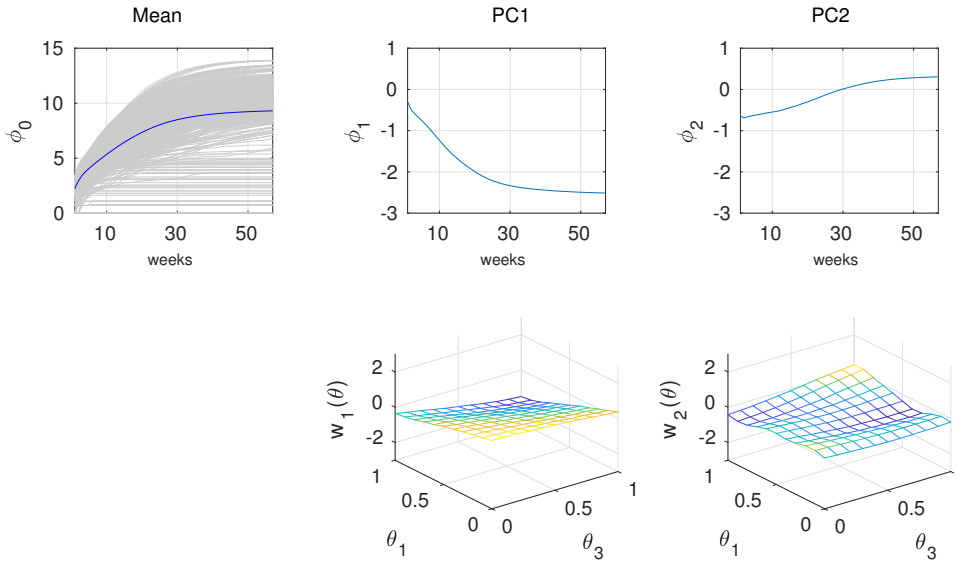


Figure 2.7: The top row shows the simulations in the left frame and first and second principal component bases in the right frame, obtained from the simulation output, whereas, the the posterior mean surfaces are for basis loadings $w_i(\boldsymbol{\theta}), i = 1, 2$ with respect to θ_1 and θ_3 in bottom row. Here the other fours parameters were held at their nominal values as θ_1 and θ_3 vary over their predefined range.

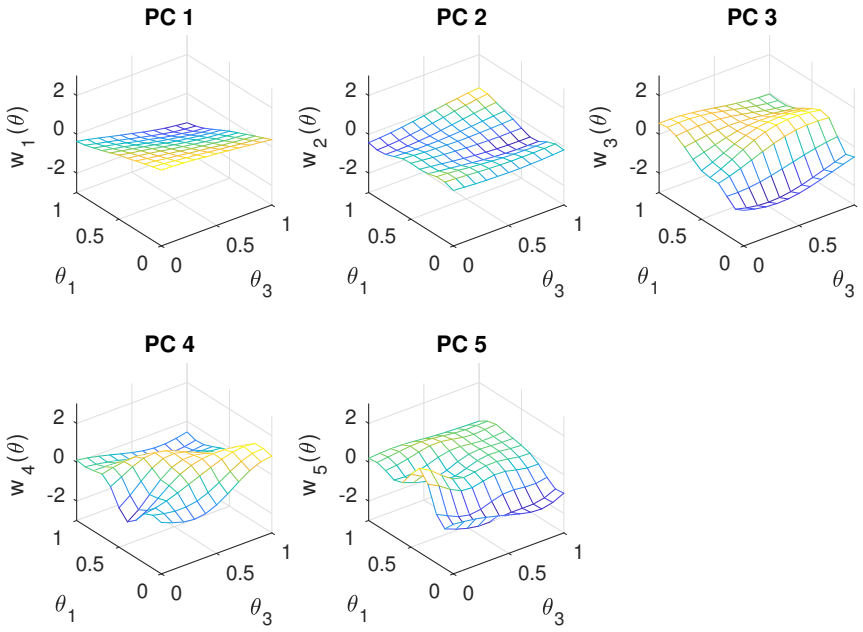


Figure 2.8: The figure shows 2-D posterior mean surface of all 5 bases w_i with respect to two input components θ_1 and θ_3, as obtained from GP emulator.

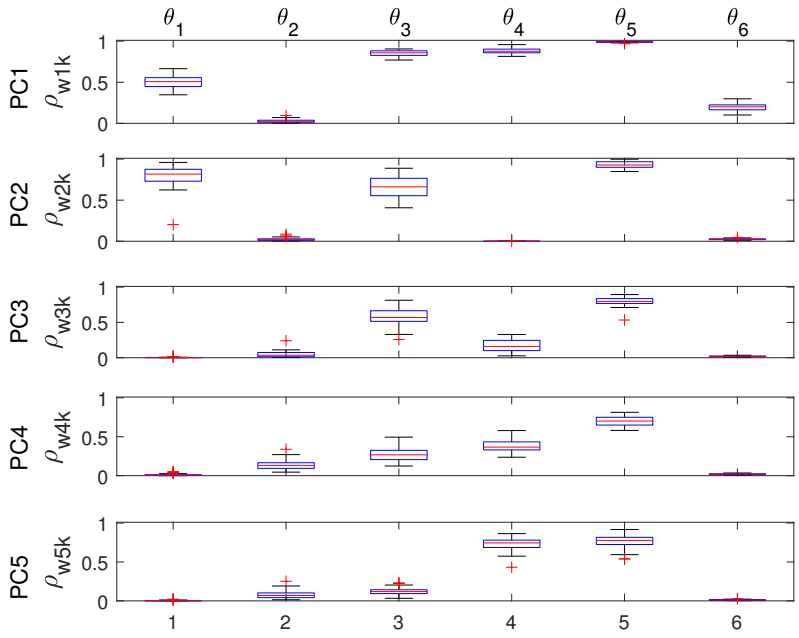


Figure 2.9: Posterior distribution of the correlation parameters ρ_{wk} in the covariance function (Eq. 2.8 for 5 bases.

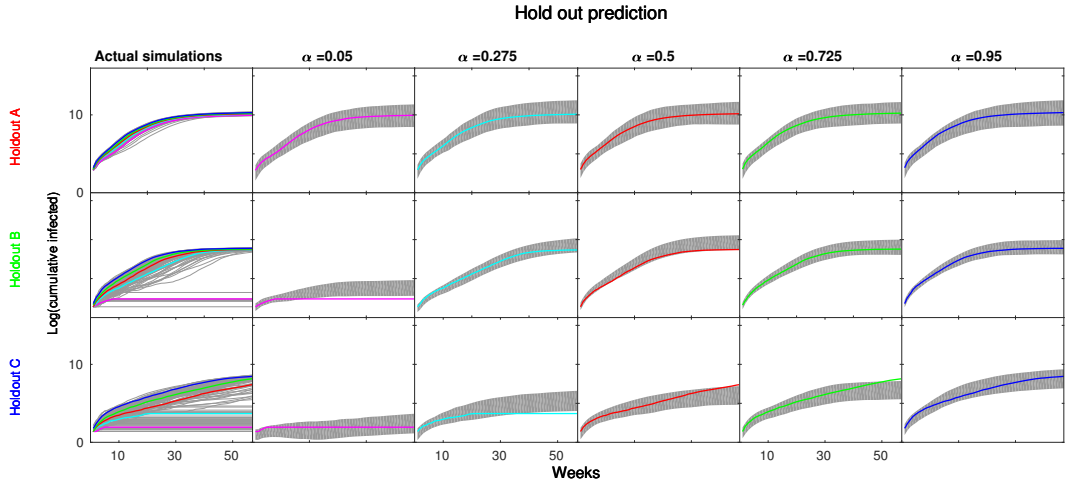


Figure 2.10: On the left most column, grey lines show the actual simulations and corresponding empirical quantiles at the pre-chosen inputs locations, denoted by the letters A, B and C and the colors from the design in Fig 2.1. The next 5 columns show pointwise 90% credible intervals of the posterior emulator prediction of quantiles at those model inputs. These ABM runs were held out from the training dataset used to train the GP emulator. Each of these five columns compares the empirical quantiles, denoted by different colors, from the actual simulations with the posterior predictions.

simulator response behaves near the posterior mean. Other marginal functionals of the simulation response can also be calculated such as sensitivity indicies or estimates of the Sobol decomposition [112, 131]. While Fig 2.11 shows the main effects of the input parameters on the multivariate model output, it is clear that the effect of the quantiles α on the model output varies along with the size of epidemic which is also controlled by the other model inputs, and it emphasizes the use of techniques to take care of input dependent variations. Fig 2.9 aids to the understanding of the sensitivity the simulator along each direction of the input parameter. The higher the values of ρ_{wk} are, the less sensitive is the simulator output with respect to the corresponding input. For example, a small change in θ_2 can result in a big change in the simulator output, whereas, θ_5 seems to have least effect on our QOI.

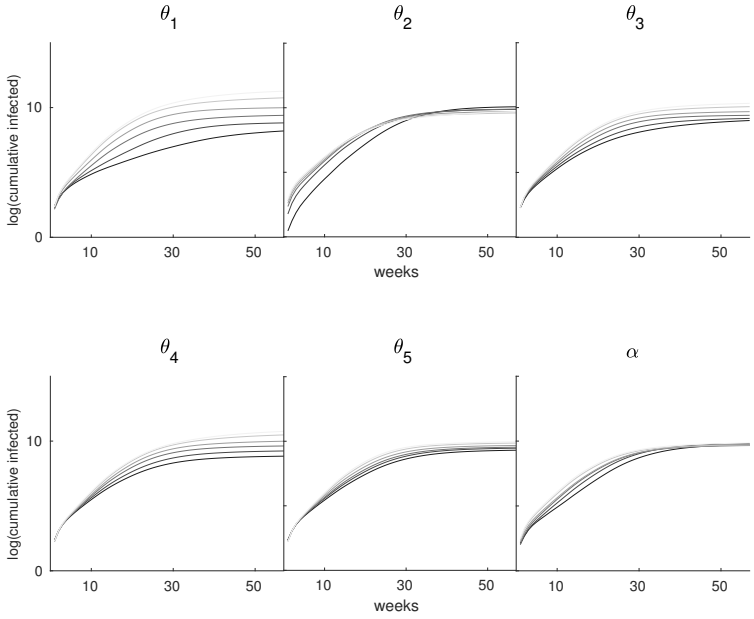


Figure 2.11: Each frame shows the posterior mean predictions (log counts as a function of time) from the GP emulator varying one input across its prior range, while holding others at their posterior mean values.

2.2.3 Calibration

Having a GP emulator for the stochastic ABM ready to use, the rest of the task is really just to incorporate this emulator into Eq. 2.5, define a model for the discrepancy term and perform calibration in order to find the posterior distributions for $\boldsymbol{\theta}$ and subsequently the posterior distributions for model output η. The discrepancy term δ is modeled, like the emulator, using a basis representation over time: 1 week $\leq$ time $\leq$ 57 weeks, as a smoothing spline [74]. The spline bases are simply normal kernels with a standard deviation of 15 weeks, and are spaced 10 weeks apart. Fig 2.12 shows the spline kernels between week 1 and week 20, which are used as the bases for discrepancy modeling.

$$\delta = \sum_{k=1}^{p_\delta} d_k v_k, \tag{2.11}$$

where $p_\delta = 7$ and each v_k has a zero mean normal prior with precision λ_δ. We use a diffuse $\Gamma(1, .0001)$ distribution prior for λ_δ

$$\pi(\lambda_\delta) \propto \lambda_\delta^{a_\delta - 1} e^{-b_\delta \lambda_\delta}.$$

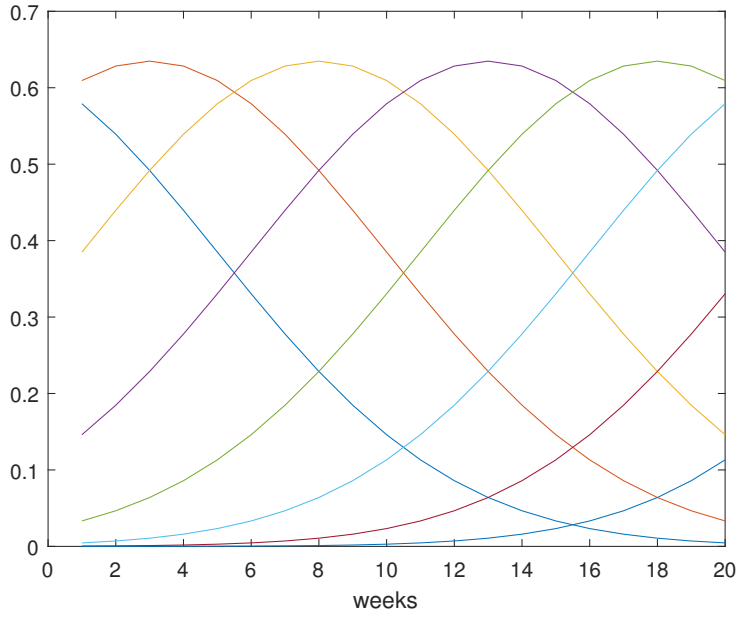


Figure 2.12: Discrepancy kernel δ_k between week 1 and week 20 (observation period).

The construction of the likelihood function is not straight forward for multiple reasons. The epidemic data arrives as recorded counts of Ebola incidences from surveillance reports obtained from various counties in Liberia, and the counts were aggregated up to the country level. The challenging and poor surveillance procedure [156] during the epidemic indicates the possibility of significant errors in the reported counts at the first place. Also, early disease incidence counts are believed to be substantially lower than the actual cases due to lack of awareness among the health professionals and general population[31, 116].

Generalized linear model using a log link and poisson distribution [104] is a popular and to-go choice to model count data. But the assumptions of independence among the observations may

not be appropriate for epidemic data due to dependence between infected individuals. Hence, we opt for a simple Gaussian error model with mean 0 and a standard deviation roughly 20% of the expected count, which has already been used and tested by epidemiologists [149].

To connect the observations y to the log of cumulative counts produced by the ABM emulator, we build a Gaussian model for the log of the cumulative, observed counts. Denoting the weekly (non-cumulative) counts to be $c = (c_1, \ldots, c_{20})^T$, we specify the corresponding vector of independent errors $e = (e_1, \ldots, e_{20})^T$ with $E(e_k) = 0$ and $\text{sd}(e_k) = \max(5, .2c_i)$, so that $E(e) = 0$ and $\text{Var}(e) = V_e$, where V_e is the diagonal matrix defined by the individual standard deviations. The error expectation and error variance for the log of the cumulative counts can then be approximated as $E(\epsilon) = 0$ and $\text{Var}(\epsilon) = \Sigma_y$ using propagation of error [24]. We allow for uncertainty in the specification of the error covariance matrix for y by including a scaling precision term λ_y so that $\text{Var}(y) = \lambda_y^{-1}\Sigma_y$. The prior for λ_y is $\Gamma(5, 5)$, aligning with our prior expectation of 20% errors in the observed counts. Treating λ_y as a parameter allows this error size to adjust to be more consistent with the observations.

Given the cumulative reported cases recorded in the $n_y = 20$-vector y, we model the log likelihood as

$$\ell(\boldsymbol{\theta}, \lambda_y; y, \eta(\cdot), \delta, \Sigma_y) = \frac{n_y}{2}\log\lambda_y - \frac{1}{2}\Big(y - \eta(\boldsymbol{\theta}) - \delta\Big)^T \lambda_y \Sigma_y^{-1}\Big(y - \eta(\boldsymbol{\theta}) - \delta\Big).$$

Combining this likelihood with the emulator and the discrepancy model, we can rewrite the above equation as:

$$\begin{aligned}\ell(\boldsymbol{\theta}, \lambda_y, w(\cdot), v, \lambda_w, \lambda_\delta, \rho_w, ; y, \phi, d, \Sigma_y) &= \frac{n_y}{2}\log\lambda_y \\ -\frac{1}{2}\Big(y - \phi_0 - \sum_{k=1}^{p_\eta}\phi_k w_k(\boldsymbol{\theta}) - \sum_{k=1}^{p_\delta} d_k v_k\Big)^T &(\lambda_y\Sigma_y^{-1} + \lambda_{w0} I)\Big(y - \phi_0 - \sum_{k=1}^{p_\eta}\phi_k w_k(\boldsymbol{\theta}) - \sum_{k=1}^{p_\delta} d_k v_k\Big).\end{aligned} \tag{2.12}$$

The full likelihood is followed from Eq.(2.12) and Eq.(2.7), and the posterior distribution for the unknown parameters is obtained using the likelihood and the prior specifications discussed above. The posterior predictive distribution for the emulator and the epidemic curves [57, 76] can then

be obtained using the posterior samples which are produced by standard, single site MCMC in GPMSA. We use th metropolis updates [106] for the components of ρ and $\boldsymbol{\theta}$ with uniform proposals centered at the current value of the parameter. The precision parameters λ_w, λ_{w0}, $\lambda_{w\epsilon}$, λ_y and λ_v are sampled using Hastings updates [71]. Here the proposals are uniform draws, centered at the current parameter values, with a width that is proportional to the current parameter value. The widths/scalings of the update proposals are initialized at .3, and then tuned using an initial pilot run [67].

2.2.4 Results

Fig 2.13 shows the posterior for the input parameters to the ABM simulator along with the augmented input quantile $(\boldsymbol{\theta}, \alpha)$. The diagonal shows the marginal posterior distribution for individual parameters. Comparing to the sensitivity plots in Fig 2.9 and 2.11, these 1-D marginal plots convey similar information regarding the relative importance of input parameters when they are constrained by 20 week observation. The off diagonal plots of the 2-D marginal distributions indicates the dependence among the input parameters as expected. We specially note the importance of the dependent relation between the quantile α and other input parameter $\boldsymbol{\theta}$, as, given data some parameter settings are only plausible for high values of α and some require a low value of α to be compatible. The inclusion of α as a calibration parameter helps to correctly allocate the posterior probabilities to $\boldsymbol{\theta}$ by integrating over α. Otherwise, one would get wrong in assigning probabilities to $\boldsymbol{\theta}$ only based on the mean of the 100 replicates, where, for example, a parameter setting may end up getting very low posterior probabilities, even though that setting produces some realizations that are close to the data. Such consideration of full parameter uncertainty is important in order to manage an epidemic where the policy makers would like to consider all possible outcome given the present scenario.

The first two frames of Fig 2.14 show the posterior pointwise 90% intervals for the calibrated simulator $\eta(\boldsymbol{\theta})$ and discrepancy δ. The last frame shows the 90% intervals for actual system or the

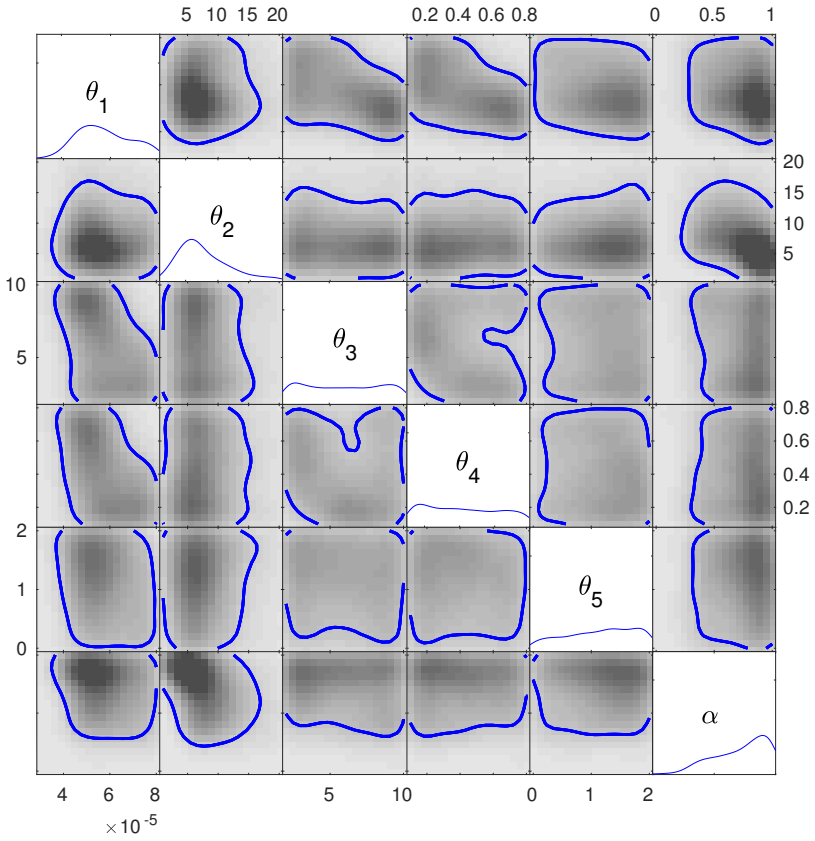


Figure 2.13: The figure shows estimated posterior distribution of the input parameters $(\theta_1, \cdots, \theta_5, \alpha)$. The diagonal shows the estimated marginal posterior pdf for each parameter; and the off-diagonal images give estimates of bivariate marginals; to contour lines show estimated 90% hpd regions.

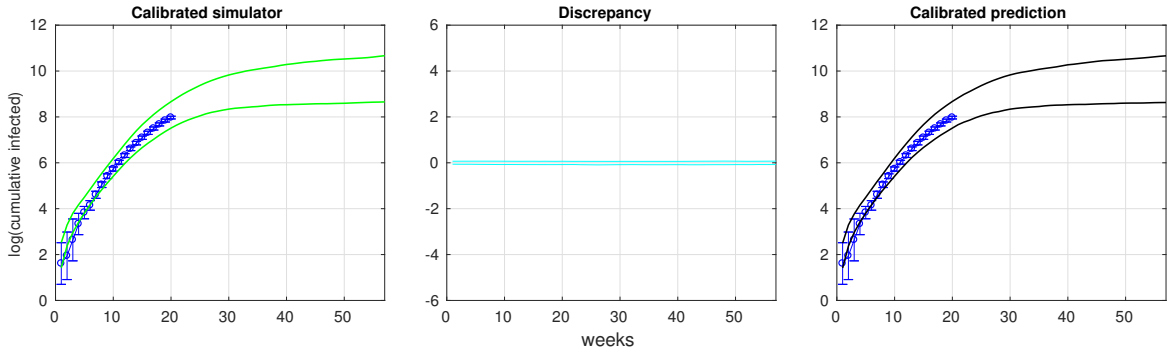


Figure 2.14: The figure shows posterior prediction decomposition into emulator and discrepancy. Left: the posterior 90% uncertainty for the calibrated emulator $\eta(\boldsymbol{\theta})$ prediction is shown. The aggregated uncertainty is due to unknown $\boldsymbol{\theta}$ and the GP emulator of ABM. Middle: posterior 90% uncertainty for δ. Right: posterior 90% intervals for the actual epidemic curve: $\eta(\boldsymbol{\theta}) + \delta$. The blue dots show the log of observed counts, along with 1 standard deviation bars given by the square root of diag(Σ_y).

epidemic process $\eta(\boldsymbol{\theta}) + \delta$. While Fig 2.13 may not suggest a significant learning regarding all the input parameters from the data, the resulting reduction in uncertainty is evident from the posterior epidemic curves (Fig 2.15, left frame). The posterior distribution of the epidemic outcome accounts for uncertainty in the parameters, model discrepancy, uncertainty in the emulation of the ABM, and observation error.

