Sequential Bayesian Design for Simulator Models

Steven Kleinegesse

Supervisor: Dr. M. U. Gutmann

School of Informatics
University of Edinburgh

This dissertation is submitted for the degree of
Masters of Science by Research

August 2018

Declaration

I hereby declare that except where specific reference is made to the work of others, the contents of this dissertation are original and have not been submitted in whole or in part for consideration for any other degree or qualification in this, or any other university. This dissertation is my own work and contains nothing which is the outcome of work done in collaboration with others, except as specified in the text and Acknowledgements.

Steven Kleinegesse
August 2018

Acknowledgements

This work was supported in part by the EPSRC Centre for Doctoral Training in Data Science, funded by the UK Engineering and Physical Sciences Research Council (grant EP/L016427/1) and the University of Edinburgh. I would like to thank my fellow CDT students for their helpful discussions and insights, as well as being great friends. I am grateful to my supervisor Dr Michael Gutmann who has been supportive and dedicated ever since he took me on as a student.

Abstract

Bayesian experimental design involves the optimal allocation of resources in an experiment, with the aim of optimising cost and performance. Design strategies can be roughly classified into static and sequential; the latter considers the update of our beliefs through real-world observations during the design process, whereas the former does not. Simulator-based statistical models define a class of models where the data generating distribution is intractable, which means that we cannot easily obtain a posterior or marginal distribution, but we can still sample data from it. There is little work on the combination of sequential Bayesian experimental design and simulator models, mainly due to the technical difficulties associated with approximating posterior distributions and utility functions, even though simulator models are highly common in the natural sciences. We address this research gap in this work by devising a novel sequential design framework. We use Likelihood-Free Inference by Ratio Estimation (LFIRE) for approximating posterior distributions and therewith optimise a decision function with grid search. Afterwards, we extend this framework by using Bayesian Optimisation instead of grid search, resulting in faster convergence. Finally, we assess our framework on two epidemiological models and, on one of them, test a non-myopic design setting, where we aim to find several optimal design points instead of just a single one. We find our novel framework to be efficient for the models tested, yielding accurate parameter estimates after only a few sequential iterations.

Table of Contents

List of Figures

List of Algorithms

Chapter 1

Introduction

1.1 Experimental Design

Experimental design is concerned with the allocation of resources when conducting an experiment, where we usually do not have control over all relevant variables. The general aim is to find design features, or experimental configurations, that may yield lower uncertainties, improve parameter estimations or compare competing models. Experimental design inherently has various applications in many scientific fields, ranging from natural sciences to social sciences, engineering and finance. In essence, the underlying question in experimental design is: where and how do we have to take data next in order to optimise cost and performance? The exact answer to this depends on the problem in question and what kind of decision strategy is used.

Finding an efficient experimental design is often viewed as an iterative optimisation problem, as different designs yield different experimental costs or performances. Traditional experimental design has been well-developed (Atkinson and Donev, 1992) and optimal designs are typically chosen based on the Fisher information matrix (e.g. Atkinson and Donev, 1992; Fedorov, 1972). To give a more conrete example, if we wanted to estimate Earth's gravitational acceleration at sea level, we could drop an object from different heights and then measure the time taken to hit the ground. Experimental design would then, for instance, involve finding out from which heights the object should be dropped or what the most optimal object shape is. We could choose to drop the object from random heights which, traditionally, is a commonly used strategy. The experiment cost would however, be higher and the resulting information gain would perhaps be lower than if we found the optimal heights to drop the object from.

While the traditional, frequentist framework works well for linear or linearisable problems, it is not suitable for non-linear models. For non-linear situations, only locally optimal experimental designs can be obtained, as the designs themselves become dependent on the values chosen for the model parameters (Ryan et al., 2016b). Bayesian statistics has some well-established, relatively old theory addressing this issue. Because of computational costs however, it has only recently gained popularity in the literature for experimental design. By updating our current knowledge about model parameters using Bayesian Statistics, we can find optimal experimental designs for non-linear models, a research field called *Bayesian experimental design* (see Chapter 2).

Much of the current literature pertains to *static* Bayesian experimental design which does not consider any real-world observations. *Sequential* Bayesian experimental design on the other hand includes updates of the current beliefs using real-world data that was taken at the optimal design points. Intuitively, this comes closer to the original motivation behind experimental design. Thus, in this work we shall focus on sequential instead of static design.