One can also obtain posterior predictions for a number of other quantities, such as peak timing of the epidemic, and the intensity at the peak time. Fig 2.15 shows the posterior epidemic curves in cyan in the left frame, and the posterior realizations of peak by blue dots in the middle frame, and the histogram of the peak timing. Note that, the curves in the middle frame show new disease incidence per week, obtained by exponentiating and differencing the cumulative curve in the left frame of figure.

This combined set up of calibration, prediction and UQ is finally applied to the Ebola challenge problem, which requires predictions of three seasonal targets at different weeks. They are,

- Peak timing – the week in which the epidemic increased by the highest number of cases;
- Peak week cases: the number of new infected individuals incurred on the peak timing week;

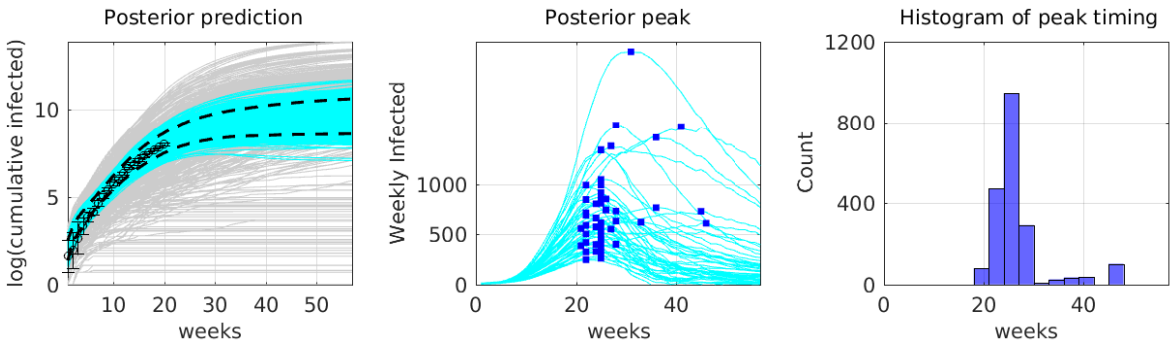


Figure 2.15: Left: Posterior and prior realizations of the epidemic curves are shown in grey and cyan respectively. The dashed lines represents the pointwise 90% credible intervals from the posterior predictions. Middle: 40 posterior realizations of the weekly counts over time. The blue dots show the peak weekly count and corresponding time. Right: histogram of posterior peak time realizations.

- Total epidemic size – the total number infected after the 57-week time period.

Top row of Fig 2.16 shows 90% credible interval (black dashed lines) for the log cumulative counts predicted at week 13, 26, 35 and 42 respectively. Note that the uncertainty on the future prediction gets narrower as we condition on more data points, however, there is not much constriction of the uncertainty at week 35 and 42. A reason could be that the epidemic is mostly over by week 30, hence, the observations after that do not add much information about the epidemic process to the posterior. The bottom frame shows the contribution of the discrepancy δ in the actual prediction $\eta(\boldsymbol{\theta}) + \delta$. Even though there is not significant contribution at week 13, 26 and 35, we do see a substantial negative discrepancy at week 42 to counter the ebola cases produced by the simulator at later time not consistent with the observations. The posterior for $\boldsymbol{\theta}$ also depends on the specification of δ [11, 22, 75, 88, 145].

The posterior distributions for the seasonal targets at different timepoints in the ebola challenge is given in Fig 2.17. As expected the posteriors are quite stable after week 26, though more careful inspection reveals that observations beyond week 26 do help in eliminating the extreme possibilities for peak timing and cases which receives positive probability at week 26.

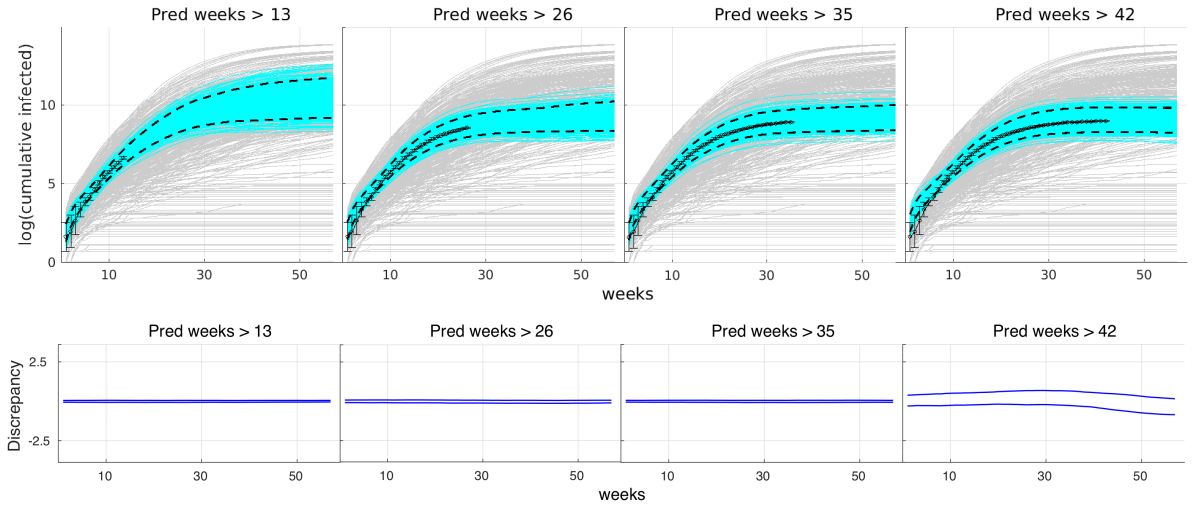


Figure 2.16: Top: Posterior realizations (cyan lines) and pointwise 90% credible intervals (dashed lines) for the actual epidemic curves ($\eta(\theta) + \delta$) for each of the ebola challenge time periods. The gray lines show epidemic curves produced from the initial ABM ensemble. Bottom: The corresponding pointwise 90% credible intervals of the discrepancy (δ) for each of the time periods. The more substantial discrepancy (from the analysis using data up to week 42) adjusts for simulations that produce more late-time cases than the observations suggest

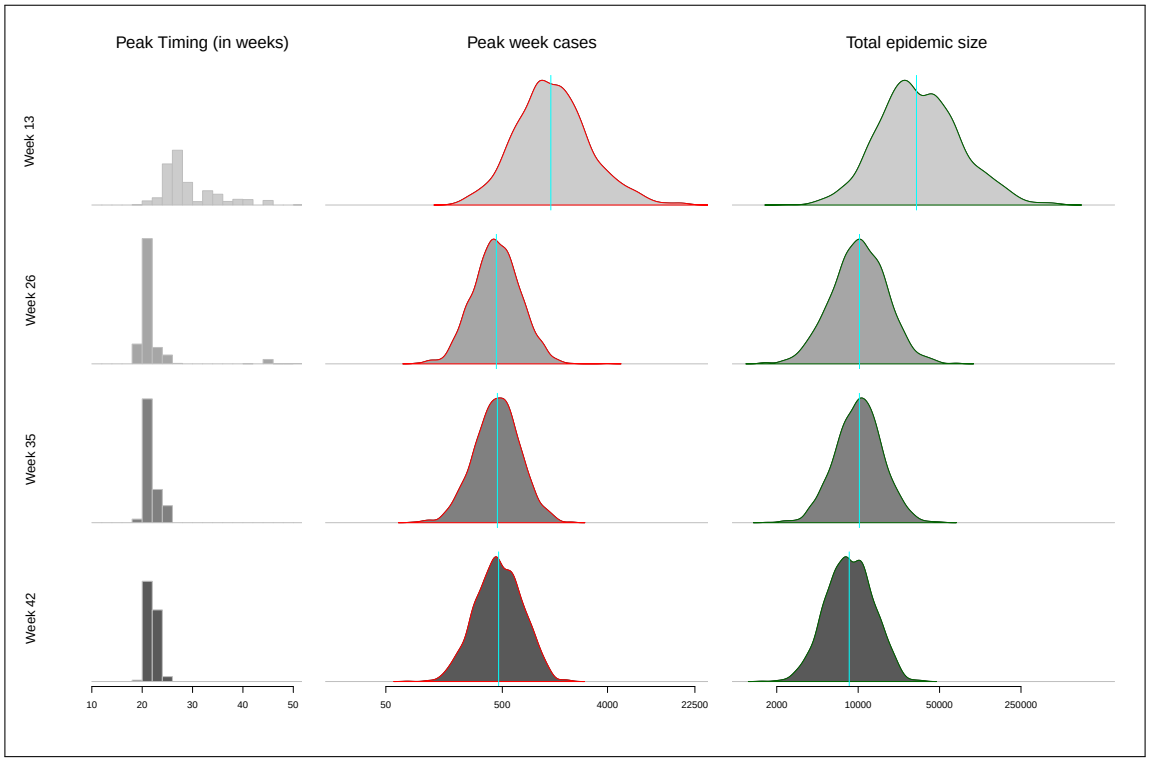


Figure 2.17: Posterior distributions for three functions of the actual epidemic curve – peak timing (left), peak weakly cases (middle), and total epidemic size (right). Each row shows the posterior density estimates conditional on epidemic data up to 13, 26, 35 and 42 weeks, moving from top to bottom.

Discussion

This method explains and demonstrates how quantile kriging idea can be adopted to emulate a stochastic computer model with multivariate output in order to perform sensitivity analysis, calibration, prediction and UQ in a Bayesian framework which accounts for uncertainties from simulation parameter settings being unknown, GP emulator prediction, systemetic difference between the simulation and reality and the stochastic nature of the ABM. Considering the computation perspective of the whole process, it allows to handle a sizeable amount of simulation output – 100 5-d input settings $\times$ 100 replicates $\times$ 57-week simulation output, within a reasonable time limit, e.g., it took roughly 20 minutes to draw 10,000 mcmc samples for all unknown parameters and hyper-parameters on a 8 core normal desktop.

We want to emphasize the difference between our solution and other existing methods to handle stochastic simulations which target to emulate the mean and covariance for the replicates as function of $\boldsymbol{\theta}$. Emulation of the mean functions is straightforward, but how to model the covariance matrix as a function of a high dimensional input space is clear. Moreover, that assumes normality for the replication variation, which is inaccurate for some input setting $\boldsymbol{\theta}$, as seen in Fig 2.2.

There are challenges and shortcomings as well in the proposed framework. While we use GP to model the dependence between quantiles, it does not enforce monotonicity: $\eta(\boldsymbol{\theta}, \alpha_1) \leq \eta(\boldsymbol{\theta}, \alpha_2)$ if $\alpha_1 \leq \alpha_2$. There are approaches available ensuring monotonicity [61, 100, 153], but their integration into general Bayesian model calibration formulations still remains vastly unexplored. Also, for replications, how to defined and estimate the empirical quantiles in general is not clear. Here we used pointwise quantile which produced trajectories quite comparable to actual simulations. But in other cases, one might require more care in computing the quantiles from multivariate output, such that a GP model can be used satisfactorily.

2.3 Calibration and Emulation via Finite Mixture modeling

While quantile kriging provides an avenue to account for replication variability in a stochastic computer model, specially when the gaussian assumption is not satisfied; a more intuitive and straight forward alternative may be a simple generalization of gaussian process emulator such as mixture of gaussian processes. Gaussian mixture model is an immensely popular technique [127] in statistics and machine learning for clustering [56], classification [79] and density estimation [48, 129] due to it's applicability, computational advantages, probabilistic inference and easy interpretability. However, a little or no study has been seen in the computer model calibration literature where one uses such approach to model a stochastic computer model emulator for the purpose of calibration and uncertainty quantification. In this section we propose and examine one such procedure based on gaussian mixture model on the same epidemic application described in section 1.3.

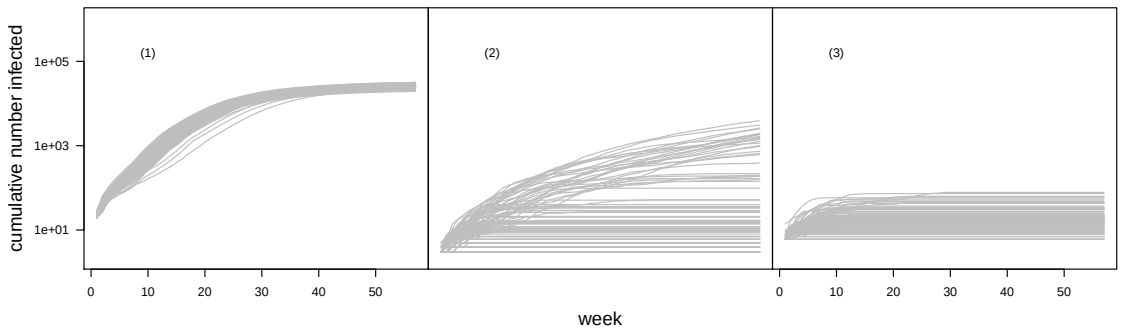


Figure 2.18: The grey lines show the simulated trajectories of the cumulative number of disease incidence for 57 weeks at three input setting.

By observing Fig 2.18 carefully, which shows the simulated epi-trajectories of cumulative infections at three different input settings, an underlying latent clustering of replicates is evident. Such phenomenon in a typical ABM disease simulation is quite common, where an infection can spread rapidly if it reaches a higher degree node in the social contact network - resulting in an outbreak, or may die off soon if infection happens only at few isolated places. One may then categorize each replicated from a simulation indexed by it's input to be in the *higher* or *lower* regime. For example,

the simulation in left frame in Fig 2.18 has all replicates in the higher regime, suggesting that the input setting at which the simulation was run most certainly produces a disease outbreak, whereas the input setting corresponding to the simulation in the right frame does not yield large number of infections almost all the time. But the simulation in the middle frame has representative from both higher and lower regime of the output space, indicating that the corresponding input setting may or may not result in a disease outbreak. This motivates us to adopt a modular approach in building an emulator for such computer model of bi-modal nature. The pivotal idea here is to use independent gaussian process surrogates for each regime of the output space, and a mechanism to predict the relative size of two regimes at any given input setting.

2.3.1 Univariate Response

Although there is no special treatment required for multivariate response case, we would still introduce the method for a simple univariate response setup in an attempt to present the multivariate response as a direct extension of this case. The modular approach is divided into two primary steps - (1) classifying the simulation output into two regimes and (2) modeling each regime using a gaussian process. Instead of weekly number of cumulative infections, we consider log of the total number of infections at the end of 57 weeks of simulation as the univariate response from the simulator. Fig 2.19 shows the distribution of univariate simulator output at 100 input locations.

Gaussian finite mixture model

In a model based clustering set up, one assumes that the components of a mixture density are usually associated with groups or clusters. Formally, if $\{x_1, \cdots, x_n\}$ are n independent samples from a finite mixture model with G (may be unknown) components, then the likelihood for each sample is written as

$$L(x_i|\boldsymbol{\pi}, \boldsymbol{\psi}) = \sum_{i=1}^{G} \pi_k f_k(x_i|\psi_k), \quad \pi_k > 0, \ \sum_{i=1}^{G} \pi_k = 1,$$

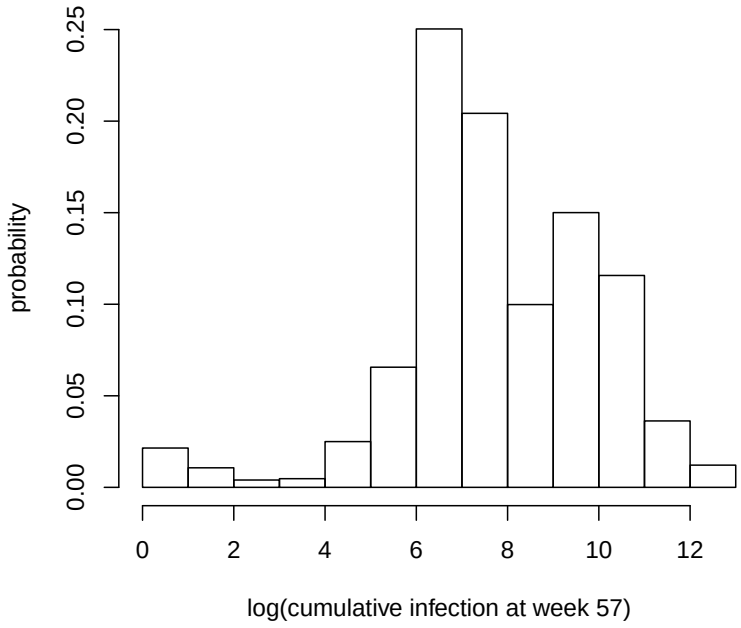


Figure 2.19: The figure shows the histogram of total infections at week 57 obtained from simulations at 100 unique input settings, each with 100 replicates.

where, $\boldsymbol{\psi}$ is the vector of ψ_ks denoting the parameters from individual components and $\boldsymbol{\pi}$ holds the proportions π_k. In gaussian mixture model each f_k is assumed to be normal and subsequently, ψ_k contains the mean and variance of the normal component. Usually, the mixture model parameters $\boldsymbol{\psi}$ and $\boldsymbol{\pi}$ are unknown and are estimated from the data. EM algorithm [38, 105] is useful in this case as maximization of the complicated log-likelihood function is often not achievable.

How one should treat the simulation output at 100 different input settings in clustering is not unique. While a simple approach can be using all 100×100 simulations together to separate them in two clusters based on marginal distribution of η, a more pragmatic solution would be to consider conditional clustering with condition being on the inputs. As noted before that the distribution of possible infection outbreaks varies as governing parameters of the disease change, it is more realistic to assume that the mixing proportions of the components as well as the parameters (e.g. mean and variance) characterizing the components. This argument in favor of conditional clustering

also aligns with the intended purpose of an emulator, i.e. having the ability of reproducing the conditional distribution $f(\eta|\boldsymbol{\theta})$ for any $\boldsymbol{\theta}$, where $f(\eta|\boldsymbol{\theta})$ potentially varies as we traverse along the input space. By indexing the simulation output by $\boldsymbol{\theta}$, we then write the conditional distribution of computer model output at $\boldsymbol{\theta}$ as,

$$f(\eta_{\boldsymbol{\theta}}|\mu_{1,\boldsymbol{\theta}}, \mu_{2,\boldsymbol{\theta}}, \sigma_{1,\boldsymbol{\theta}}, \sigma_{2,\boldsymbol{\theta}}) = \sum_{i=1}^{2} \pi_{k,\boldsymbol{\theta}} f_k(\eta_{\boldsymbol{\theta}}|\mu_{k,\boldsymbol{\theta}}, \sigma_{k,\boldsymbol{\theta}}).$$

The estimation of μ, σ and π are done using the EM implementation as package 'mclust' [135] in R software [122].

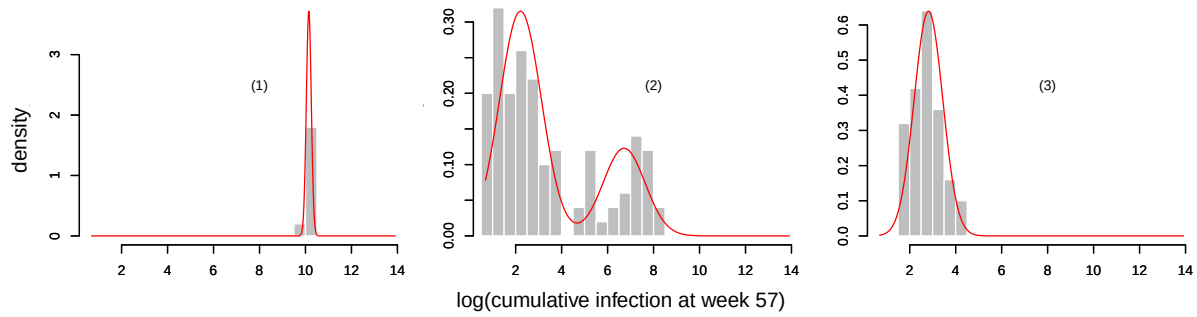


Figure 2.20: The figure shows the grey histograms of total infections at week 57 obtained from simulations at 3 input settings, each with 100 replicates (as used in Fig 2.18), and the estimated densities from gaussian mixture model is shown in red.

The estimated densities $\hat{f}(\eta_{\boldsymbol{\theta}}|\mu_{1,\boldsymbol{\theta}}, \mu_{2,\boldsymbol{\theta}}, \sigma_{1,\boldsymbol{\theta}}, \sigma_{2,\boldsymbol{\theta}})$ for each $\boldsymbol{\theta}$ in the simulation are used to classify the simulation replicates either in one of the two categories (high infection and low infection regime). We used a threshold of 0.5 for π_k, i.e., a replicate is classified as a member cluster k if $\pi_k > \pi_{k'}$. We refer the cluster assignment by the variable c, which takes two possible values 0 and 1.

Input Dependent Classification Model

As we want our final emulator to estimate the conditional densities $f(\eta|\boldsymbol{\theta})$ consistently with the actual simulation output, it ought to be able to estimate the mixture proportions π_k as a function

of simulation input $\boldsymbol{\theta}$. Although plethora of parametric and (semi)non-parametric classification models are available such as logistic regression, naive bayes, decision tree, random forest to name a few, we turn our head towards gaussian process based classification model [124] for a number of reasons. First of all, modeling the distribution $P(c|\boldsymbol{\theta})$ directly is harder and sometimes requires strong distributional assumptions. Secondly, in a computer model emulation setup one would likely try to exploit the non-linear spatial dependence on the response, for which gaussian process is a natural choice. For example, if most of the simulation runs at input $\boldsymbol{\theta}^*$ produce output in the higher regime, one would expect similar behavior at an input near $\boldsymbol{\theta}^*$. However, since gaussian distribution produces values between $(-\infty, \infty)$, it is often normal practice to model the underlying latent process and use a link function to connect the latent process to the actual observations.

We put a gaussian process prior with constant mean on the latent unobserved $z(\cdot)$ and use the logit function to transform z to the cluster probabilities.

$$
\begin{aligned}
z(\boldsymbol{\theta}) &\sim GP(\mu, C(\boldsymbol{\theta}, \boldsymbol{\theta}')) \\
C(\boldsymbol{\theta}, \boldsymbol{\theta}') &= \tau^2 \exp\left(- \frac{||\boldsymbol{\theta} - \boldsymbol{\theta}'||^2}{2l^2} \right) \\
\pi(\boldsymbol{\theta}) &= \frac{\exp(z(\boldsymbol{\theta}))}{1 + \exp(z(\boldsymbol{\theta}))}
\end{aligned}
$$

The estimation of the unknowns l, τ, μ is done using the R function *gausspr* implemented in kernlab package [87].

Gaussian Process Model with Input Dependent Noise

The other part of the full emulator requires to model the individual components of the mixture distributions of the simulation output, a straight forward choice for which is gaussian process model [76, 88, 131]. However, we note that a simple GP with constant error (nugget) is not appropriate in this case, where the conditional distribution of simulation output varies across the input space (see Fig 2.2). Non-constant variance is usually encountered in many stochastic computer models, and

is dealt with number of ways. For example, [51, 121] considered using quantiles to reconstruct the arbitrary distribution conditioning on the simulation inputs, whereas [4] extended the traditional GP model by adding an additional (constant) noise term, and [126] uses a mixture of normal with constant mean and variance as function of input to model the arbitrary distribution. But the idea in [62] to consider an input dependent (latent) error process in a GP model set up, and a fast algorithm to estimate the GP parameters in [18] make this an attractive candidate to be considered for modeling this ABM emulator.

A heterogeneous GP differs from the constant error counterpart by the extra input dependencies in the error variance. By writing the emulator model as $\eta(\boldsymbol{\theta}) = w(\boldsymbol{\theta}) + \epsilon(\boldsymbol{\theta})$, with $\epsilon(\boldsymbol{\theta}) \sim N(0, \sigma^2(\boldsymbol{\theta}))$ and w being modelled as standard GP, the likelihood become a multivariate normal with covariance $K + \Sigma$, where K is obtained by applying covariance kernel of choice on the initial design input settings, and $\Sigma = \text{Diag}(\sigma^2(\boldsymbol{\theta}_1), \cdots, \sigma^2(\boldsymbol{\theta}_m))$. Given an estimate of parameters in K and $\sigma^2(\boldsymbol{\theta})$, the predictive distribution at a new input setting $\boldsymbol{\theta}^*$ follows from the gaussian conditional distribution $f(\eta(\boldsymbol{\theta}^*)|\eta(\boldsymbol{\theta}_1), \cdots, \eta(\boldsymbol{\theta}_m))$. Modeling the variability as $\exp(h(x))$ for simple functions h (e.g., polynomials) as a simple extension is suggested by [20], however it does not serve its purpose in many complicated applications. [18] proposes a GP prior for latent observations whose predicted mean is assumed to be $\log(\sigma^2(\boldsymbol{\theta}))$ coupled with a standard GP prior on w, as a means to model the intrinsic input depending variance. An implementation of this formulation and fast maximum likelihood estimation is given in [17].

Once the pre-run simulation outputs are classified into two categories (high and low regimes) using a gaussian mixture model, and a GP classifier is in place to predict the mixing proportions at new input setting, the rest of the remaining task is to train two independent GPs for two types of simulation output. The little ticks at the bottom of the estimated density in Fig 2.21 display the actual simulation outputs and the different colors show the different regimes of the simulation runs. Two independent heterogeneous GPs are used to model the red simulations and the black simulations.

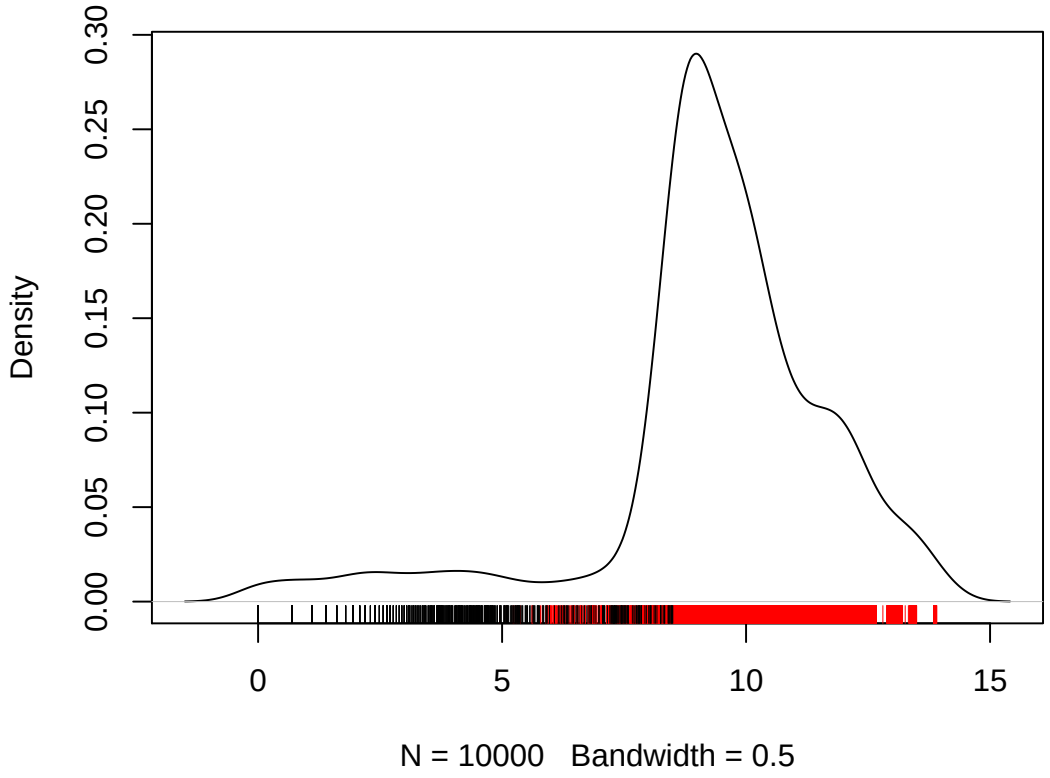


Figure 2.21: The figure shows the kernel density estimate of the all simulation output (same as the histogram in Fig 2.19), along with the actual outputs at the bottom in two colors. The colors characterizes the cluster memberships of the simulation output (high or low regime)

.

Combined Emulator

Combining the GP classifier along with the heterogeneous GPs, we finally write the complete emulator for the ABM as below:

$$\begin{aligned}
\eta(\boldsymbol{\theta}) =& I_{[\pi(\boldsymbol{\theta})>0.5]} w_1(\boldsymbol{\theta}) + I_{[\pi(\boldsymbol{\theta})<0.5]} w_2(\boldsymbol{\theta}) \\
\pi(\boldsymbol{\theta}) =& \frac{\exp(z(\boldsymbol{\theta}))}{1+\exp(z(\boldsymbol{\theta}))} \\
z(\boldsymbol{\theta}) \sim& GP(0, \Sigma(\boldsymbol{\theta}, \boldsymbol{\theta}')) \\
w_k(\boldsymbol{\theta}) \sim& GP(\mu_k, \Sigma_k(\boldsymbol{\theta}, \boldsymbol{\theta}')), \quad k = 1, 2,
\end{aligned} \tag{2.13}$$

where, I is the indicator function.

We use anisotropic gaussian kernels for all GP covarinaces. Predicting simulation output at an unknown $\boldsymbol{\theta}^*$ requires predictions from three GPs, $z(\boldsymbol{\theta}), w_1(\boldsymbol{\theta}), w_2(\boldsymbol{\theta})$ in Eq. 2.13. An illustration of in-sample predictive performance is depicted in Fig 2.21 for the same three input settings as shown in Fig 2.18, 2.20. Noticeably our multi-stage emulator is able to reproduce the simulation behaviors

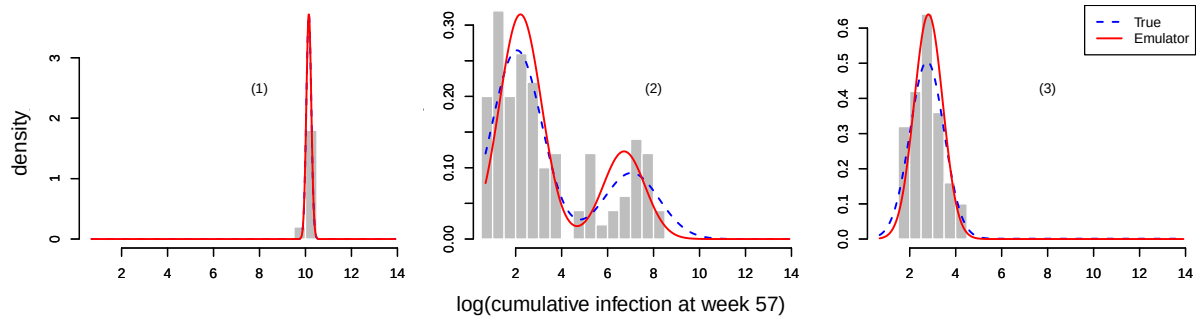


Figure 2.22: The figure shows true distribution (according to gaussian mixture model) of the simulation at three different input locations (same as Fig 2.18, 2.20) in blue and the predictive distribution using the combined emulator in Eq. 2.13 is shown in red. The gray histrograms in the background are correspond to the actual simulation output.

at three very different input settings for example. At input setting (1) in 2.21, the actual simulation runs resulted in very high infection counts and as expected the emulator correctly predicts the high proportion of being in the high value regime of the simulation, as well as the mean and marginal variance of the log of infections. Similarly, at input location (3), as most of the simulation runs produced very low infection, suggesting the unlikeliness of getting high infection count at that input setting, the emulator accurately determines the probabilities of producing low of high valued output, and the marginal mean and variance of the actual simulator output depending on the input. The middle frame shows a scenario where the ABM simulation at the input setting (2) may sometime results in higher number of infections and sometimes low or zero infection. The mixture model based emulator appropriately recovers the bi-modal predictive distribution of the simulator response at that input setting.

2.3.2 Multivariate Response

The classification based combined emulator can easily be extended to handle multivariate response from the simulator, however there are multiple instances where one would to make some choices or decisions based on actual end purpose. Our original simulator produces a time series of length 57 as output. To prepare the auxiliary dataset for training a GP classifier, one would require to classify the simulator response in one of the two categories, namely high regime and low regime. Unlike in univariate setup, how one classifies a vector output may itself be an independent research topic. Several functional clustering techniques are proposed lately, [82] surveys a few of them, notably, [125] adopted bayesian wavelet methods, while [59] and several others introduced a non-parametric dirichlet process mixture modeling for spatial data. However, we argue that a much simpler classification of the ABM simulator response would do our job well requiring less effort in this initial data preparation setup by leveraging some useful simulator properties. For example, by considering the cumulative counts of the infections as simulator response we guarantee that the output vector is monotonically increasing, and relatively smooth over time, eliminating the chance of putting a particular simulator response in two regimes simultaneously. To be specific, it is unlikely to observe a high rate of infection spread at the beginning ending up in a low total infection count region. By this same observation, we justify the use of only the last component of the vector simulator output (i.e. the log of total infection counts at the end of 57 weeks) for classifying the multivariate output. Essentially, this is exactly same as in the univariate set up. Nonetheless it is important to note that while the univariate solution works well in this case, other simulator may require more serious attention, or perhaps an unified approach in building an emulator.

Fig 2.23 shows an illustration of the simulator response when classified into two clusters based on the last component of the vector valued response. As expected, the classification conforms to our presumption that the total infection count serves as a good representative to the entire simulation trajectory for the purpose of categorizing the response into high or low regime.

We use the same modeling apparatus as used in the univariate case to model the multivariate

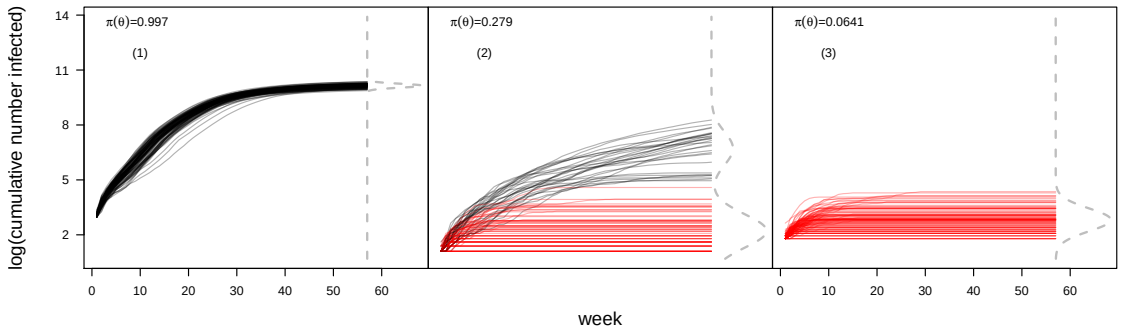


Figure 2.23: The figure shows the same set of simulations as in Fig 2.20 classified into two clusters (black and red) using the same gaussian mixture model, and the estimated cluster membership probability $\pi(\boldsymbol{\theta})$ is shown at top left corner in each frame. The un-normalized conditional distribution is also shown vertically in grey dashed line.

simulator output by re-indexing the vector valued response using an additional time variable. In mathematical terms, we express the 57-d vector $\eta(\boldsymbol{\theta})$ as $(\eta(\boldsymbol{\theta}, 1), \eta(\boldsymbol{\theta}, 2), \cdots, \eta(\boldsymbol{\theta}, 57))$. This formulation reduces the challenge and turns it into an usual univariate modeling problem. Subsequently, the input design matrix is also readjusted by appending an extra column consisting of time indices. The updated input dimension changes by 1, however, we make a note here distinguishing the extra input dimension from the other 5 actual input to the simulations, that the time indices is treated as a controlled input as done in many physical experiments [76]. Hence, this extra input is not treated as a calibration parameter, rather a mapping between the univariate response with it's multivariate counterpart.

The ABM simulator response $\eta(\boldsymbol{\theta})$ at $m = 100$ unique $\boldsymbol{\theta}$ with $r = 100$ replications at each $\boldsymbol{\theta}$ are cast into a long $100 \times 100 \times 57$ sized vector

$$\begin{aligned}\boldsymbol{\eta} =&(\eta_1(\boldsymbol{\theta}_1^*, 1), \cdots, \eta_1(\boldsymbol{\theta}_1^*, 57), \cdots, \eta_r(\boldsymbol{\theta}_1^*, 1), \cdots, \eta_r(\boldsymbol{\theta}_1^*, 57),\\ &\cdots, \cdots, \eta_1(\boldsymbol{\theta}_m^*, 1), \cdots, \eta_1(\boldsymbol{\theta}_m^*, 57), \cdots, \eta_r(\boldsymbol{\theta}_m^*, 1), \cdots, \eta_r(\boldsymbol{\theta}_m^*, 57))\end{aligned}$$

Similarly, 100×5 design matrix is reconstructed to create a $100 \times 100 \times 57$ times 6 dimensional

matrix compatible with the large response vector $\boldsymbol{\eta}$.

$$\mathcal{X} = \begin{pmatrix} \theta^*_{11} & \cdots & \theta^*_{1p} & 1 \\ \vdots & \ddots & \vdots & \vdots \\ \theta^*_{11} & \cdots & \theta^*_{1p} & 1 \\ \vdots & \ddots & \vdots & \vdots \\ \theta^*_{m1} & \cdots & \theta^*_{mp} & 57 \\ \vdots & \ddots & \vdots & \vdots \\ \theta^*_{m1} & \cdots & \theta^*_{mp} & 57 \end{pmatrix}$$

For two independent GP model, the big matrix $\mathcal{X}$ and the vector $\boldsymbol{\eta}$ are divided into two smaller matrices and vector respectively, according to the classification obtained by the GP classifier as seen in Fig 2.23, and two heterogeneous GP model [18] are trained using them. The final emulator is now expressed as:

$$\begin{aligned} \eta(\boldsymbol{\theta}, t) =& I_{[\pi(\boldsymbol{\theta})>0.5]} w_1(\boldsymbol{\theta}, t) + I_{[\pi(\boldsymbol{\theta})<0.5]} w_2(\boldsymbol{\theta}, t) \\ \pi(\boldsymbol{\theta}) =& \frac{\exp(z(\boldsymbol{\theta}))}{1+\exp(z(\boldsymbol{\theta}))} \\ z(\boldsymbol{\theta}) \sim& GP(0, \Sigma(\boldsymbol{\theta}, \boldsymbol{\theta}')) \\ w_k(\boldsymbol{\theta}, t) \sim& GP(\mu_k, \Sigma_k((\boldsymbol{\theta}, t), (\boldsymbol{\theta}', t'))), \quad k = 1, 2 \end{aligned} \tag{2.14}$$

For both Σ and Σ_k, an anisotropic gaussian covariance is used. Note that the multivariate emulator formulation only differs by an extra input in w_k from its univariate counterpart, $\pi(\boldsymbol{\theta})$ remains as is.

It is straightforward to derive the marginal distribution of $\eta(\boldsymbol{\theta}^*, t)$ at any $\boldsymbol{\theta}^*$, which is a mixture of two gaussians, as w_1 and w_2 are normally distributed marginally. This is indeed the property of the simulator that we wanted to mimic through this emulator. How one use this emulator for further study depends on the intended purpose. As we will see in the next subsection how we incorporate this emulator into an calibration setup, or how one can attempt determine the sensitivity of different

inputs on the simulator response.

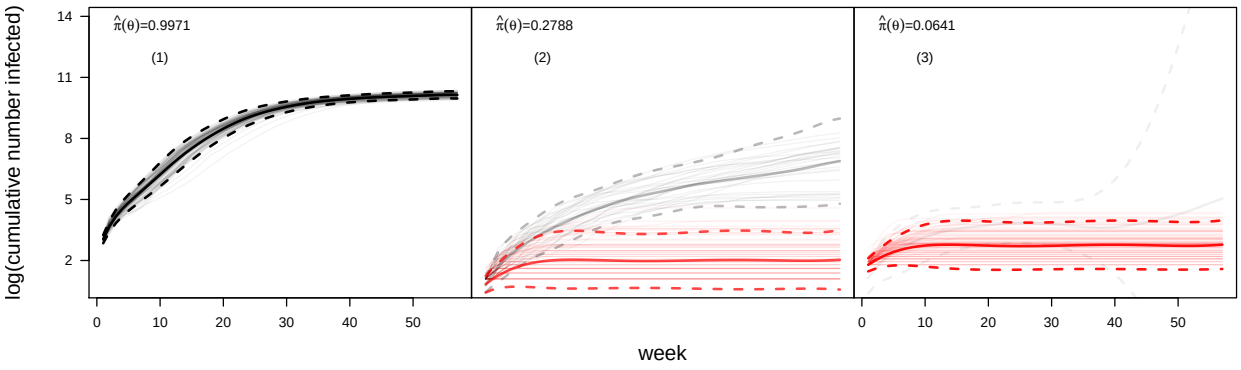


Figure 2.24: The solid lines, black and red respectively, show the in sample mean prediction from two GPs, and dotted lines represent corresponding 90% predicted CI. The fractions at top left of each frame denotes the estimated probability of simulator output being in the cluster denoted by black (higher regime). The opacity is also consistent with this estimated probability. Grey lines in the background are the actual simulation output.

The in-sample emulator performance if quite optimistic in Fig 2.24, as the predicted mean surface and 90% confidence interval is consistent with the actual simulator output distribution. For input setting (1), the GP classifier predicts that in more than 99 cases out of 100, the simulation should produce higher amount of infections as we see from the actual simulation runs. At input setting (2) the actual simulation results in much wider range of values. Our combined emulator puts $>$ 0.7 probability to be in the lower regime, and at input setting (3), almost with probability 1 the emulator predicts low valued response. However, the out of sample predictions in Fig 2.25 lacks accuracy, especially at input location where the simulation may produce significant proportion of low valued output. Such cases may be attributed to the fact that out of total 100×100 simulations, only 900 are classified as low regime output, limiting the data used to train a GP on a five dimensional input space. The GP model for this low regime simulations is expected to suffer in segregating the noise from the signal from only few data points.

One can also make an attempt to make sensitivity plots using the combined emulator. However, what should one report as emulator output for a multivariate stochastic computer model is not unique. In univariate case, one may choose to use the marginal emulative distribution, or some

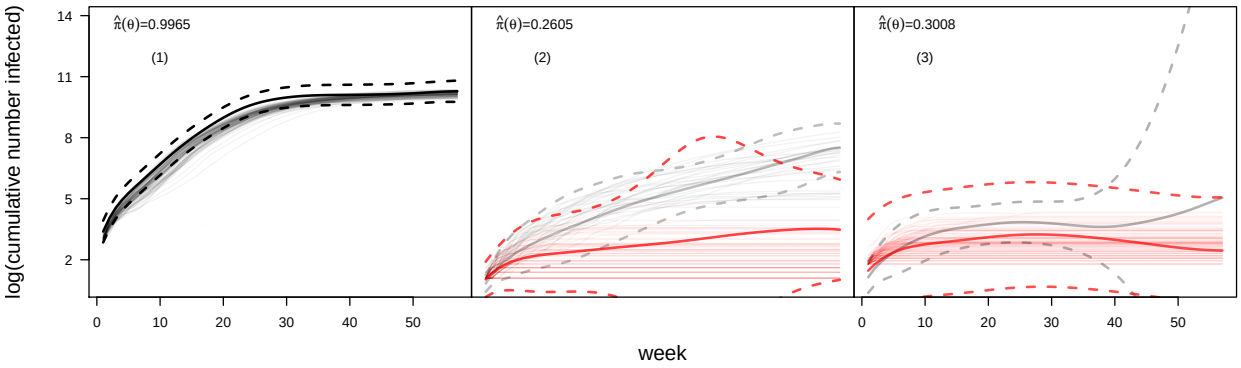


Figure 2.25: The same emulator prediction as in Fig 2.24, except they are result from leave one out exercise.

summary statistics from this distribution. But for multivariate case, although one may think of emulative distribution at each week index, it may not be easy to visualize from such figure how the whole simulator response reacts when an input is changed. Hence, keeping things consistent with Fig 2.11, we show the mean GP predictive lines for which $\pi(\boldsymbol{\theta}) > 0.5$.

The ABM simulator is most sensitive to the first input $\boldsymbol{\theta}_1$ (transmissibility) as seen in Fig 2.26, which agrees with the findings in Fig 2.11. In each frame of Fig 2.26, we vary one input (noted in the plot) between its predefined range while keeping other inputs fixed at their nominal values. The more darker lines correspond to the higher input values. All inputs show similar marginal positive relations, i.e., as they increase the simulation produces higher valued output.