1.2 Simulator-Based Statistical Models

Bayesian methodologies involve the update of a prior distribution $p(\boldsymbol{\theta})$ on the unknown model parameters $\boldsymbol{\theta}$ with a data generating distribution $p(\mathbf{y} \mid \boldsymbol{\theta})$, i.e. the contribution made by some observations $\mathbf{y}$, via *Bayes' Rule*:

$$p(\boldsymbol{\theta} \mid \mathbf{y}) = \frac{p(\mathbf{y} \mid \boldsymbol{\theta}) p(\boldsymbol{\theta})}{p(\mathbf{y})} \tag{1.1}$$

In reality however, the data generating distribution is often too costly to compute or even intractable; this is the case for so-called *simulator-based statistical models*, or simulator models. They define a class of models where it is possible to stochastically sample from the likelihood if the parameters are fixed but it is impossible to numerically or analytically evaluate the data generating distribution or likelihood.

Simulator models are common in the natural sciences and are therefore useful in many disciplines. Some example uses of simulator models may include: the 'Ricker Model' in ecology (Ricker, 1954), bacterial infections in day care centers (Numminen et al., 2013), cosmological simulations (Alsing et al., 2018; M. Schafer and Freeman, 2012) or modelling particle collisions (Agostinelli et al., 2003; Sjöstrand et al., 2008). Thus, it is crucial to have an efficient framework to learn from data for these types of models.

Naturally, if the analytical form of the likelihood is unknown, Bayesian inference according to Equation 1.1 becomes difficult. *Likelihood-free inference* methods have emerged in order to make Bayesian inference possible for simulator models. They are a class of stochastic methods to numerically approximate the likelihood distribution and therefore the posterior distribution as well (see Chapter 2).

1.3 Motivation and Outline

The motivation behind this work stems from the end-goal of performing experimental design in situations where we have a simulator model. In general, these models are non-linear and therefore we have to use Bayesian experimental design rather than the traditional, frequentist framework. In the Bayesian formulation, decision functions usually depend on the posterior distribution. Because this is difficult to evaluate for simulator models, we also have to utilise likelihood-free inference methods to approximate the posterior distribution. There is however, little literature about the combination of Bayesian experimental design and likelihood-free inference methods. The aim of this work henceforth is thus to provide an efficient and novel framework for the combination of these research fields and validate it through a number of real-life examples.

We identify three gaps that we aim to address in this work. First of is the construction of a general framework for the combination of sequential Bayesian experimental design and likelihoof-free inference. Afterwards, we want to find an efficient method of optimising the decision function used in the framework. Lastly, we also wish to test how well the devised framework works for *non-myopic design*, i.e. situations where we aim to select a set of next optimal design points instead of just a single one.

The theoretical background of Bayesian experimental design and likelihood-free inference, in particular by using the method of *Ratio Estimation* (Dutta et al., 2016), is given in Chapter 2. This is followed by a description of relevant related work pertaining to Bayesian experimental design for simulator models in Chapter 3, as well as the statement of our research goals. Chapter 4 then goes into detail about how to actually combine these two ideas theoretically, as well as how to efficiently implement this. We explain how to improve the efficiency of our algorithm by introducing Bayesian Optimisation into our framework in Chapter 5; we shall briefly test how this framework performs for non-myopic design. The proposed framework and algorithms are illustrated and tested on a simple toy model and two popular models from epidemiology. Finally, Section 6 summarises the outcomes of this investigation and discusses possible future work.

Chapter 2

Background

2.1 Toy Model: Fitting a straight line

A running toy model is oftentimes useful in illustrating various techniques more intuitively. Moreover, if we select a toy model that takes the form of a straight line, we can compute the posterior distribution analytically by assuming that all involved distributions are Gaussian. This will allow us to compare most techniques discussed throughout this work to an analytical solution.

Instead of discussing the textbook-task of just fitting a straight line to some random data, we are going to consider a more real-world formulation: consider the fall of atmospheric pressure P as a function of height z above sea level. Under the assumptions of hydrostatic equilibrium, one can obtain the *barometric formula* (Braeunig, 2014)

$$P = P_0 \exp\left(-\frac{z}{H}\right) \qquad \Longrightarrow \qquad \log P = \log P_0 - \frac{z}{H} \tag{2.1}$$

This formula essentially says that, in log-space, the atmospheric pressure falls off linearly with altitude. The atmospheric pressure P_0 at sea level and the so-called scale height H are physical constants. We shall assume true values of 101325 Pa and 8.5 km for the sea level pressure and scale height, respectively (Williams, 2017). For this toy model, let us define $y = \log P$, $\theta_0 = \log P_0$ and $\theta_1 = -1/H$, such that Equation 2.1 becomes

$$y = \theta_0 + \theta_1 z \tag{2.2}$$

Using the aforementioned values for the physical constants, the true parameter values in this linear model are thus $\theta_0^* = \log 101325 \approx 5.00$ and $\theta_1^* = -1/8.5 \approx -0.12$, up to two decimal places.