2.3.3 Calibration

Unlike quantile based emulator in the previous section, here we opt for an optimization based approach to calibrate the ABM via using classification based emulator, and rely entirely on the fitted heterogeneous GP predictions for its predictive uncertainty. In other words, we simply want to minimize the distance (or a suitable function of it) between the simulator and the observations, and want to find the input setting(s) at which the minimum distance occurs. The forward predictions and the uncertainties are then obtained just by running the combined emulator at optimized input

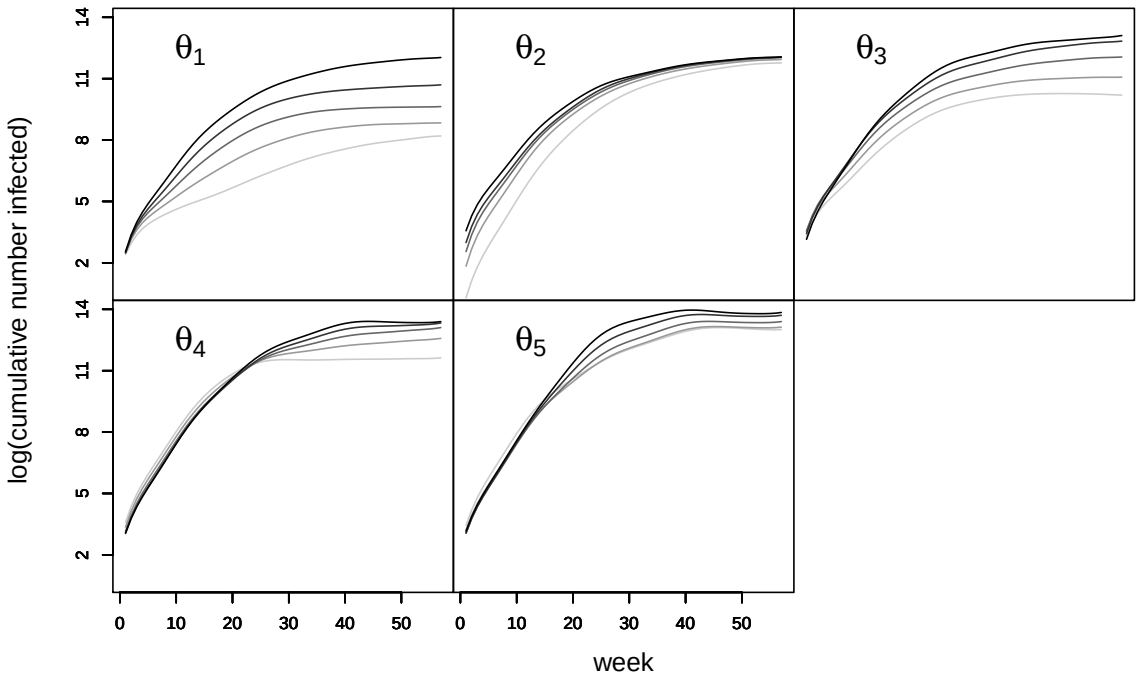


Figure 2.26: Predictive mean simulator predictions (log counts as a function of time) varying one input across its prior range, holding others at their nominal values

setting(s).

Mathematically speaking, the optimal input setting is $\hat{\boldsymbol{\theta}}$, which minimizes $y - \eta(\theta, t)$. However, as GP emulator produces probabilistic forecasts, one should also consider its predictive uncertainty into account while measuring the distance between the observation and the simulation prediction at a given $\boldsymbol{\theta}$. A simple choice in such situations would be mahalanobis distance, but one may wish to use a better metric with theoretical guarantees in order to obtain better calibration performance. A Scoring rule is a numerical value, defined as a measure for evaluating probabilistic forecasts, based on the predictive distribution and on the observation. This is also popularly referred to as utilities in Bayesian literature [14]. A scoring rule is *proper* if the predictive model maximizes the expected score given an observation from the same distribution as the predictive model gives. Motivating by proper scoring rules discussed in [60], we adapt the predictive model choice criterion (PMCC), originally proposed by [58] as our distance (score) metric to be minimized with respect to $\boldsymbol{\theta}$. The

main idea in [58] is for predictive distribution which only depends on its mean and variance, a factor containing the predictive variance should be added to the score. However, [60] argues that, to make the scoring rule *proper*, it should be based on a rule proposed in [36] based on generalized entropy function.

Here, we propose a similar scoring rule proposed by [36]. Note that our emulator $\eta(\boldsymbol{\theta}, t)$, unlike a simple GP, produces forecast from two GPs simultaneously, along with a probability of selecting one of the two predictions as the final one. For the simple case, where we model the simulator by a simple GP, we would define the proper score as:

$$-\sum_t \Big(\frac{y_t - \hat{\mu}(\boldsymbol{\theta}, t)}{\hat{\sigma}(\boldsymbol{\theta}, t)_t}\Big)^2 - \log \hat{\sigma}(\boldsymbol{\theta}, t)_t^2,$$

where, $\hat{\mu}(\boldsymbol{\theta}, t)$ and σ_t denotes the predictive mean and variance from the GP model. However, to accommodate our classification based emulator, we modify the scoring rule as:

$$\begin{aligned} S(\boldsymbol{\theta}) = &\sum_t \Big[-\Big(\frac{y_t - \hat{\mu}_{w_1}(\boldsymbol{\theta}, t)}{\hat{\sigma}(\boldsymbol{\theta}, t)_{t,w_1}}\Big)^2 - \log \hat{\sigma}^2(\boldsymbol{\theta}, t)_{t,w_1}\Big] \times \pi(\boldsymbol{\theta}) + \\ &\sum_t \Big[-\Big(\frac{y_t - \hat{\mu}_{w_2}(\boldsymbol{\theta}, t)}{\hat{\sigma}^2(\boldsymbol{\theta}, t)_{t,w_2}}\Big) - \log \hat{\sigma}(\boldsymbol{\theta}, t)_{t,w_2}^2\Big] \times (1 - \pi(\boldsymbol{\theta})) \end{aligned} \tag{2.15}$$

where, the subscript w_i corresponds to the two GPs that are used to build the combined emulator in 2.14. The adjustment made in eq 2.15 is inline with the emulator, such that, like predictions the individual scores from the GPs are also weighted by the corresponding cluster probabilities. The calibrated value of $\boldsymbol{\theta}$ is obtained by maximizing $S(\boldsymbol{\theta})$ with respect to $\boldsymbol{\theta}$.

Results

We optimized the distance between the observed data and the ABM simulator via our proposed cluster based GP surrogate and obtain forward prediction (extrapolation) along with its uncertainties. For the optimization, we use gradient based BFGS (Broyden–Fletcher–Goldfarb–Shanno) [54] algorithm, which belongs to the family of quasi-Newton methods. An implementation of this

algorithm is available in R [122] as an option to the base function *optim()*, which also estimates the gradients of the function to be optimized. In order to safe guard against any local maxima, we initialize the algorithm at random locations and run the optimization multiple times.

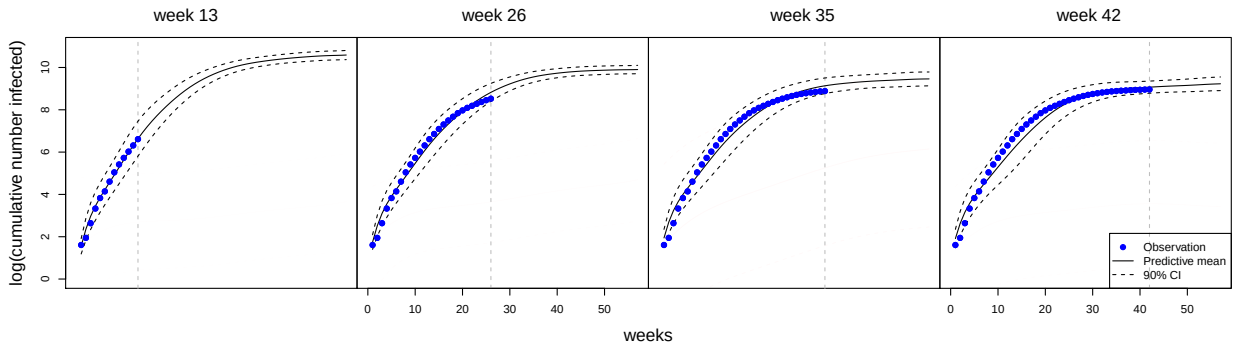


Figure 2.27: Each frame of the figure shows the observed data in blue, and the mean prediction in solid black dots, and 90% confidence interval in dashed black line.

The calibrated results are shown in Fig 2.27 and 2.28. Four independent prediction scenarios are shown, at week 13, 26, 35 and 42 respectively. For each of these scenarios, we use the aforementioned amount of observed data to calibrate and make predictions. Note that in all cases, the calibrated models correspond to the *higher* regime of the simulation output at those calibrated output. Fig 2.28 shows additional summaries such as total infection at the end of simulation, peak incremental infections and the corresponding week, obtained from full curves in Fig 2.27.

It is not very clear from Fig 2.27 how the prediction uncertainties change over time as we observe more data between different scenarios. On the contrary, at later stages of the epidemic, uncertainties seem to grow a bit larger with more data. The mean predictions seem to agree with the observed data in most weeks. However, the prediction accuracy and uncertainties on other epidemic characteristics seem to benefit from more data. Prediction of total infection and peak infection stabilizes after week 26. While the mean predictions are consistent with the data, the whole solution is quite different from the one we obtained by using quantile based emulator, specially the uncertainties.

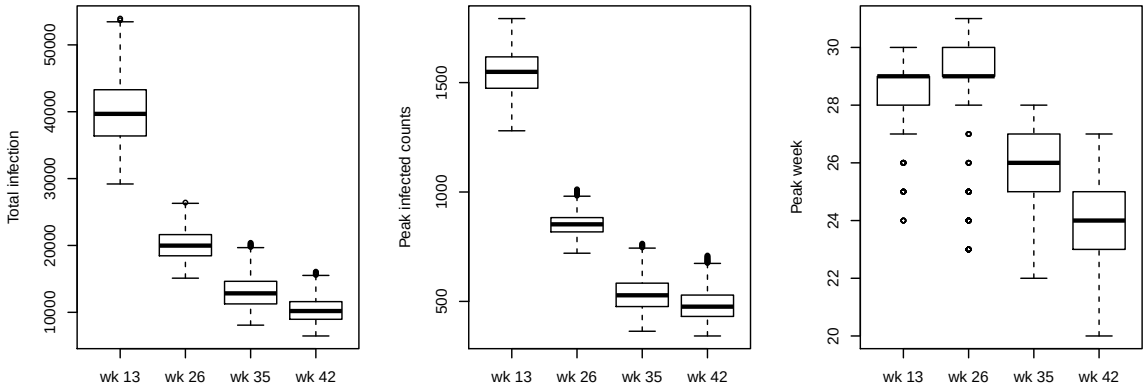


Figure 2.28: The left figure shows the boxplots of predicted total infection at each of the four scenarios from Fig 2.27. The middle frame corresponds to peak total infections, and the right frame shows peak week.

Discussion

While the classification based GP emulator does the job of finding the optimal input setting to match the simulator with observed data, it is worth mentioning that, in such optimization style calibration, one would sacrifice the full uncertainty quantification on the final prediction from the combined emulator. Although, by default GP surrogates provide one of the many uncertainties, it can only be attributed to uncertainty due to not observing the actual simulation output at that input setting.

However, there are few advantages over a fully bayesian solution as used in quantile based setup. The first one is to be able to avoid the likelihood associated with the observation y, i.e., assumning zero measurement error on the y. The no error model may not be inline with most basic statistical modeling principal, but construction of such likelihood here is not intuitive. In previous setup, a 20% error is assumed on the observed data, which is purely heuristic driven. Changing the heuristic might end up in a different solution, suggesting the subjective nature in likelihood choice resulting in a strong prior for accurate uncertainty quantification. Bypassing the need of an error model, we not only create a more objective model, it also saves computational efforts by avoiding big matrix inversions that would have arisen due to multivariate gaussian distribution. Having said that, there

is no doubt that the importance of having a likelihood whenever possible can not be looked upon, and should always be given a priority, when the ultimate goal is to perform a full uncertainty quantification, specially from a stochastic computer model.

The second advantage comes through the ability of running the emulator faster than the previous approach, specially when the number of prediction points are large. Use of woodbury trick in the implementation of heterogeneous gaussian process [17] in R, it takes significantly less amount of time to produce a realization from GP, which aids in performing optimization and sensitivity analysis to name a few in a short amount of time. However, for full bayesian solution in the quantile based approach, it took about 20 minutes to draw 10,000 mcmc samples from the joint posterior having more than 30 parameters, whereas, heterogeneous GP training on full $100 \times 100 \times 57$ data points took close to an hour. Although, the data volume in the quantile based approach is much less, since we work only with the quantiles derived from the full data.

Inspite of having few advantages over full bayesian solution, if getting an accurate predictive uncertainty is of interest, the cluster based solution as presented here lacks some of this. There are many sources of uncertainties that may be present as described by the authors in [88], one of which is input uncertainty, i.e., the uncertainty in the prediction due to not knowing the true values of the calibration parameters. In most cases, this uncertainty is propagated through the actual computer simulation, or statistical emulator. Using an emulator introduces additional parametric uncertainties, which is accountable, but may be tricky as in the case here. Prediction from cluster based GP emulator involves prediction from 3 GPs simultaneously. While we consider the uncertainties from heterogeneous gaussian processes for actual simulation output, we only use the mean prediction for the cluster probabilities from classification model. A more comprehensive analysis would involve uncertainties arising from that classification model as well. Even if one accounts classification uncertainty, how can that be incorporated in optimization setup is again not clear, and any significant gain in finding the optimal input setting is not guaranteed.

Last but not the least, we use a prior knowledge that the simulation output may be divided in

two classes (higher and lower regimes), with the hope that it covers the simulation characteristics for the entire input space. Although this heuristic is aligned with the subject matter experts and justifies in the context of epidemic literature, it limits the usability of the method in other applications where number of clusters can not be fixed beforehand. In such cases more complicated models such as dirichlet process based methods [59], which puts a prior on the number of clusters is desired. However, those models take away the simplicity of the emulator, and relies heavily on computationally expensive mcmc, limiting their applications when time and computation budget is limited. Even when the number of clusters are known and fixed, separating simulation output into clusters in the pre-processing step can become complicated depending on the application. For example, for multivariate simulation like the ABM, our clustering is based on only the last component of the vector output, assuming it characterizes the full simulation. Otherwise, function clustering would have to be introduced, requiring more computation and complicated calculation for the full uncertainty. It is also important to note that we have not talked about any kind of discrepancy in this setup.

While the cluster based combined GP emulator shows an ingenious way to tackle stochastic computer model calibration problem, undoubtedly there are multiple scopes to improve and make it more robust so that other types of stochastic computer simulations satisfy the applicability criteria. Possible improvement may include but not limited to full uncertainty from the classification model, non-parametric classification, a full bayes solution and inclusion of discrepancy in the full model. We try to take care of some of them in next few sections.

Chapter 3

Sequential Calibration of Computer Model

3.1 Related Work

Bayesian inference of unknown parameters in a model relies on properties of the parameter's posterior density conditioned on some given observation. In some cases, the posterior can be analytically studied (e.g. gaussian state space model), and in many other cases posterior is available only up to a constant and not does not belong to a known family of distributions. One resorts to simulation based methods such as Markov Chain Monte Carlo (MCMC), Sequential Monte Carlo (SMC) in order to compute the posterior distribution. Monte carlo sampling techniques are general procedures to draw random realizations from an arbitrary distribution. SMC methods consist of several simple, convenient, flexible, parallelisable and easy to implement methods, which are applicable in very general settings [41]. Many of such closely related algorithms are particle filters, monte carlo filters, bootstrap filters, interacting particle approximation etc.. Problems where the observations arrive sequentially in time, SMC based inference is more appropriate in updating the posterior sequentially in time. The authors in [134] describes the use of SMC in solving non-linear system identification problems, and provides a comparative discussion on different SMC techniques.

While there are tons of SMC examples in real applications (a nice collection can be found in [40], there has not been much use of sequential monte carlo in computer model calibration problems, often due to inability of running an expensive computer simulation many many times. In addition

to that, sequential approaches often encounter degeneracy issues [97] in later stages of the inference. Importance sampling is an easily implementable and interpretable sampling technique, but suffers from impoverishment problems. Iterative importance sampling [108] and bayesian network particle filter [25] improve the performance of sampling scheme and are described in a general probabilistic setting by the authors. Use of SMC in the context of computer model appears in [101], where authors attempt to calibrate option valuation models, which depends on time varying input parameters, and several different methods are employed – ensemble Kalman filters [77], iterated extended Kalman filter [98], weighted least squares, penalized weighted least squares, and shows the advantages of SMC over other methods. Another application of SMC in computer model calibration can be found in [84], showing a comparative study between MCMC and SMC in calibrating hydrological models.

As an alternative to calibration via model surrogates, we propose a simple SMC based filtering method, and extend that to handle stochastic computer models. We describe our method with an example from disease propagation in New York city modeled by a meta population based stochastic compartmental disease model. We will also see that there is no extra effort needed for computer simulations with multivariate output in this SMC based set up.

3.2 Patch Model Revisited

We revisit the patch model [148] described in chapter 1, in the context of modeling seasonal influenza activity New York city during a typical flu season in US. Seasonal influenza in US has received significant attention in recent years due to its significant heath impacts and economic burden [1]. Forecasting influenza activity in the United States has been an active research area among the epidemiologists, an extensive review of some is found in [27, 111]. While most statistical models rely on discovering patters in the time series data of past flu seasons, other methods such as mechanistic models use disease dynamics itself, described through mathematical equations to

[1] https://www.cdc.gov/flu/about/burden/2017-2018.htm

represent the disease propagation. Advantages of mechanistic models have already been discussed in chapter 1. Variants of metapopulation models can be found in action for forecasting infectious diseases [9, 119, 143]. While spread of infectious disease is influences by social contacts, researchers often leverage this fact and use information on human mobility to predict the disease characteristics. Data on movement of individuals at coarse level are readily available in public domain [2], encouraging their use in modeling infectious diseases in urban areas [8, 155]. Here we adopt the patch model to model the disease characteristics in New York city, by defining each of five boroughs as patches (meta populations), and by incorporating heterogeneity through the commuter travel information from US census.

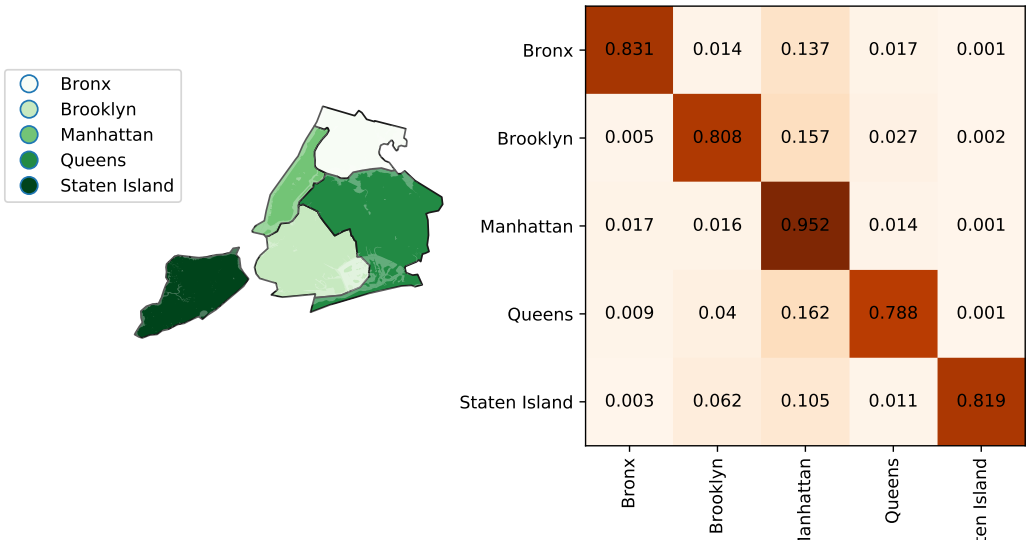


Figure 3.1: The left frame shows 5 boroughs that New York city consists of, and the right matrix plot shows proportion of population traveling from a borough on the row to a borough on the column.

The SEIR based patch model takes a list of inputs, they are:

- initial seeding: number of infected individuals at the start of simulation,
- transmissibility: rate of transition from S to E,
- alpha: rate of transition from E to I,

[2] https://www.census.gov/topics/employment/commuting.html

• gamma: rate of transition from I to R,

• scaling factor: reporting rate of actual disease incidence.

Other information that is needed to run the patch model includes population counts for each patches, a travel matrix representing the human movement between patches (rowsums must be 1) and vaccination information (optional, if available). Fig 3.1 shows a map of New York city which consists of 5 boroughs - Bronx, Brooklyn, Manhattan, Queens and Staten Island, and the matrix on the right shows average normalized population fraction who travel from one borough to another borough.Temporal travel matrix is also allowed to incorporate seasonal pattern in the travel, but for the sake of simplicity we avoid a temporal travel matrix here. Fig 3.2 gives an example of a simulation output from the patch model on 5 NYC boroughs for 30 weeks. Note that here the patch model is run in stochastic mode, i.e., output will vary from one model run to another at same input setting.

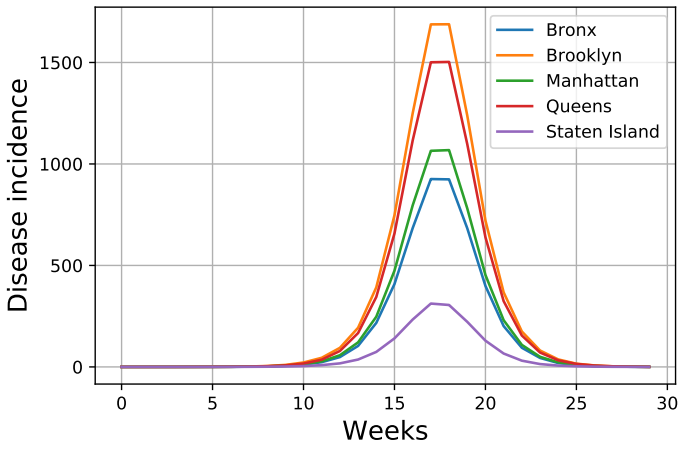


Figure 3.2: An output from flu simulation on New York city patch model.

In next few sections, we would address the calibration problem for different scenarios, e.g. calibrating with respect to total episize of NYC, and then a spatial calibration for each borough separately [119].

3.3 Sequential Calibration (Temporal)

3.3.1 Basic SMC on State-Space Models

In this section, a calibration approach similar to online inference [86] is proposed, where data comes in in regular intervals and the posterior distribution of calibration parameters is updated sequentially based on new data. In a typical sequential monte carlo (SMC) based inference problem, one would work with a stochastic process $\{Y_t\}$ which depends on a (latent) hidden markov process $\{X_t\}$, and the goal would be to infer about the unobserved stochastic process through the observed process, which is assumed to be conditionally independent given the latent process. An online inference is implemented where inference about $\{X_t\}$ (or its characterizing parameters) is updated for each new $\{Y_t\}$, for $n > 0$. In summary, the model is written as follows:

$$X_0 \sim \pi(\cdot|\boldsymbol{\theta}), \quad X_t|X_{t-1} \sim f(\cdot|X_{t-1};\boldsymbol{\theta}),$$
$$Y_t|X_0,\cdots,X_t \sim g(\cdot|X_0,\cdots,X_t;\boldsymbol{\theta}),$$

where π is the initial density, f is the markov transition density, characterized by $\boldsymbol{\theta}$ and g is the conditional marginal density. This class of model covers many non-linear and non-gaussian models, and is known as general state space model. One may be interested in finding the posterior density of $p(X_0,\cdots,X_t|Y_0,\cdots,Y_t)$ when $\boldsymbol{\theta}$ is known and $p(\boldsymbol{\theta}; X_0,\cdots,X_t|Y_0,\cdots,Y_t)$ when unknown. From Bayes theorem when $\boldsymbol{\theta}$ is known, we can write the following:

$$p_{\boldsymbol{\theta}}(X_0,\cdots,X_t|Y_0,\cdots,Y_t) = \frac{p_{\boldsymbol{\theta}}(X_0,\cdots,X_t,Y_0,\cdots,Y_t)}{p_{\boldsymbol{\theta}}(Y_0,\cdots,Y_t)},$$

where

$$p_{\boldsymbol{\theta}}(X_0,\cdots,X_n,Y_0,\cdots,Y_t) = \pi(X_0|\boldsymbol{\theta}) \prod_{i=1}^{t} f(X_k|X_{k-1};\boldsymbol{\theta}) \prod_{i=0}^{t} g(Y_k|X_k;\boldsymbol{\theta})$$

and

$$p_{\boldsymbol{\theta}}(Y_0, \cdots, Y_t) = \int p_{\boldsymbol{\theta}}(X_0, \cdots, X_t, Y_0, \cdots, Y_t)\, dX$$

After some basic calculation, it is easy to see that,

$$\begin{aligned} p_{\boldsymbol{\theta}}(X_0, \cdots, X_t | Y_0, \cdots, Y_t) &= p_{\boldsymbol{\theta}}(X_0, \cdots, X_{t-1} | Y_0, \cdots, Y_{t-1}) \\ &= \frac{f(X_t | X_{t-1}; \boldsymbol{\theta}) g(Y_t | X_t; \boldsymbol{\theta})}{p(Y_t | Y_0, \cdots, Y_{t-1}; \boldsymbol{\theta})} \end{aligned} \tag{3.1}$$

Note that, given t-th observation the posterior is proportional to the posterior conditional up to $(t-1)$-th observation, the one step transition probability f from the hidden markov process and the conditionally independent distribution g, suggesting *sequential* in its name and how it relates to online inference. SMC algorithms numerically approximate the posterior densities and provides an easy avenue for producing samples from the posterior.

Importance sampling [110, 141] is a numerical approximation technique used to estimate properties of an unknown distribution by drawing samples from a known *importance* distribution and re-weighting them according to the target distribution. For example, expectation of a distribution f can be approximated by:

$$E(f(X)) = \int x\, \frac{f(x)}{g(x)}\, g(x)\, dx \approx \frac{1}{N} \sum_{i=1}^{N} x_i\, \frac{f(x_i)}{g(x_i)},$$

where, x_i denotes random sample from importance distribution $g(\cdot)$, and the quantity $w_i = \frac{f(x)}{g(x)}$ is known as importance weight. Similarly, a random sample from $f(\cdot)$ can be obtained by re-sampling $x_i \sim g(\cdot)$ according to the normalized importance weights w_i. An estimate of the distribution f is then follows from:

$$\hat{f}(x) = \sum_{i=1}^{N} w_i \delta_{x_i},$$

where, δ is the dirac-delta function. This is often used to draw samples from unknown posterior using prior as an importance distribution [32].

In an sequential setting, one would require to draw random samples successively from updated posterior at every time instance, when a new data point is augmented. Sequential importance sampling or SIS works on the same principal; using the conditional Independence assumption and markovian property one could deduce that the updated importance weights are proportional to older ones, and differ only by the factors of one step transitional probabilities and data likelihood (eq 3.1).

The same principal can be applied in a (deterministic) computer model calibration set up as well, where the goal is to find the posterior $p(\boldsymbol{\theta}|\eta, y)$. Suppose the likelihood $p(y|\eta, \boldsymbol{\theta})$ and the prior $\pi(\boldsymbol{\theta})$ are well defined, and the computer model η can be run infinitely many times (pretend that we have unlimited time and memory budget), then a bunch of posterior samples can be generated by (1) first generating $\boldsymbol{\theta}$ according to a suitable design, (2) running the computer model at $\boldsymbol{\theta}$ and calculating $p(y|\eta, \boldsymbol{\theta})$, (3) and lastly, computing the importance weights for each $\boldsymbol{\theta}$. Sequential calibration comes in to picture when the inference is online, i.e., data comes in regular interval, or the likelihood can be factorized in to two or more components targeting different data metrics. However, the same idea does not work in the case of stochastic computer model due to additional variability present in the likelihood from simulation replicates. In the next sub section we propose a updated sequential calibration technique suitable for handling stochastic simulations.

3.3.2 SMC for stochastic computer model

One-step Calibration

Among many uncertainties, we would focus on the uncertainties due to unknown model parameters and uncertainties from the replication variability which varies across input parameter space. For example, note in Fig 2.18, which shows 100 simulation outputs each at three different input parameters, that the variations in the replicates are not same in three frames. In a stochastic computer model inverse problem, one is not only interested in finding the marginal posterior $p(\boldsymbol{\theta}|\eta, y)$, but

also in keeping track of (good) replicates for each $\boldsymbol{\theta}$ in $p(\boldsymbol{\theta}|\eta, y)$ which are consistent with data. One may choose to parameterize the replicates via quantiles or population clusters and calibrate an additional parameter (as we have seen in chapter 2), thus requiring a pre-processing step of the simulation suit, which is often not feasible. As an alternative, we propose a new parameterization where each of the replicate is indexed by a random number (consider it as random number generator seed). While the exact formulation is given later, it is worth noting here that this is largely motivated by two main facts, (1) the replicates are not to be calibrated, but rather needs only to be kept track of and (2) our goal is to find the marginal posterior $p(\boldsymbol{\theta}|\eta, y)$, which does not include replicates.

We begin by writing the simple data model as follows:

$$\boldsymbol{y} = \eta(\boldsymbol{\theta}, r_{\boldsymbol{\theta}}) + \boldsymbol{e},$$

where, $\boldsymbol{y}$ is the field observation (can be a vector, or multidimensional), $\eta(\boldsymbol{\theta}, r_{\boldsymbol{\theta}})$ denotes r-th simulation output at input $\boldsymbol{\theta}$, and $\boldsymbol{e}$ consists of all unaccounted errors. Given the prior $\pi(\boldsymbol{\theta})$ and likelihood for y, the marginal posterior of $\boldsymbol{\theta}$ is written as:

$$\begin{aligned} p(\boldsymbol{\theta}|\boldsymbol{y}) &= \frac{L(\boldsymbol{y}|\boldsymbol{\theta}, \eta)\pi(\boldsymbol{\theta})}{p(\boldsymbol{y})} \\ &\propto L(\boldsymbol{y}|\boldsymbol{\theta}, \eta)\pi(\boldsymbol{\theta}) \end{aligned} \tag{3.2}$$

However, the marginal likelihood $L(\boldsymbol{y}|\boldsymbol{\theta}, \eta)$ is not directly available to us, since simulation output depends on both $\boldsymbol{\theta}$ and replicate $r_{\boldsymbol{\theta}}$. Instead, for each $\boldsymbol{\theta}$ and a replicate one can compute the full likelihood $L(\boldsymbol{y}|\boldsymbol{\theta}, r_{\boldsymbol{\theta}}, \eta)$, using which one may try to approximate the integral $\int L(\boldsymbol{y}|\boldsymbol{\theta}, r_{\boldsymbol{\theta}}, \eta)\, dF(r_{\boldsymbol{\theta}})$, provided the distribution $F(r_{\boldsymbol{\theta}})$ is known. Since the replicate indices are nothing but a one-to-one

mapped randomly generated numbers, a simple monte-carlo estimate is given by:

$$L(\boldsymbol{y}|\boldsymbol{\theta},\eta) = \int L(\boldsymbol{y}|\boldsymbol{\theta}, r_{\boldsymbol{\theta}}, \eta)\, dF(r_{\boldsymbol{\theta}}) \approx \frac{1}{n_r}\sum_{r_{\boldsymbol{\theta}}} L(\boldsymbol{y}|\boldsymbol{\theta}, r_{\boldsymbol{\theta}}, \eta) \tag{3.3}$$

Algorithm 1: Important Sampling on $p(\boldsymbol{\theta}|\boldsymbol{y})$

output: Posterior samples $\boldsymbol{\theta}_1, \cdots, \boldsymbol{\theta}_n$

for $i = 1$ **to** N **do**
- Draw $\boldsymbol{\theta}_i \sim \pi(\boldsymbol{\theta})$;
- **for** $j = 1$ **to** R **do**
  - Run simulation and produce $\eta(\boldsymbol{\theta}_i, j)$;
  - Compute importance weight $w_{ij} = L(\boldsymbol{y}|\boldsymbol{\theta}_i, j, \eta)$;
- Compute $w_i = \frac{1}{R}\sum_{j=1}^{R} w_{ij} = \left\{\frac{1}{R}\sum_{j=1}^{R} L(\boldsymbol{y}|\boldsymbol{\theta}_i, j, \eta)\right\}$;
- Resample n times with replacement from $\boldsymbol{\theta}_1, \cdots, \boldsymbol{\theta}_N$ with probabilities proportional to $w_1, \cdots, w_N$;

Algorithm 1 describes how to obtain posterior samples for the simulator input parameter $\boldsymbol{\theta}$ using the likelihood $L(\boldsymbol{y}|\boldsymbol{\theta}_i, r_{\boldsymbol{\theta}}, \eta)$. A demonstration of Algorithm 1 is given in next few paragraphs.

For the purpose of illustration, we consider $\boldsymbol{\theta}$ to be a scalar, in particular it represents the transmissibility parameter in the patch model, other model parameters are held fixed. Next, we generate $\boldsymbol{y}$ by running the patch model at $\boldsymbol{\theta} = 0.5$ for 30 weeks, where $\boldsymbol{y}$ represents the number of infected individuals. Note that the simulated $\boldsymbol{y}$ is only a random realization of the simulation at $\boldsymbol{\theta} = 0.5$. Following the data transformation done in chapter 2, we consider the log of cumulative infections as our final output from the simulation. The error distribution is assumed to be gaussian with a standard deviation of 25% of the observed incremental disease incidences. Fig 3.3 shows the simulated $\boldsymbol{y}$ along with error bars.

While the full 30-week $\boldsymbol{y}$ will be used for full calibration, we used only up to week 10 to demonstrate the importance sampling in Algorithm 1. With $\boldsymbol{y} = (y_1, y_2, \cdots, y_{10})$, where, y_t is log of cumulative

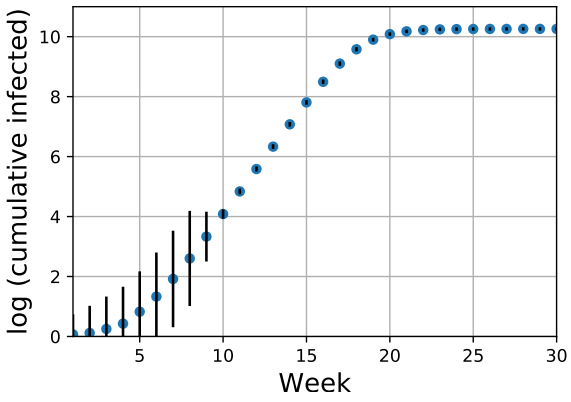


Figure 3.3: Simulated output from flu simulation on New York city patch model, with 1-sd error bars.

infected, 10×10 covariance matrix Σ_y, and marginal precision λ_y ,the likelihood is written as:

$$L(\boldsymbol{y}|\boldsymbol{\theta}_i, r_{\boldsymbol{\theta}}, \eta) \propto \lambda_y^{-5}|\Sigma_y|^{-1}(\boldsymbol{y} - \eta(\boldsymbol{\theta}, r_{\boldsymbol{\theta}}))'|\Sigma_y|^{-1/2}(\boldsymbol{y} - \eta(\boldsymbol{\theta}, r_{\boldsymbol{\theta}})).$$

The prior $\pi(\boldsymbol{\theta})$ is set to be uniform(0.4, 0.6). By setting $N = 100$ and $R = 10$, a total of 1000 simulation runs at 100 unique settings are done. In the re-sample step we draw 500 samples with replacement. The left frame in Fig 3.4 shows the histogram of posterior samples along with density estimate. The right frame shows calibrated model output in dark grey which agrees with the simulated $\boldsymbol{y}$ (in blue dots). Number of unique $\boldsymbol{\theta}$ in posterior samples selected in re-sampling step is 56. However, not all simulation replicates at a posterior $\boldsymbol{\theta}^*$ are consistent with the data. Hence, one needs to weight the corresponding replicates and keep plausible ones given the data according to their individual likelihoods $L(\boldsymbol{y}|\boldsymbol{\theta}_i, r_{\boldsymbol{\theta}}, \eta)$. This can be achieved by a similar strategy as performing an importance sampling locally for each posterior sample $\boldsymbol{\theta}^*$. Given that the individual likelihoods are already been pre-calculated, the rest of the task reduces to draw replicates with replacement for each $\boldsymbol{\theta}^*$ from $p(\boldsymbol{\theta}|\boldsymbol{y})$.

Importance sampling is known to suffer from sample degeneracy [32, 86, 97]. Starting with more prior samples or choosing an importance distribution close to posterior may work towards solving

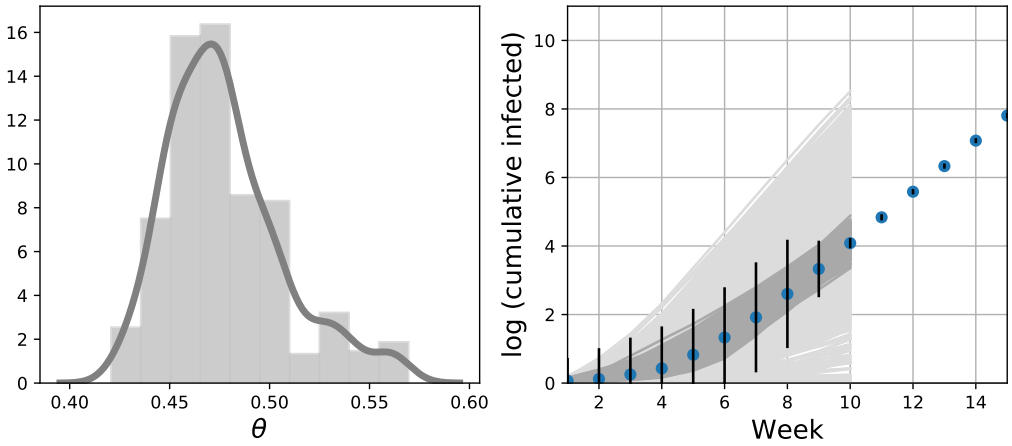


Figure 3.4: Histogram of posterior samples of calibration parameter $\boldsymbol{\theta}$ (left), prior and posterior simulation output (in light and dark grey respectively) along with the simulated data.

the degeneracy issue partially, but they come with some costs. Using more prior samples at the beginning of importance sampling may not guarantee large number of unique $\boldsymbol{\theta}$ in posterior sample if the prior is uninformative such that many of simulation runs are implausible and are wasted. Choosing an appropriate importance distribution is still an art and often not feasible for a black box simulation model. Instead we adopt a procedure based on MCMC, known as particle MCMC or pMCMC [3], to draw more sample using a markov chain whose stationary distribution is the posterior of interest. The central idea is exactly similar to a standard step where at each iteration a candidate is drawn from a proposal and accepted according to an acceptance probability which depends on the target distribution. An MCMC simulation starts from any arbitrary starting point and needs to run longer in order to guarantee convergence. This requirement is nullified by using already obtained posterior samples from the importance sampling step as a starting point in MCMC.

Algorithm 2 describes the above mentioned MCMC step to enhance the collection of samples from the marginal posterior $p(\boldsymbol{\theta}|\boldsymbol{y})$. The same conditional sampling would work to select plausible replicates at each posterior $\boldsymbol{\theta}$ which are consistent with the data, in order to make the posterior prediction. Fig 3.5 shows a comparison between two histograms of the posterior samples from

Algorithm 2: MCMC step

input : Posterior samples $\boldsymbol{\theta}_1, \cdots, \boldsymbol{\theta}_n$ from Algorithm 1
output: Posterior samples $\boldsymbol{\theta}'_1, \cdots, \boldsymbol{\theta}'_n$

for $i = 1$ **to** n **do**
  Draw $\boldsymbol{\theta}_i^* \sim g(\boldsymbol{\theta}_i^* | \boldsymbol{\theta}_i)$;
  for $j = 1$ **to** R **do**
    Run simulation and produce $\eta(\boldsymbol{\theta}_i^*, j)$;
    Compute likelihood $L_{ij} = L(\boldsymbol{y} | \boldsymbol{\theta}_i^*, j, \eta)$;
  Compute marginal likelihood $L(\boldsymbol{y} | \boldsymbol{\theta}_i^*, \eta) = \frac{1}{R} \sum_{j=1}^{R} L_{ij}$;
  Compute acceptance probability $p = \min \left\{ 1, \frac{L(\boldsymbol{y} | \boldsymbol{\theta}_i^*, \eta) \pi(\boldsymbol{\theta}_i^*) q(\boldsymbol{\theta}_i | \boldsymbol{\theta}_i^*)}{L(\boldsymbol{y} | \boldsymbol{\theta}_i, \eta) \pi(\boldsymbol{\theta}_i) q(\boldsymbol{\theta}_i^* | \boldsymbol{\theta}_i)} \right\}$;
  Draw $u \sim U(0, 1)$;
  if $u < p$ **then**
    Set $\boldsymbol{\theta}'_i = \boldsymbol{\theta}_i^*$
  else set $\boldsymbol{\theta}'_i = \boldsymbol{\theta}_i$;

algorithm 1 and algorithm 2, and Fig 3.6 shows the corresponding posterior simulation predictions.

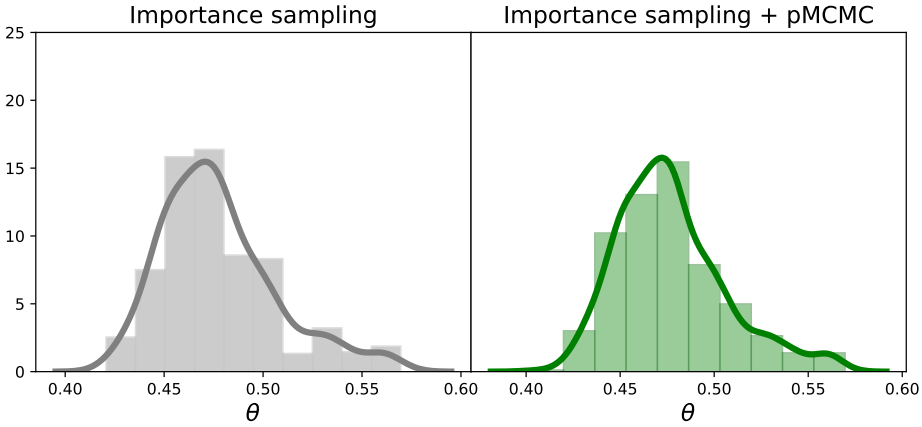


Figure 3.5: Posterior distribution of $\boldsymbol{\theta}$ from algorithm 1 (left) and algorithm 1 + 2 (right).

Sequential Calibration

The sequential aspect comes into picture due the nature of the application we are dealing with. Unlike in a state-space model where the posterior inference on n-th hidden states are updated sequentially, here the posterior on $\boldsymbol{\theta}$ is updated as more data comes in each week. In the simulated

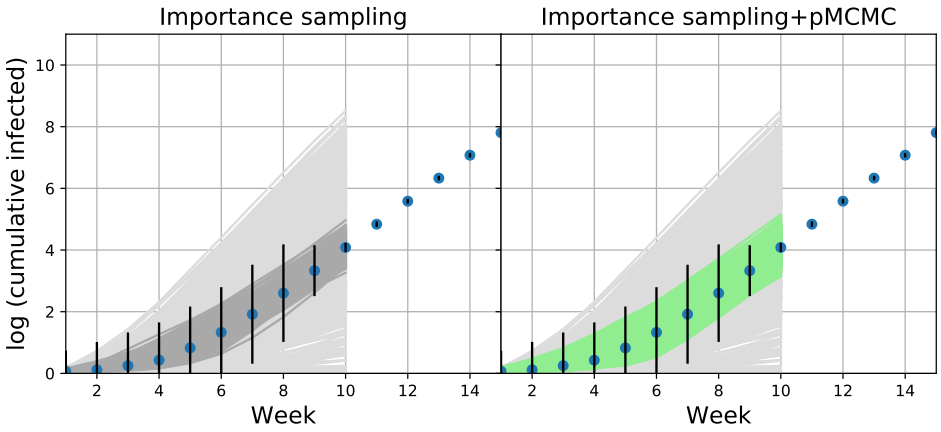


Figure 3.6: Posterior samples from calibrated simulator from algorithm 1 (left) and algorithm 1 + 2 (right).

example, the initial calibration is done using first 10 weeks of infection counts. By considering 10 more weeks of data to $\boldsymbol{y}$ becomes a 20-dimensional vector $(y_1, \cdots, y_{20})$, and the covariance matrix Σ_y is updated accordingly. One can produce samples from the new posterior using algorithm 1 and 2, but a more efficient sampling scheme would use already sampled $\boldsymbol{\theta}$ from the intermediate posterior conditioned on 10 weeks of data.

Define $\boldsymbol{y}^{(1)} = (y_1, \cdots, _{10})$, and $\boldsymbol{y}^{(2)} = (y_{10}, \cdots, y_{20})$. Then the posterior is written as

$$\begin{aligned} p(\boldsymbol{\theta}|\boldsymbol{y}^{(1)}, \boldsymbol{y}^{(2)}) &\propto L(\boldsymbol{y}^{(1)}, \boldsymbol{y}^{(2)}|\boldsymbol{\theta}, \eta)\pi(\boldsymbol{\theta}) \\ &= L(\boldsymbol{y}^{(2)}|\boldsymbol{\theta}, \eta, \boldsymbol{y}^{(1)})L(\boldsymbol{y}^{(1)}|\boldsymbol{\theta}, \eta)\pi(\boldsymbol{\theta}) \\ &= L(\boldsymbol{y}^{(2)}|\boldsymbol{\theta}, \eta, \boldsymbol{y}^{(1)})p(\boldsymbol{\theta}|\boldsymbol{y}^{(1)}) \end{aligned}$$

Hence, the updated posterior is proportional to the intermediate posterior conditioned on $\boldsymbol{y}^{(1)}$. This formulation along with the existing sample from intermediate posterior $p(\boldsymbol{\theta}|\boldsymbol{y}^{(1)})$ provides an straight forward extension of algorithm 1 to generate samples from $p(\boldsymbol{\theta}|\boldsymbol{y}^{(1)}, \boldsymbol{y}^{(2)})$.

Subsequently an MCMC step can be deployed to enhance the pool of posterior samples from

Algorithm 3: Sequential Important Sampling on $p(\boldsymbol{\theta}|\boldsymbol{y}^{(1)}, \boldsymbol{y}^{(2)})$

input : $\boldsymbol{\theta}_1^{(1)}, \cdots, \boldsymbol{\theta}_N^{(1)} \sim p(\boldsymbol{\theta}|\boldsymbol{y}^{(1)})$ (from algorithm 1 and 2).
output: Samples $\boldsymbol{\theta}_1^{(2)}, \cdots, \boldsymbol{\theta}_n^{(2)}$ from $p(\boldsymbol{\theta}|\boldsymbol{y}^{(1)}, \boldsymbol{y}^{(2)})$.

for $i = 1$ **to** N **do**
  for $j = 1$ **to** R **do**
    Run simulation and produce $\eta(\boldsymbol{\theta}_i^{(2)}, j)$;
    Compute importance weight $w_{ij} = L(\boldsymbol{y}^{(2)}|\boldsymbol{\theta}_i^{(2)}, j, \eta, \boldsymbol{y}^{(1)})$;
  Compute $w_i = \frac{1}{R}\sum_{j=1}^{R} w_{ij} = \left\{\frac{1}{R}\sum_{j=1}^{R} L(\boldsymbol{y}|\boldsymbol{\theta}_i^{(2)}, j, \eta, \boldsymbol{y}^{(1)})\right\}$;
  Resample n times with replacement from $\boldsymbol{\theta}_1^{(1)}, \cdots, \boldsymbol{\theta}_N^{(1)}$ with probabilities proportional to $w_1, \cdots, w_N$;

$p(\boldsymbol{\theta}|\boldsymbol{y}^{(1)}, \boldsymbol{y}^{(2)})$. The right frame in Fig 3.7 summarizes the calibration output from the sequential procedure. After selecting plausible input setting $\boldsymbol{\theta}$ constrained by observed data until week 10 – simulation responses at which are shown by light green curves, we extend those curves (i.e. run the patch model longer at those selected input settings) to make forward predictions for next few weeks, which is shown in dark grey between week 10 and 20. In the next phase, when observed data between week 10 and 20 is available, the dark grey lines are constrained by the new likelihood producing posterior samples from the simulator – shown in dark green lines. To make predictions

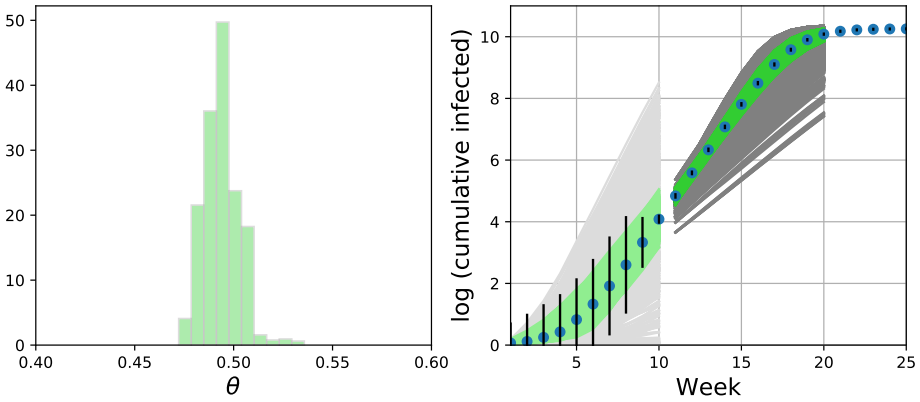


Figure 3.7: Histogram of posterior conditioned on 20 weeks of data (left) and calibrated prediction in green (right).

for next time points, one would require to extend those dark green simulation output. Predictions

summaries are directly obtained from already available posterior simulation response. Fig 3.8 compares the two posteriors $p(\boldsymbol{\theta}|\boldsymbol{y}^{(1)})$ and $p(\boldsymbol{\theta}|\boldsymbol{y}^{(1)}, \boldsymbol{y}^{(2)})$, and shows the constriction of parameter uncertainty after observing more data.

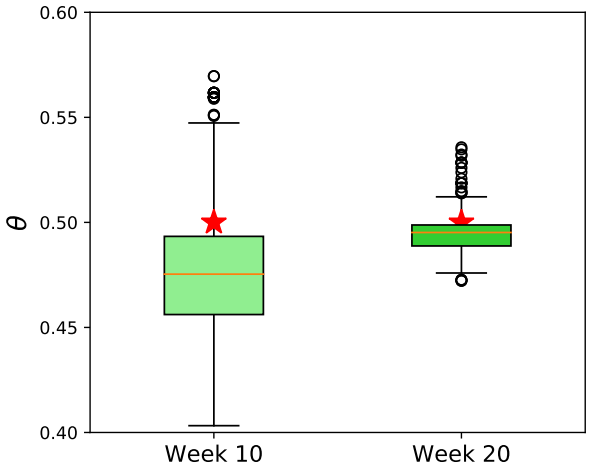


Figure 3.8: Boxplot of two posteriors $p(\boldsymbol{\theta}|\boldsymbol{y}^{(1)})$ and $p(\boldsymbol{\theta}|\boldsymbol{y}^{(1)}, \boldsymbol{y}^{(2)})$. Red star shows the actual value of $\boldsymbol{\theta}$.

3.4 Spatial Calibration

In this section we attempt to perform spatial calibration by employing the same sequential monte carlo based calibration procedure. Spatial calibration in epidemiology has received few attentions, where scientists aim to calibrate a disease simulation model under spatially observed data. An example of such attempt can be found in [144] in the context of predicting the cholera epidemic in Haiti in 2010 using a stochastic simulation model, whereas, authors model global spread of influenza in [10] among few other applications. However, the central focus of such studies have been the modeling aspect, with little or no attention towards quantifying the uncertainties in the model, or in the observe data used for calibration, or the calibration process itself. [119] uses a metapopulation model to characterize and forecast seasonal influenza in 6 south eastern states in US by using ensemble kalman filter [50] as calibration tool, similar to an application in [77] exploiting

the linearity assumption of the computer model. Contrary to that, our monte carlo based procedure is a general solution to any kind of stochastic computer model, and easy to implement and execute when the actual simulation runs fairly quickly. Complex disease models are highly non-linear in nature, restricting use of simple kalman filter type tools. Instead a direct approach can be proven to be more appropriate in such cases.

Fig 3.9 shows simulated observed data from the patch model along with it's uncertainty, similar to Fig 3.3, except now we consider each of 5 boroughs in NYC separately. As before, we consider transmissiblity as calibration parameter, fixing other patch model parameters at historical values. Algorithm 1 and 2 do not need to be modified here, rather just the data likelihood is constructed

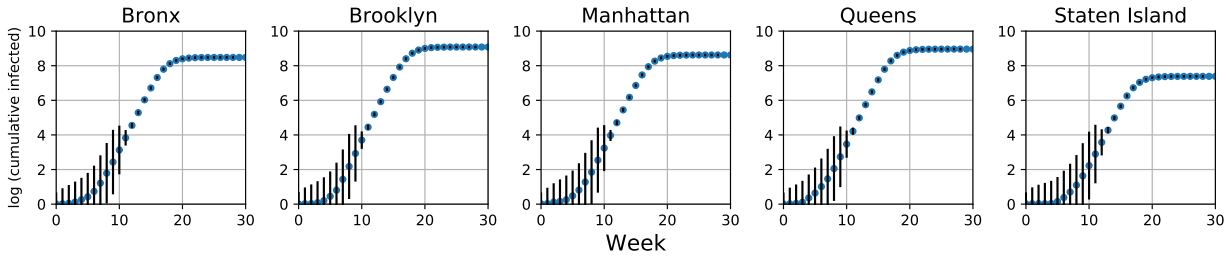


Figure 3.9: Patch model output from flu simulation in five boroughs of New York city, with 1-sd error bars.

accordingly. The data model can be re-written as:

$$\boldsymbol{y}_b = \eta(\boldsymbol{\theta}, r_{\boldsymbol{\theta}})_b + \boldsymbol{e_b}, \quad b = 1, \cdots, 5,$$

where the subscript b is for boroughs. $\boldsymbol{e}_i$ is assumed to be independent of $\boldsymbol{e}_j$ for any $i \neq j$ – the dependence within boroughs is encrypted in the commute information that goes into the simulation. Following previous example, the likelihood for $\boldsymbol{y}_b$ (each borough) is constructed independently using gaussian likelihood:

$$L(\boldsymbol{y}_b|\boldsymbol{\theta}_i, r_{\boldsymbol{\theta}}, \eta) \propto \lambda_y^{-5}|\Sigma_{y_b}|^{-1}(\boldsymbol{y}_b - \eta(\boldsymbol{\theta}, r_{\boldsymbol{\theta}})_b)'|\Sigma_{y_b}|^{-1/2}(\boldsymbol{y}_b - \eta(\boldsymbol{\theta}, r_{\boldsymbol{\theta}})_b).$$

The full likelihood is nothing but product of 5 individual likelihoods for 5 boroughs following the independence assumption.

$$L(\boldsymbol{y}|\boldsymbol{\theta}_i, r_{\boldsymbol{\theta}}, \eta) = \prod_{b=1}^{5} L(\boldsymbol{y}_b|\boldsymbol{\theta}_i, r_{\boldsymbol{\theta}}, \eta).$$

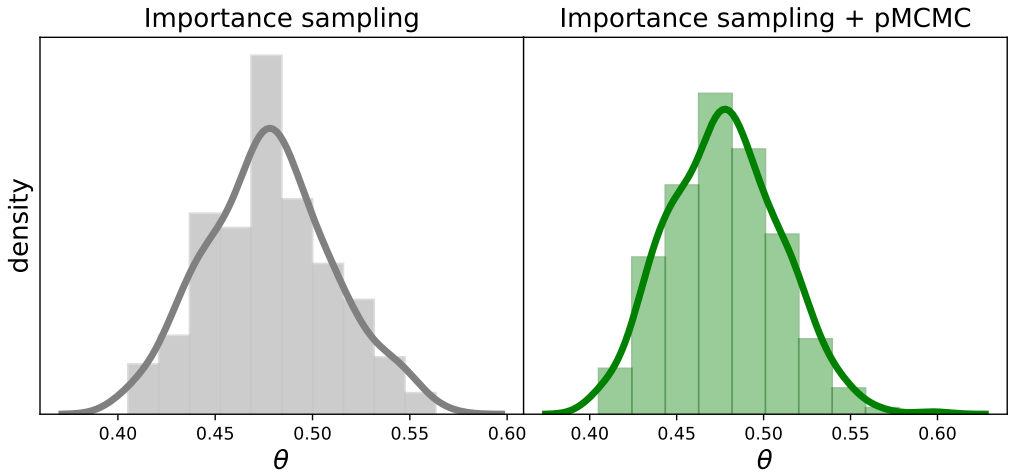


Figure 3.10: Posterior samples from calibrated spatial simulator until week 10 from algorithm 1 (left) and algorithm 1 + 2 (right).

As expected, the posterior at week 10 shows larger uncertainties on the input parameter $\boldsymbol{\theta}$ in Fig 3.10 due to high error variance in the begining of the epidemic. Additional posterior samples are obtained by a MCMC step using random walk proposal on already available samples from importance sampling. The posterior realizations from calibrated simulator until week 10 are shown in light green in Fig 3.11, and prediction beyond week 10 is given by dark grey curves. The next batch of calibration is performed using the data between week 10 and week 20, and the posterior uncertainty is much narrower as data constricts the input parameter in a very small interval. Since most of the epidemic is stable after week 10, very little uncertainty is remaining in the value of input parameter after observing first 10 weeks of epidemic. The reduction in uncertainty in $\boldsymbol{\theta}$ is also evident from Fig 3.12.

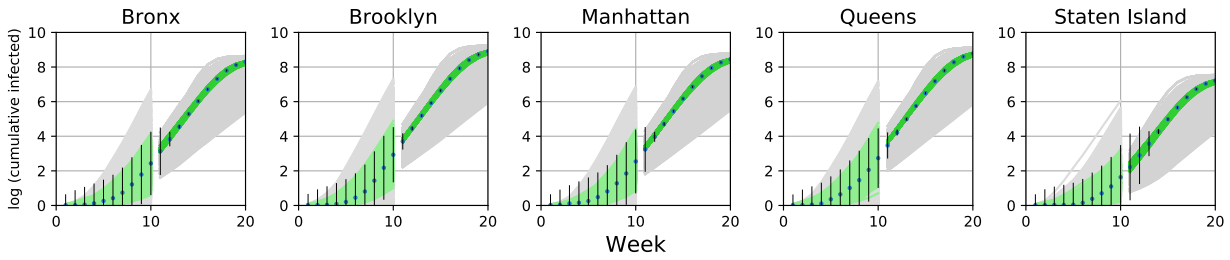


Figure 3.11: The figure shows posterior realizations from multistage calibrated patch model for each borough in NYC. The light grey curves between week 1 and 10 shows prior realizations, whereas light green curves are posterior simulation output from first calibration using week 1-10 data. Dark grey curves between week 10 and 20 are posterior predictions based on previous 10 weeks of data, and dark green curves are posterior realizations using week 10-20 data.

3.4.1 Discussion

This method explains and demonstrates how a simple sequential monte carlo idea can be adopted for calibrating a stochastic computer model with multivariate output in order to make prediction and UQ in a Bayesian framework which accounts for uncertainties from simulation parameter settings being unknown and the stochastic nature of the patch model. However, we want to emphasize before discussing advantages and disadvantages of the proposed framework that our target problems are not those which deal with very expensive computer models with limited budget, both in time and computing resources. It only makes sense when an 'online' inference is sought after, such as applications having time aspect, or several calibration criteria (dependent or independent) so that likelihood can be easily factorized in smaller components, helping with the bigger problem by decomposing it into smaller calibration problems.

A huge advantage in sequential calibration (or filtering) comes from the fact that it does not involve any extra statistical or mathematical model unlike in the case of an emulator, hence one less uncertainty to be taken care of. Building an accurate emulator for stochastic computer model in itself is an involved process with no general solution, and requires sincere considerations into choosing the right covariance structure, finding an optimal design for generating training simulation

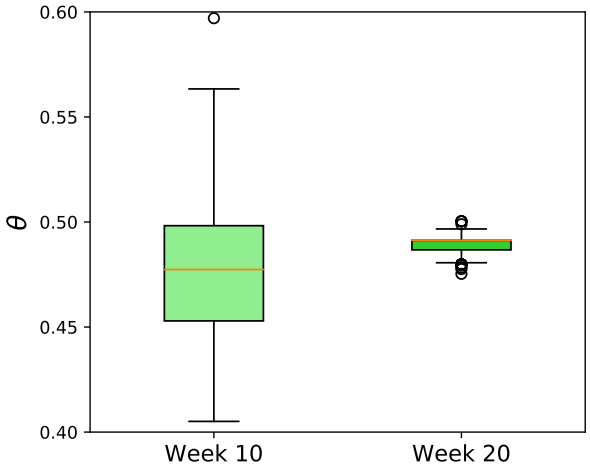


Figure 3.12: Boxplot of two posteriors $p(\boldsymbol{\theta}|\boldsymbol{y}^{(1)})$ and $p(\boldsymbol{\theta}|\boldsymbol{y}^{(1)}, \boldsymbol{y}^{(2)})$. Red star shows the actual value of $\boldsymbol{\theta}$.

data, and inventing ways to represent replicate variability. Several GP emulation techniques also rely on sequential design strategies which select next design point where actual simulation should be run sequentially by optimizing some criteria (e.g. minimizing mean square error). In those cases, one assumes the ability to run the actual simulation frequently as part of building the emulator. Similarly in our case, we run the actual simulation multiple times, however, unlike in a sequential design, the actual simulations can be performed in batches, for a bunch of input setting together, possibly executing them in parallel whenever possible. Since there is no emulator involved, the posterior uncertainty is readily available, and any standard sampling technique would be apt (importance sampling, MCMC, accept-reject etc.) to generate posterior realizations. However, to make the best use of probably expensive simulation we use an importance sample to initially generate some realizations from the posterior, and an subsequent MCMC step adds more samples with few rejections. Starting at a high density region ensures the algorithm produces more plausible input parameters than implausible ones.

The other advantage is particular to application such as epidemic simulations. Suit of epidemic simulators are not only useful to model and make predictions for an ongoing epidemic, but they are extensively used by policy makers to conduct what-if scenarios related to various intervention

strategies for example, which are otherwise impossible or unethical to experiment in reality. ABMs in particular stand out due to it's flexibility and adaptability to complex interventions, even in the middle of a simulation. A particular example would be studying the effect of school closures at the middle of an ongoing epidemic. Calibration which is done in sequential manner can easily entertain such scenarios as no additional update is required in the calibration setup, contrary to any emulation based techniques where one would have to build a new emulation for updated computer model (i.e. closing schools in ABM).

Predicting spatial spread of infectious disease is receiving significant attentions, as it helps refining and shaping efficient local intervention strategies, by understanding the local behavior of the epidemic. While few groups of scientists have taken a shot at this problem [10, 119, 144], scalability still remains a valid question of interest. We have demonstrated our approach for 5 spatially connected regions, and have understood the potential of extending this for entire US which has approximately 3000 counties (if we consider each counties as separate patch in the patch model). By leveraging the spatial dependency in the mechanistic model, the data likelihood and posterior are simple and requires no superficial assumption.

Discrepancy plays a significant role in computer model calibration, specially in applications related to epidemic as we have seen in Fig 2.16. Although we have not talked about discrepancy, adding a component in the data model will change the likelihood by a factor, and in case of GP based discrepancy [76, 88] only the covariance will need an update, keeping everything else same. However, an appropriate discrepancy model may benefit from prior knowledge, for example, putting higher discrepancies at the beginning of a flu season when less data is available and smaller discrepancies in the later stage when the infection reaches mass population. We end this section by acknowledging that while many computer simulation experiments do not come under the category where one can leverage unlimited budget, there are still examples available in epidemiology, climate models, finance etc., where ABMs are used and model perturbation within an ongoing simulation is entertained. Sequential style calibration and UQ approaches are well suited for such situations and may even be proven to be efficient than redoing an extra expensive emulation step.

Chapter 4

Future Direction

Three different methods have been proposed and discussed in detail so far in the context of stochastic computer model calibration – two of which are based on emulators while the other relies on repeated runs of the actual simulation model. We have also seen a quantile based emulator (section 2.2.2) and a mixture density emulator (section 2.3) which take care of the non-gaussian replicate variability in the computer model. While there has been growing popularity of stochastic computer simulations in physical, environmental, engineering, economic processes in recent years, very few attempts have been made in developing new techniques or updating existing methodologies to handle this stochastic nature in the parameter inference. This document is a venture in that direction and presents an avenue towards further improvisation and research of the proposed solutions. In this chapter, we discuss about few possible improvisations that follow directly from the previous chapters, along with examples whenever applicable.

4.1 Mixture Density Emulator

We have already seen Gaussian Process [124] in action in emulating quantiles (section 2.2.2) or the mean and variance (section 2.3) of the replicated simulation output. While the quantile approach approximates the simulator distribution function by estimating the quantiles, the success in doing so relies on availability of sufficiently large number of simulation replicates. On the other hand mixture density emulator is based on approximating any arbitrary distribution by a mixture of finitely many normals, given that the number of mixture component is known. Although in many applications

(such as disease modeling) the assumption of known and fixed number of mixture components is not unrealistic, a direct extension would be to treat the number to be unknown. In the bayesian setting that translates to putting an uncertainty on the number of mixture component through prior distribution. Dirichlet process mixture (DPM) models [47, 81] are popular in non-parametric Bayesian literature, which incorporate Dirichlet Process (DP) priors proposed in Ferguson [52], Ferguson et al. [53] in a Bayesian hierarchical setup, producing a flexible class of models for non-normal data. Due to availability of simple and efficient MCMC [81] methods, DPM models are found in variety of applications, see [12, 69, 91] for example. DPM in its vanilla version is prior on univariate or multivariate random distributions for discrete measurements. By convolving with a continuous measure, the support is then extended for continuous measurements.

To model $\{y_1, \cdots, y_n\}$ as DPM, we write the generative model as $f(y) = \sum_{k=1}^{\infty} \pi_k f_k(y|\boldsymbol{\theta}_k)$, where each $\boldsymbol{\theta}_k$ are independent draws from a realization G of a Dirichlet Process parameterized by a base distribution G_0 and precision α. The mixing proportion π_k is sampled from a stick breaking process. According to Ishwaran and James [80] G admits a stick breaking prior if G can be written as $G = \sum_1^{\infty} \pi_h \delta_{\boldsymbol{\theta}_h}$, $0 \leq \pi_h \leq 1$ with $\sum_1^{\infty} \pi_h = 1$, where $\delta_{\boldsymbol{\theta}_h}$ is a discrete measure concentrated at $\boldsymbol{\theta}_k$, $\pi_h = v_h \prod_{l<h}(1 - v_l)$ are random weights with v_h independently drawn from Beta(1, α), and $\boldsymbol{\theta}$ independently drawn from G_0 with G_0 being the base probability measure. The general class of stick breaking priors are discussed in Ishwaran and James [80], whereas the particular DP case is first coined by Sethuraman [137]. The almost sure discreteness of G is a corollary to the stick break representation of DP [137], and a variety of distribution can be approximated with arbitrary precision due to kolmogorov existence theorem [78]. The usual DPM can be extended to incorporate predictor dependence in both $\boldsymbol{\theta}_k$ and π_k, providing a flexible non-parametric set up to model input-output relations. In the next section, we review some existing work on the extensions of DPM.

4.1.1 Related Work

In the presence of input x one can model $\boldsymbol{\theta}_k$ as function of input x, which constitutes a random process on the space of X, and the observation $\{y(x) : x \in X\}$ is assumed to be a sample from a realization of the random process on X. A gaussian random field assumption would lead to a mulvariate gaussian likelihood for the observations $y(x_i), i = 1, \cdots, n$, where $\{x_i\}_{i=1}^n$ are the observed input locations. A possible deviation from this restrictive gaussian assumption is achieved by putting a random probability measure on the space of distribution functions defined on X. Gelfand et al. [59] denotes this new type of random process as *spatial dirichlet process* (SDP), since a spatial structure is put on on the distribution of $\boldsymbol{\theta}_k$. By extending the usual stick breaking representation of G, a realization from such spatial process can be written as $G = \sum_1^\infty \pi_h \delta_{\boldsymbol{\theta}_{h,X}}$, where $\boldsymbol{\theta}_{h,X}$ with the new subscript X refers to a collection of marginally normally distributed random variables at locations where the data is observed. In other words $\{\boldsymbol{\theta}_h(x_1), \cdots, \boldsymbol{\theta}_h(x_n)\}$ is a random realization from a stochastic process G_0, aka the base distribution of SDP. Hence G induces a random probability measure on the space of distribution functions for $y(x_i), i = 1, \cdots, n$. The specification for w and α remain same as before, i.e., they do not depend on x. In a nutshell SDP provides a straight forward extension over traditional DPM model by treating each $\boldsymbol{\theta}_k$ as realization from a stochastic process.

Spatial dirichlet process can also be regarded as a special case of dependent dirichlet process (DDP) [102], which provides a larger and more general framework to allow for a collection of non-parametric distributions, with dependent realizations at the covriate level. The theoretical existence guarantee and properties of DDP are laid down in MacEachern [102], and similar properties follow for SDP. Adaptation of DDP can be found in ANOVA models for random probabilty measures [37], time series models [23], and in stochastic ordering [46]. However, applications based on DDP use fixed weights w and introduces dependence only through $\boldsymbol{\theta}$, limiting the scope of recycling those models for our purpose, which we will discuss later.

Alternatively, Griffin and Steel [68] introduces order based DDP (πDDP) to allow input dependent

weights along with input dependent $\boldsymbol{\theta}$, by defining a stochastic process for weight function π taking values on the natural number space. The authors use a point process to derive and infer the parameters of the stochastic process for π. An extension of SDP is found in Duan et al. [43], where a multivariate stick breaking prior is introduced for input dependent π. A related but different approach inject non-stationarity by using mixture of more than one GPs, each for mutually exclusive subset of the input space X [123]. A dirichlet process prior is used to derive gating network based on which the input space X is divided into subspaces, and individual GP regression is used for each input subspace. Another way of modeling a collection of dependent random distributions is to take convex combinations of indpendent DPs [109], which has been used and extended in Dunson [44], Dunson and Park [45] and Dunson and Peddada [46]. Similar to infinite mixture of GPs [123], a local DP is proposed in Chung and Dunson [29]. In a local DP, the random distribution at any input location is constructed by weighted sum of concentrated mass, where the components in the sum is chosen according to neighborhood structure of few pre-defined input locations, or where the data is observed.

4.1.2 Proposed Framework

Dirichlet process prior is widely popular due to its properties. The authors in [29] note down few such desirable properties for an input dependent process G_x. They are – (1) the dependence between G_x and $G_{x'}$ should depend on the distance between x and x', (2) the expectation and variance of G_x and covariance between G_x and $G_{x'}$ are expected to be simple and interpretable, (3) at each input x, the process marginally reduces to simple DP prior, and lastly (4) an efficient and easily implementable MCMC is available for posterior computation. The authors argue that none of the said extensions of DPM achieve all four properties individually except their proposed local dirichlet process. In next paragraphs we discuss a motivating idea of using an input dependent DP prior to model a stochastic computer model emulator.

As our motivating application we refer to the ABM example in chapter 2 where we want to model

the simulation output as function of $\boldsymbol{\theta}$ and time. For the sake of simplicity we would demonstrate the idea for the case where the input is a scalar, i.e., in our ABM example we would like to predict $\eta(t)$ when $\boldsymbol{\theta}$ is fixed (we omit $\boldsymbol{\theta}$ for rest of the discussion). Our primary focus remains the same, i.e., to model the distribution of the random quantity $\eta(t)$ at any t by a DP mixture (property (3)); and secondly, to model the spatial dependence between the distributions at t and t' (property (1)).

Generative Model

Let $t_1, t_2, \cdots, t_m$ be a fixed set of input points. To start with we define independent dirichlet process at each t_i. Let $G_0^{(i)}$ be the base measure for input location t_i, and $\alpha^{(i)}$ be the corresponding precision. Then from standard DP model we write the following for each $i = 1, \cdots, m$:

$$\begin{aligned} G^{(i)} &\sim DP(\alpha^{(i)} G_0^{(i)}) \\ \phi_j^{(i)} | G^{(i)} &\overset{i.i.d.}{\sim} G^{(i)} \\ \eta(t_i) | \phi_j^{(i)} &\overset{i.i.d.}{\sim} f(\cdot | \phi_j^{(i)}) \end{aligned}$$

Once we establish marginal DP at each input location t_i, the input dependency in $G^{(i)}$, or equivalently in $\phi^{(i)}$ is taken care of by placing a GP prior as function of $t_1, t_2, \cdots, t_m$. Following the stick breaking representation of DP [137] we can write the marginal distribution of $\eta(t_i)$ as

$$f(\eta(t_i) | \boldsymbol{\phi}^{(i)}, \boldsymbol{\pi}^{(i)}) = \sum_{k=1}^{\infty} \pi_k^{(i)} f(\eta(t_i) | \phi_k^{(i)}),$$

where, $\phi_j^{(i)}$)s are defined as $\pi_j^{(i)} = v_j^{(i)} \prod_{l<j} (1 - v_l^{(i)})$ with each $v_j^{(i)} \overset{i.i.d.}{\sim} \beta(1, \alpha^{(i)})$. As the number of mixing components at each t_i is unknown, and so the membership of the ϕ's at t_i, it posses a challenge to define GP priors on them. Order based dependent dirichlet process (πDDP) [68] is an attempt in solving the same problem. Here we propose a much simpler modeling scheme based on ordering of the inferred values of $\phi_j^{(i)}$.

Parameter Estimation

The simulation is run at each input location, each with r_j replicates, $j = 1, \cdots, m$, resulting in an output array

$$\{(\eta_1(t_1), \cdots, \eta_{r_1}(t_1)), \cdots, \cdots, (\eta_1(t_m), \cdots, \eta_{r_m}(t_m))\}.$$

By approximating the infinite sum in the stick breaking prior by a sufficiently large N, we can sample $\{\phi_j^{(i)}\}_{j=1}^N$ and $\{\pi_j^{(i)}\}_{j=1}^N$ from appropriate posterior for $i = 1, \cdots, m$. Next, we arrange the ϕ's in increasing order at each t_i such that, $\phi_1^{(i)} < \phi_2^{(i)} < \cdots < \phi_N^{(i)}$. Once these are ordered at each t_i, we can now train N GP models, one for each of the ordered ϕ, i.e., $(\phi_j^{(1)}, \cdots, \phi_j^{(m)})$, and another N GP models for correspondingly ordered π, i.e., $(\pi_j^{(1)}, \cdots, \pi_j^{(m)})$.

$$\boldsymbol{\phi}^{(i)} \sim GP(0, R(t_h, t_{h'})), \;\; \text{logit}(\boldsymbol{\pi}^{(i)}) \sim GP(0, R(t_h, t_{h'})) \;\; i = 1, \cdots, m.$$

Predicted simulator distribution at a new t^* can then be obtained by:

$$\hat{f}(\eta(t^*)|\cdots) = \sum_{k=1}^{N} \hat{\pi}_k^{(i)} f(\eta(t^*)|\hat{\phi}_k^{(i)})$$

4.2 Discrepancy in Sequential Calibration

In chapter 3 we have discussed a proposed novel strategy to sequentially calibrate a stochastic computer model, by factorizing the full posterior into conditionally independent components and by successively selecting plausible model configurations conditioned on the observed data. The innovation is two fold here – (1) combining an importance sampling step with a one step metropolis hasting sampling, and (2) approximating the likelihood from a stochastic simulation. The basic idea in this framework relies on exploring intermediate posterior of input parameters, conditioned on subset of data. In other words, it is similar to history matching [2] in a sense that we keep refining the parameter space by selecting plausible values successively from that space.

Modeling the systematic discrepancy and making forward prediction in any computer experiment is a challenging problem. Kennedy and O'Hagan [88] suggested an all-purpose GP model for discrepancy, Higdon et al. [76] uses an informed structure while building such GP discrepancy, and Tuo and Jeff Wu [145] points out potential identifiability issue arising from using an additive discrepancy in the calibration setup. We argue that with prior information (often application specific) and proper regularization, a discrepancy model can bring insight into missing physics in the computational model. Hence, incorporating a discrepancy model in the sequential calibration framework is a research question to pursue in future. A related idea can be found in Higdon et al. [77], where the authors discuss the use of GP discrepancy in a ensemble kalman filter calibration which relies on exploiting the linearity assumption between input and output through gaussian likelihoods.

A different approach to deal with model inadequacy is to modify the input parameter by adding a discrepancy factor to it, which is also known as embed uncertainty (Huan et. al. Sandia National Lab). The model then becomes $\boldsymbol{y}(x) = \eta(x, \boldsymbol{\theta} + \delta) + \boldsymbol{e}$. The embed δ into the emulator allows for targeted error placement while conserving the model structure and physical constraints, and orthogonally distinguishes data error from the model. However, the embed discrepancy may depend on extra parameters which need to be calibrated.

For a stochastic computer simulation we rewrite the data model as $\boldsymbol{y}(x) = \eta(x, \boldsymbol{\theta}, r_{\boldsymbol{\theta}}) + \boldsymbol{\delta}(x) + \boldsymbol{e}$, with an additional term $\boldsymbol{\delta}$ for discrepancy. Note the $r_{\boldsymbol{\theta}}$ in the emulator, which indicates that not only we need to find the best value(s) of $\boldsymbol{\theta}$, but also the replicates $r_{\boldsymbol{\theta}}$ indexed by $\boldsymbol{\theta}$ which are compatible with $\boldsymbol{y}$. If assuming a statistical model for $\boldsymbol{\delta}$, additional hyperparameter estimation needs to be done along with $\boldsymbol{\theta}$. Importance sampling is known to suffer from curse of dimensionality problem, requiring one to densely sample the high-dimensional input space which subsequently increases the computational burden as the actual simulation need to be run at each densely input sample. However even then a low monte-carlo error in the posterior estimation is not guaranteed. Hence, a reasonable model for discrepancy and an efficient estimation procedure remain to be valid questions to be answered in subsequent work. We end by commenting that, like any other

calibration framework, making extrapolative prediction from such a model would always require extra attention and careful consideration of appropriate uncertainties.

Bibliography

[1] Linda JS Allen and Amy M Burgin. Comparison of deterministic and stochastic sis and sir models in discrete time. *Mathematical biosciences*, 163(1):1–33, 2000.

[2] Ioannis Andrianakis, Nicky McCreesh, Ian Vernon, Trevelyan J McKinley, Jeremy E Oakley, Rebecca N Nsubuga, Michael Goldstein, and Richard G White. Efficient history matching of a high dimensional individual-based hiv transmission model. *SIAM/ASA Journal on Uncertainty Quantification*, 5(1):694–719, 2017.

[3] Christophe Andrieu, Nando De Freitas, Arnaud Doucet, and Michael I Jordan. An introduction to mcmc for machine learning. *Machine learning*, 50(1-2):5–43, 2003.

[4] Bruce Ankenman, Barry L Nelson, and Jeremy Staum. Stochastic kriging for simulation metamodeling. *Operations research*, 58(2):371–382, 2010.

[5] SR Arridge, JP Kaipio, V Kolehmainen, M Schweiger, E Somersalo, T Tarvainen, and M Vauhkonen. Approximation errors and model reduction with an application in optical diffusion tomography. *Inverse problems*, 22(1):175, 2006.

[6] S ren Asmussen and Reuven Y Rubinstein. Response surface estimation and sensitivity analysis via efficient change of measure. *Stochastic Models*, 9(3):313–339, 1993.

[7] Yanping Bai and Zhen Jin. Prediction of sars epidemic by bp neural networks with online prediction strategy. *Chaos, Solitons & Fractals*, 26(2):559–569, 2005.

[8] Paolo Bajardi, Chiara Poletto, Jose J Ramasco, Michele Tizzoni, Vittoria Colizza, and Alessandro Vespignani. Human mobility networks, travel restrictions, and the global spread of 2009 h1n1 pandemic. *PloS one*, 6(1):e16591, 2011.

[9] Duygu Balcan, Vittoria Colizza, Bruno Gonçalves, Hao Hu, José J Ramasco, and Alessandro Vespignani. Multiscale mobility networks and the spatial spreading of infectious diseases. *Proceedings of the National Academy of Sciences*, 106(51):21484–21489, 2009.

[10] Duygu Balcan, Bruno Gonçalves, Hao Hu, José J Ramasco, Vittoria Colizza, and Alessandro Vespignani. Modeling the spatial spread of infectious diseases: The global epidemic and mobility computational model. *Journal of computational science*, 1(3):132–145, 2010.

[11] Maria J Bayarri, James O Berger, Rui Paulo, Jerry Sacks, John A Cafeo, James Cavendish, Chin-Hsu Lin, and Jian Tu. A framework for validation of computer models. *Technometrics*, 49(2):138–154, 2007. doi: 10.1198/004017007000000092. URL https://doi.org/10.1198/004017007000000092.

[12] Matthew J Beal, Zoubin Ghahramani, and Carl E Rasmussen. The infinite hidden markov model. In *Advances in neural information processing systems*, pages 577–584, 2002.

[13] Einat Neumann Ben-Ari and David M Steinberg. Modeling data from computer experiments: an empirical comparison of kriging with mars and projection pursuit regression. *Quality Engineering*, 19(4):327–338, 2007.

[14] José M Bernardo and Adrian FM Smith. *Bayesian theory*, volume 405. John Wiley & Sons, 2009.

[15] D Bernoulli. Essai d'une nouvelle analyse de la mortalite cause par la petite verole et des a vantages de finoculation pour la prevenir. mem math. & phys. de i acad. roy. sci. *Hist. de I Acad. Paris*, 1, 1766.

[16] Julian Besag, Peter Green, David Higdon, and Kerrie Mengersen. Bayesian computation and stochastic systems. *Statistical science*, pages 3–41, 1995.

[17] Mickael Binois and Robert B. Gramacy. `hetGP`*: Heteroskedastic Gaussian Process Modeling and Design under Replication*, 2018. R package version 1.1.1.

[18] Mickaël Binois, Robert B. Gramacy, and Mike Ludkovski. Practical heteroscedastic gaussian process modeling for large simulation experiments. *Journal of Computational and Graphical Statistics*, 27(4):808–821, 2018. doi: 10.1080/10618600.2018.1458625. URL https://doi.org/10.1080/10618600.2018.1458625.

[19] David Bishai, Benjamin Johns, Divya Nair, Juliet Nabyonga-Orem, Braka Fiona-Makmot, Emily Simons, and Alya Dabbagh. The cost-effectiveness of supplementary immunization activities for measles: a stochastic model for uganda. *The Journal of infectious diseases*, 204 (suppl_1):S107–S115, 2011.

[20] Alexis Boukouvalas, Dan Cornford, and Milan Stehlík. Optimal design for correlated processes with input-dependent noise. *Computational Statistics & Data Analysis*, 71:1088–1102, 2014.

[21] George EP Box and Norman R Draper. *Response surfaces, mixtures, and ridge analyses*, volume 649. John Wiley & Sons, 2007.

[22] Jennỳ Brynjarsdóttir and Anthony O Hagan. Learning about physical parameters: The importance of model discrepancy. *Inverse Problems*, 30(11):114007, 2014.

[23] Francois Caron, Manuel Davy, Arnaud Doucet, Emmanuel Duflos, and Philippe Vanheeghe. Bayesian inference for linear dynamic models with dirichlet process mixtures. *IEEE Transactions on Signal Processing*, 56(1):71–84, 2007.

[24] George Casella and Roger L Berger. *Statistical inference*, volume 2. Duxbury Pacific Grove, CA, 2002.

[25] KC Chang and Donghai He. Efficient iterative importance sampling inference for dynamic bayesian networks. In *2005 7th International Conference on Information Fusion*, volume 1, pages 7–pp. IEEE, 2005.

[26] Keewhan Choi and Stephen B Thacker. An evaluation of influenza mortality surveillance,

1962–1979: I. time series forecasts of expected pneumonia and influenza deaths. *American journal of epidemiology*, 113(3):215–226, 1981.

[27] Jean-Paul Chretien, Dylan George, Jeffrey Shaman, Rohit A Chitale, and F Ellis McKenzie. Influenza forecasting in human populations: a scoping review. *PloS one*, 9(4):e94130, 2014.

[28] Nicholas A Christakis and James H Fowler. Social network sensors for early detection of contagious outbreaks. *PloS one*, 5(9):e12948, 2010.

[29] Yeonseung Chung and David B Dunson. The local dirichlet process. *Annals of the Institute of Statistical Mathematics*, 63(1):59–80, 2011.

[30] Stella M Clarke, Michael F Zaeh, and Jan H Griebsch. Predicting haptic data with support vector regression for telepresence applications. In *Design and application of hybrid intelligent systems*, pages 572–581. IOS Press, 2003.

[31] Anne Cori, Christl A Donnelly, Ilaria Dorigatti, Neil M Ferguson, Christophe Fraser, Tini Garske, Thibaut Jombart, Gemma Nedjati-Gilani, Pierre Nouvellet, Steven Riley, et al. Key data for outbreak evaluation: building on the ebola experience. *Phil. Trans. R. Soc. B*, 372 (1721):20160371, 2017.

[32] JEAN-MARIE CORNUET, JEAN-MICHEL MARIN, Antonietta Mira, and Christian P Robert. Adaptive multiple importance sampling. *Scandinavian Journal of Statistics*, 39 (4):798–812, 2012.

[33] National Research Council. *Assessing the Reliability of Complex Models: Mathematical and Statistical Foundations of Verification, Validation, and Uncertainty Quantification*. The National Academies Press, Washington, DC, 2012. ISBN 978-0-309-25634-6. doi: 10.17226/13395. URL https://www.nap.edu/catalog/13395/assessing-the-reliability-of-complex-models-mathematical-and-statistical-foundations.

[34] Peter S Craig, Michael Goldstein, Jonathan C Rougier, and Allan H Seheult. Bayesian forecasting for complex systems using computer simulators. *Journal of the American Statistical Association*, 96(454):717–729, 2001.

[35] Carla Currin, Toby Mitchell, Max Morris, and Don Ylvisaker. Bayesian prediction of deterministic functions, with applications to the design and analysis of computer experiments. *Journal of the American Statistical Association*, 86(416):953–963, 1991.

[36] A Philip Dawid, Paola Sebastiani, et al. Coherent dispersion criteria for optimal experimental design. *The Annals of Statistics*, 27(1):65–81, 1999.

[37] Maria De Iorio, Peter Müller, Gary L Rosner, and Steven N MacEachern. An anova model for dependent random measures. *Journal of the American Statistical Association*, 99(465): 205–215, 2004.

[38] Arthur P Dempster, Nan M Laird, and Donald B Rubin. Maximum likelihood from incomplete data via the em algorithm. *Journal of the Royal Statistical Society: Series B (Methodological)*, 39(1):1–22, 1977.

[39] Peter J Diggle, Jonathan A Tawn, and RA Moyeed. Model-based geostatistics. *Journal of the Royal Statistical Society: Series C (Applied Statistics)*, 47(3):299–350, 1998.

[40] Arnaud Doucet. Sequential monte carlo methods & particle filters resources, 0000.

[41] Arnaud Doucet, Nando De Freitas, and Neil Gordon. An introduction to sequential monte carlo methods. In *Sequential Monte Carlo methods in practice*, pages 3–14. Springer, 2001.

[42] Dorin Drignei, Chris E Forest, Doug Nychka, et al. Parameter estimation for computationally intensive nonlinear regression with an application to climate modeling. *The Annals of Applied Statistics*, 2(4):1217–1230, 2008.

[43] Jason A Duan, Michele Guindani, and Alan E Gelfand. Generalized spatial dirichlet process models. *Biometrika*, 94(4):809–825, 2007.

[44] David B Dunson. Bayesian dynamic modeling of latent trait distributions. *Biostatistics*, 7 (4):551–568, 2006.

[45] David B Dunson and Ju-Hyun Park. Kernel stick-breaking processes. *Biometrika*, 95(2): 307–323, 2008.

[46] David B Dunson and Shyamal D Peddada. Bayesian nonparametric inference on stochastic ordering. *Biometrika*, 95(4):859–874, 2008.

[47] Michael D Escobar. Estimating normal means with a dirichlet process prior. *Journal of the American Statistical Association*, 89(425):268–277, 1994.

[48] Michael D Escobar and Mike West. Bayesian density estimation and inference using mixtures. *Journal of the american statistical association*, 90(430):577–588, 1995.

[49] Stephen Eubank, Hasan Guclu, VS Anil Kumar, Madhav V Marathe, Aravind Srinivasan, Zoltan Toroczkai, and Nan Wang. Modelling disease outbreaks in realistic urban social networks. *Nature*, 429(6988):180, 2004.

[50] Geir Evensen. The ensemble kalman filter: Theoretical formulation and practical implementation. *Ocean dynamics*, 53(4):343–367, 2003.

[51] Arindam Fadikar, Dave Higdon, Jiangzhuo Chen, Bryan Lewis, Srinivasan Venkatramanan, and Madhav Marathe. Calibrating a stochastic, agent-based model using quantile-based emulation. *SIAM/ASA Journal on Uncertainty Quantification*, 6(4):1685–1706, 2018.

[52] Thomas S Ferguson. A bayesian analysis of some nonparametric problems. *The annals of statistics*, pages 209–230, 1973.

[53] Thomas S Ferguson et al. Prior distributions on spaces of probability measures. *The annals of statistics*, 2(4):615–629, 1974.

[54] Roger Fletcher. *Practical methods of optimization*. John Wiley & Sons, 2013.

[55] Michael S Floater and Armin Iske. Multistep scattered data interpolation using compactly supported radial basis functions. *Journal of Computational and Applied Mathematics*, 73 (1-2):65–78, 1996.

[56] Chris Fraley and Adrian E Raftery. Model-based clustering, discriminant analysis, and density estimation. *Journal of the American statistical Association*, 97(458):611–631, 2002.

[57] J. Gattiker, K. Myers, B. Williams, D. Higdon, M. Carzolio, and A. Hoegh. Gaussian process-based sensitivity analysis and bayesian model calibration with gpmsa. In R. Ghanem, D. Higdon, and Houman Owhadi, editors, *Handbook of Uncertainty Quantification*, pages 1867–1907. Springer, Switzerland, 2016.

[58] Alan E Gelfand and Sujit K Ghosh. Model choice: a minimum posterior predictive loss approach. *Biometrika*, 85(1):1–11, 1998.

[59] Alan E Gelfand, Athanasios Kottas, and Steven N MacEachern. Bayesian nonparametric spatial modeling with dirichlet process mixing. *Journal of the American Statistical Association*, 100(471):1021–1035, 2005.

[60] Tilmann Gneiting and Adrian E Raftery. Strictly proper scoring rules, prediction, and estimation. *Journal of the American Statistical Association*, 102(477):359–378, 2007.

[61] Shirin Golchi, Derek R Bingham, Hugh Chipman, and David A Campbell. Monotone emulation of computer experiments. *SIAM/ASA Journal on Uncertainty Quantification*, 3(1): 370–392, 2015.

[62] Paul W Goldberg, Christopher KI Williams, and Christopher M Bishop. Regression with input-dependent noise: A gaussian process treatment. In *Advances in neural information processing systems*, pages 493–499, 1998.

[63] D. Goldsman, R. E. Nance, and J. R. Wilson. A brief history of simulation. In *Proceedings*

of the 2009 Winter Simulation Conference (WSC), pages 310–313, Dec 2009. doi: 10.1109/WSC.2009.5429341.

[64] Michael Goldstein and Jonathan Rougier. Bayes linear calibrated prediction for complex systems. *Journal of the American Statistical Association*, 101(475):1132–1143, 2006.

[65] Steven M Goodreau, Mark S Handcock, David R Hunter, Carter T Butts, and Martina Morris. A statnet tutorial. *Journal of statistical software*, 24(9):1, 2008.

[66] Robert B Gramacy and Herbert K H Lee. Bayesian treed gaussian process models with an application to computer modeling. *Journal of the American Statistical Association*, 103(483): 1119–1130, 2008.

[67] Todd L Graves. Automatic step size selection in random walk metropolis algorithms. *arXiv preprint arXiv:1103.5986*, 2011.

[68] Jim E Griffin and MF J Steel. Order-based dependent dirichlet processes. *Journal of the American statistical Association*, 101(473):179–194, 2006.

[69] Thomas L Griffiths, Michael I Jordan, Joshua B Tenenbaum, and David M Blei. Hierarchical topic models and the nested chinese restaurant process. In *Advances in neural information processing systems*, pages 17–24, 2004.

[70] A. Agashe M. Alam K. Alexander S. Arifuzzaman C. Barrett R. Beckman K. Bisset J. Chen Y. Chungbaek S. Eubank S. Gupta M. Khan C. Kuhlman E. Lofgren B. Lewis A. Marathe E. Nordberg C. Rivers P. Stretz S. Swarup A. Wilson D. Xie M. Marathe H. Mortveit, A. Adiga. Synthetic populations and interaction networks for guinea, liberia and sierra leone. *BI Technical Report*, 2015.

[71] W. K. Hastings. Monte Carlo sampling methods using Markov chains and their applications. *Biometrika*, 57:97–109, 1970.

[72] Simon Haykin. *Neural networks: a comprehensive foundation*. Prentice Hall PTR, 1994.

[73] Daniel A Henderson, Richard J Boys, Kim J Krishnan, Conor Lawless, and Darren J Wilkinson. Bayesian emulation and calibration of a stochastic computer model of mitochondrial dna deletions in substantia nigra neurons. *Journal of the American Statistical Association*, 104(485):76–87, 2009.

[74] Dave Higdon. Space and space-time modeling using process convolutions. In C. Anderson, V. Barnett, P. C. Chatwin, and A. H. El-Shaarawi, editors, *Quantitative Methods for Current Environmental Issues*, pages 37–56, London, 2002. Springer Verlag.

[75] Dave Higdon, Marc Kennedy, James C Cavendish, John A Cafeo, and Robert D Ryne. Combining field data and computer simulations for calibration and prediction. *SIAM Journal on Scientific Computing*, 26(2):448–466, 2004.

[76] Dave Higdon, James Gattiker, Brian Williams, and Maria Rightley. Computer model calibration using high-dimensional output. *Journal of the American Statistical Association*, 103 (482):570–583, 2008.

[77] Dave Higdon, Jim Gattiker, Earl Lawrence, Charles Jackson, Michael Tobis, Matt Pratola, Salman Habib, Katrin Heitmann, and Steve Price. Computer model calibration using the ensemble kalman filter. *Technometrics*, 55(4):488–500, 2013.

[78] Nils Lid Hjort, Chris Holmes, Peter Müller, and Stephen G Walker. *Bayesian nonparametrics*, volume 28. Cambridge University Press, 2010.

[79] Yonghong Huang, Kevin B Englehart, Bernard Hudgins, and Adrian DC Chan. A gaussian mixture model based classification scheme for myoelectric control of powered upper limb prostheses. *IEEE Transactions on Biomedical Engineering*, 52(11):1801–1811, 2005.

[80] Hemant Ishwaran and Lancelot F James. Gibbs sampling methods for stick-breaking priors. *Journal of the American Statistical Association*, 96(453):161–173, 2001.

[81] Hemant Ishwaran and Mahmoud Zarepour. Exact and approximate sum representations for the dirichlet process. *Canadian Journal of Statistics*, 30(2):269–283, 2002.

[82] Julien Jacques and Cristian Preda. Functional data clustering: a survey. *Advances in Data Analysis and Classification*, 8(3):231–255, 2014.

[83] Samuel M Jenness, Steven M Goodreau, and Martina Morris. Epimodel: an r package for mathematical modeling of infectious disease over networks. *Journal of statistical software*, 84, 2018.

[84] Erwin Jeremiah, Scott Sisson, Lucy Marshall, Rajeshwar Mehrotra, and Ashish Sharma. Bayesian calibration and uncertainty analysis of hydrological models: A comparison of adaptive metropolis and sequential monte carlo samplers. *Water Resources Research*, 47(7), 2011.

[85] Ruichen Jin, Wei Chen, and Timothy W Simpson. Comparative studies of metamodelling techniques under multiple modelling criteria. *Structural and multidisciplinary optimization*, 23(1):1–13, 2001.

[86] Nicholas Kantas, Arnaud Doucet, Sumeetpal Sindhu Singh, and Jan Marian Maciejowski. An overview of sequential monte carlo methods for parameter estimation in general state-space models. *IFAC Proceedings Volumes*, 42(10):774–785, 2009.

[87] Alexandros Karatzoglou, Alex Smola, Kurt Hornik, and Achim Zeileis. kernlab-an s4 package for kernel methods in r. *Journal of statistical software*, 11(9):1–20, 2004.

[88] Marc C. Kennedy and Anthony O'Hagan. Bayesian calibration of computer models. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 63(3):425–464, 2001. doi: 10.1111/1467-9868.00294. URL https://rss.onlinelibrary.wiley.com/doi/abs/10.1111/1467-9868.00294.

[89] William Ogilvy Kermack and Anderson G McKendrick. A contribution to the mathematical

theory of epidemics. *Proceedings of the royal society of london. Series A, Containing papers of a mathematical and physical character*, 115(772):700–721, 1927.

[90] Ji-hoon Kim, Tom Abel, Oscar Agertz, Greg L Bryan, Daniel Ceverino, Charlotte Christensen, Charlie Conroy, Avishai Dekel, Nickolay Y Gnedin, Nathan J Goldbaum, et al. The agora high-resolution galaxy simulations comparison project. *The Astrophysical Journal Supplement Series*, 210(1):14, 2013.

[91] Sinae Kim, Mahlet G Tadesse, and Marina Vannucci. Variable selection in clustering via dirichlet process mixture models. *Biometrika*, 93(4):877–893, 2006.

[92] Jack PC Kleijnen. Design and analysis of simulation experiments. international series in operations research and management science, 2008.

[93] Jack PC Kleijnen. Kriging metamodeling in simulation: A review. *European journal of operational research*, 192(3):707–716, 2009.

[94] Earl Lawrence, Katrin Heitmann, Martin White, David Higdon, Christian Wagner, Salman Habib, and Brian Williams. The coyote universe. iii. simulation suite and precision emulator for the nonlinear matter power spectrum. *The Astrophysical Journal*, 713(2):1322, 2010.

[95] H Lee, R Gramacy, Crystal Linkletter, and G Gray. Optimization subject to hidden constraints via statistical emulation. *Pacific Journal of Optimization*, 7(3):467–478, 2011.

[96] Phenyo E Lekone and Bärbel F Finkenstädt. Statistical inference in a stochastic epidemic seir model with control intervention: Ebola as a case study. *Biometrics*, 62(4):1170–1177, 2006.

[97] Tiancheng Li, Shudong Sun, Tariq Pervez Sattar, and Juan Manuel Corchado. Fight sample degeneracy and impoverishment in particle filters: A review of intelligent approaches. *Expert Systems with applications*, 41(8):3944–3954, 2014.

[98] Li Liang-Qun, Ji Hong-Bing, and Luo Jun-Hui. The iterated extended kalman particle filter. In *IEEE International Symposium on Communications and Information Technology, 2005. ISCIT 2005.*, volume 2, pages 1213–1216. IEEE, 2005.

[99] Jau-Hong Lin, Miao-Ju Hsu, Hsin-Wen Hsu, Hung-Chia Wu, and Ching-Lin Hsieh. Psychometric comparisons of 3 functional ambulation measures for patients with stroke. *Stroke*, 41 (9):2021–2025, 2010.

[100] Lizhen Lin and David B Dunson. Bayesian monotone regression using gaussian process projection. *Biometrika*, 101(2):303–317, 2014.

[101] Erik Lindström, Jonas Ströjby, Mats Brodén, Magnus Wiktorsson, and Jan Holst. Sequential calibration of options. *Computational Statistics & Data Analysis*, 52(6):2877–2891, 2008.

[102] Steven N MacEachern. Dependent dirichlet processes. *Unpublished manuscript, Department of Statistics, The Ohio State University*, pages 1–40, 2000.

[103] Amandine Marrel, Bertrand Iooss, Sébastien Da Veiga, and Mathieu Ribatet. Global sensitivity analysis of stochastic computer models with joint metamodels. *Statistics and Computing*, 22(3):833–847, 2012.

[104] Peter McCullagh and John A Nelder. *Generalized linear models*, volume 37. CRC press, 1989.

[105] Geoffrey J McLachlan, Sharon X Lee, and Suren I Rathnayake. Finite mixture models. *Annual review of statistics and its application*, 6:355–378, 2019.

[106] N. Metropolis, A. Rosenbluth, M. Rosenbluth, A. Teller, and E. Teller. Equations of state calculations by fast computing machines. *Journal of Chemical Physics*, 21:1087–1091, 1953.

[107] Nicholas Metropolis, Arianna W Rosenbluth, Marshall N Rosenbluth, Augusta H Teller, and Edward Teller. Equation of state calculations by fast computing machines. *The journal of chemical physics*, 21(6):1087–1092, 1953.

[108] Matthias Morzfeld, Marcus S Day, Ray W Grout, George Shu Heng Pau, Stefan A Finsterle, and John B Bell. Iterative importance sampling algorithms for parameter estimation. *SIAM Journal on Scientific Computing*, 40(2):B329–B352, 2018.

[109] Peter Müller, Fernando Quintana, and Gary Rosner. A method for combining inference across related nonparametric bayesian models. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 66(3):735–749, 2004.

[110] Radford M Neal. Annealed importance sampling. *Statistics and computing*, 11(2):125–139, 2001.

[111] Elaine O Nsoesie, John S Brownstein, Naren Ramakrishnan, and Madhav V Marathe. A systematic review of studies on forecasting the dynamics of influenza outbreaks. *Influenza and other respiratory viruses*, 8(3):309–316, 2014.

[112] J. Oakley and A. O'Hagan. Probabilistic sensitivity analysis of complex models. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 66:751–769, 2004.

[113] Jeremy E Oakley and Anthony O'Hagan. Probabilistic sensitivity analysis of complex models: a bayesian approach. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 66(3):751–769, 2004.

[114] Jeremy E Oakley and Benjamin D Youngman. Calibration of stochastic computer simulators using likelihood emulation. *Technometrics*, 59(1):80–92, 2017.

[115] Anthony O'Hagan and John Frank Charles Kingman. Curve fitting and optimal design for prediction. *Journal of the Royal Statistical Society. Series B (Methodological)*, pages 1–42, 1978.

[116] Kei Owada, Tim Eckmanns, Kande-Bure O'Bai Kamara, and Olushayo Oluseun Olu. Epidemiological data management during an outbreak of ebola virus disease: key issues and observations from sierra leone. *Frontiers in public health*, 4, 2016.

[117] Richard G Palmer, W Brian Arthur, John Henry Holland, Blake LeBaron, and Paul Tayler. Artificial economic life: a simple model of a stockmarket. *Physica D: Nonlinear Phenomena*, 75:264–274, 1994.

[118] Rui Paulo, Gonzalo García-Donato, and Jesús Palomo. Calibration of computer models with multivariate output. *Computational Statistics & Data Analysis*, 56(12):3959–3974, 2012.

[119] Sen Pei, Sasikiran Kandula, Wan Yang, and Jeffrey Shaman. Forecasting the spatial transmission of influenza in the united states. *Proceedings of the National Academy of Sciences*, 115(11):2752–2757, 2018.

[120] Matthew Plumlee. Bayesian calibration of inexact computer models. *Journal of the American Statistical Association*, 112(519):1274–1285, 2017. doi: 10.1080/01621459.2016.1211016. URL https://doi.org/10.1080/01621459.2016.1211016.

[121] Matthew Plumlee and Rui Tuo. Building accurate emulators for stochastic simulations via quantile kriging. *Technometrics*, 56(4):466–473, 2014.

[122] R Core Team. *R: A Language and Environment for Statistical Computing*. R Foundation for Statistical Computing, Vienna, Austria, 2018. URL https://www.R-project.org/.

[123] Carl E Rasmussen and Zoubin Ghahramani. Infinite mixtures of gaussian process experts. In *Advances in neural information processing systems*, pages 881–888, 2002.

[124] Carl Edward Rasmussen. Gaussian processes in machine learning. In *Advanced lectures on machine learning*, pages 63–71. Springer, 2004.

[125] Shubhankar Ray and Bani Mallick. Functional clustering by bayesian wavelet methods. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 68(2):305–332, 2006.

[126] Brian J Reich, Eric Kalendra, Curtis B Storlie, Howard D Bondell, and Montserrat Fuentes. Variable selection for high dimensional bayesian density estimation: application to human

exposure simulation. *Journal of the Royal Statistical Society: Series C (Applied Statistics)*, 61(1):47–66, 2012.

[127] Douglas Reynolds. *Gaussian Mixture Models*, pages 659–663. Springer US, Boston, MA, 2009. ISBN 978-0-387-73003-5. doi: 10.1007/978-0-387-73003-5_196. URL https://doi.org/10.1007/978-0-387-73003-5_196.

[128] Luis EC Rocha, Fredrik Liljeros, and Petter Holme. Simulated epidemics in an empirical spatiotemporal network of 50,185 sexual contacts. *PLoS computational biology*, 7(3):e1001109, 2011.

[129] Kathryn Roeder and Larry Wasserman. Practical bayesian density estimation using mixtures of normals. *Journal of the American Statistical Association*, 92(439):894–902, 1997.

[130] Reuven Y. Rubinstein. Optimization of computer simulation models with rare events. *European Journal of Operational Research*, 99(1):89 – 112, 1997. ISSN 0377-2217. doi: https://doi.org/10.1016/S0377-2217(96)00385-2. URL http://www.sciencedirect.com/science/article/pii/S0377221796003852.

[131] Jerome Sacks, William J Welch, Toby J Mitchell, and Henry P Wynn. Design and analysis of computer experiments. *Statistical science*, pages 409–423, 1989.

[132] Marcel Salathé, Maria Kazandjieva, Jung Woo Lee, Philip Levis, Marcus W Feldman, and James H Jones. A high-resolution human contact network for infectious disease transmission. *Proceedings of the National Academy of Sciences*, 107(51):22020–22025, 2010.

[133] H. M. Sauro, D. Harel, M. Kwiatkowska, C. A. Shaffer, A. M. Uhrmacher, M. Hucka, P. Mendes, L. Stromback, and J. J. Tyson. Challenges for modeling and simulation methods in systems biology. In *Proceedings of the 2006 Winter Simulation Conference*, pages 1720–1730, Dec 2006. doi: 10.1109/WSC.2006.322948.

[134] Thomas B Schön, Fredrik Lindsten, Johan Dahlin, Johan Wågberg, Christian A Naesseth, Andreas Svensson, and Liang Dai. Sequential monte carlo methods for system identification. *IFAC-PapersOnLine*, 48(28):775–786, 2015.

[135] Luca Scrucca, Michael Fop, T Brendan Murphy, and Adrian E Raftery. mclust 5: clustering, classification and density estimation using gaussian finite mixture models. *The R journal*, 8 (1):289, 2016.

[136] Robert E Serfling. Methods for current statistical analysis of excess pneumonia-influenza deaths. *Public health reports*, 78(6):494, 1963.

[137] Jayaram Sethuraman. A constructive definition of dirichlet priors. *Statistica sinica*, pages 639–650, 1994.

[138] Constantinos I Siettos and Lucia Russo. Mathematical modeling of infectious disease dynamics. *Virulence*, 4(4):295–306, 2013.

[139] WHO Ebola Response Team. Ebola virus disease in west africa—the first 9 months of the epidemic and forward projections. *New England Journal of Medicine*, 371(16):1481–1495, 2014.

[140] Leigh Tesfatsion. Agent-based computational economics: Growing economies from the bottom up. *Artificial life*, 8(1):55–82, 2002.

[141] Luke Tierney. Markov chains for exploring posterior distributions. *the Annals of Statistics*, pages 1701–1728, 1994.

[142] Ingo J. Timm and Fabian Lorig. A survey on methodological aspects of computer simulation as research technique. In *Proceedings of the 2015 Winter Simulation Conference*, WSC '15, pages 2704–2715, Piscataway, NJ, USA, 2015. IEEE Press. ISBN 978-1-4673-9741-4. URL http://dl.acm.org/citation.cfm?id=2888619.2888930.

[143] Michele Tizzoni, Kaiyuan Sun, Diego Benusiglio, Márton Karsai, and Nicola Perra. The scaling of human contacts and epidemic processes in metapopulation networks. *Scientific reports*, 5:15111, 2015.

[144] Ashleigh R Tuite, Joseph Tien, Marisa Eisenberg, David JD Earn, Junling Ma, and David N Fisman. Cholera epidemic in haiti, 2010: using a transmission model to explain spatial spread of disease and identify optimal control interventions. *Annals of internal medicine*, 154(9): 593–601, 2011.

[145] Rui Tuo and CF Jeff Wu. A theoretical framework for calibration in computer models: parametrization, estimation and convergence properties. *SIAM/ASA Journal on Uncertainty Quantification*, 4(1):767–795, 2016.

[146] Rui Tuo, CF Jeff Wu, et al. Efficient calibration for imperfect computer models. *The Annals of Statistics*, 43(6):2331–2352, 2015.

[147] Steffen Unkel, C Farrington, Paul H Garthwaite, Chris Robertson, and Nick Andrews. Statistical methods for the prospective detection of infectious disease outbreaks: a review. *Journal of the Royal Statistical Society: Series A (Statistics in Society)*, 175(1):49–82, 2012.

[148] S Venkatramanan, J Chen, A Fadikar, K Nwosu, S Gupta, Y Ren, B Lewis, M Marathe, H Mortveit, and A Vullikanti. Spatio-temporal optimization of seasonal vaccination and stockpile allocation using a metapopulation model of influenza. *BI Technical Report*, 1039, 2017.

[149] Srinivasan Venkatramanan, Bryan Lewis, Jiangzhuo Chen, Dave Higdon, Anil Vullikanti, and Madhav Marathe. Using data-driven agent-based models for forecasting emerging infectious diseases. *Epidemics*, 22:43–49, 2018.

[150] Ian Vernon, Michael Goldstein, Richard G Bower, et al. Galaxy formation: a bayesian uncertainty analysis. *Bayesian Analysis*, 5(4):619–669, 2010.

[151] Hans von Storch and Francis W. Zwiers. *Statistical Analysis in Climate Research*. Cambridge University Press, New York, 1999.

[152] Grace Wahba. *Spline models for observational data*, volume 59. Siam, 1990.

[153] Xiaojing Wang and James O Berger. Estimating shape constrained functions using gaussian processes. *SIAM/ASA Journal on Uncertainty Quantification*, 4(1):1–25, 2016.

[154] William J Welch, Robert J Buck, Jerome Sacks, Henry P Wynn, Toby J Mitchell, and Max D Morris. Screening, predicting, and computer experiments. *Technometrics*, 34(1):15–25, 1992.

[155] Amy Wesolowski, Nathan Eagle, Andrew J Tatem, David L Smith, Abdisalan M Noor, Robert W Snow, and Caroline O Buckee. Quantifying the impact of human mobility on malaria. *Science*, 338(6104):267–270, 2012.

[156] WHO Ebola Response Team. Ebola virus disease in west africa—the first 9 months of the epidemic and forward projections. *The New England journal of medicine*, 371(16):1481, 2014.

[157] Eric Winsberg. Computer simulations in science. In Edward N. Zalta, editor, *The Stanford Encyclopedia of Philosophy*. Metaphysics Research Lab, Stanford University, summer 2018 edition, 2018.

[158] Raymond KW Wong, Curtis B Storlie, and Thomas CM Lee. A frequentist approach to computer model calibration. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 79(2):635–648, 2017.

[159] Kenny Ye, William Li, and Agus Sudjianto. Algorithmic construction of optimal symmetric latin hypercube designs. *Journal of statistical planning and inference*, 90(1):145–159, 2000.

[160] Jun Yuan, Szu Hui Ng, and Kwok Leung Tsui. Calibration of stochastic computer models using stochastic approximation methods. *IEEE Transactions on Automation Science and Engineering*, 10(1):171–186, 2013.