The relevant question that we would like to solve via Bayesian experimental design is then: at what altitudes z do we need to make measurements of $y = \log P$ such that our estimation of $\boldsymbol{\theta} = [\theta_0, \theta_1]^\top$ is as statistically accurate and computationally efficient as possible.

For simplicity, we shall assume that the prior distribution and likelihood are Gaussian, i.e. $p(\boldsymbol{\theta}) = \mathcal{N}(\boldsymbol{\theta}; \boldsymbol{\theta}_0, V_0)$ and $p(y \mid \mathbf{z}, \boldsymbol{\theta}) = \mathcal{N}(y; \boldsymbol{\theta}^\top \mathbf{z}, \sigma_y^2)$, which corresponds to assuming a Gaussian measurement error on y. Here we have defined $\mathbf{z} = [1, z]^\top$ and we assume that we know the measurement noise σ_y^2, as well as the prior hyper parameters $\boldsymbol{\theta}_0$ and V_0. The resulting posterior distribution is then also a Gaussian (Murphy, 2012):

$$p(\boldsymbol{\theta} \mid \mathbf{y}, \mathbf{z}) = \mathcal{N}(\boldsymbol{\theta}; \boldsymbol{\theta}_N, V_N), \quad \text{where} \tag{2.3}$$

$$V_N = \sigma_y^2 (\sigma_y^2 V_0^{-1} + Z^\top Z)^{-1} \tag{2.4}$$

$$\boldsymbol{\theta}_N = V_N V_0^{-1} \boldsymbol{\theta}_0 + \frac{1}{\sigma_y^2} V_N Z^\top \mathbf{y}, \tag{2.5}$$

where Z is a matrix whose rows consist of design points $\mathbf{z}$ and $\mathbf{y}$ is a vector containing corresponding observations. Having an analytical expression for the posterior distribution over a set of altitudes allows us to analytically solve several of the problems mentioned hereafter.

2.2 Bayesian Experimental Design

Because frequentist experimental design is not particularly suitable for non-linear problems (Ryan et al., 2016b), Bayesian methodologies have recently become popular for finding optimal designs. Due to the Bayesian formulation, an observation at a single design point is enough to update the prior and form the posterior distribution; this is generally a significant advantage because observations are assumed to be expensive in experimental design problems.

At its core, Bayesian experimental design involves defining a utility function $U(\mathbf{d}, \boldsymbol{\theta}, \mathbf{y})$ that describes the value of design $\mathbf{d}$, given future data $\mathbf{y}$ and model parameters $\boldsymbol{\theta}$. In order to qualify as a 'fully Bayesian design', this utility function has to be a functional of the posterior distribution $p(\boldsymbol{\theta} \mid \mathbf{d}, \mathbf{y})$ (Ryan et al., 2016b). The Bayesian optimal design $\mathbf{d}^*$ that we are trying to find is the design point that maximises the expected utility function $U(\mathbf{d})$ over the

whole design space,

$$\mathbf{d}^* = \arg\max_{\mathbf{d}} U(\mathbf{d}). \tag{2.6}$$

The expected utility $U(\mathbf{d})$ in Equation 2.6 is calculated with respect to the joint distribution of data $\mathbf{y}$ and parameters $\boldsymbol{\theta}$, given design point $\mathbf{d}$:

$$\begin{aligned} U(\mathbf{d}) &= \mathbb{E}_{p(\boldsymbol{\theta},\mathbf{y}|\mathbf{d})}[U(\mathbf{d},\boldsymbol{\theta},\mathbf{y})] && (2.7)\\ &= \int\int U(\mathbf{d},\boldsymbol{\theta},\mathbf{y}) p(\boldsymbol{\theta},\mathbf{y} \mid \mathbf{d}) \mathrm{d}\boldsymbol{\theta}\mathrm{d}\mathbf{y} && (2.8)\\ &= \int\int U(\mathbf{d},\boldsymbol{\theta},\mathbf{y}) p(\mathbf{y} \mid \boldsymbol{\theta},\mathbf{d}) p(\boldsymbol{\theta}) \mathrm{d}\boldsymbol{\theta}\mathrm{d}\mathbf{y}, && (2.9) \end{aligned}$$

where we have assumed that $p(\boldsymbol{\theta} \mid \mathbf{d}) = p(\boldsymbol{\theta})$, i.e. the prior distribution is independent of the design.

The double integral in Equation 2.9 involves the utility function $U(\mathbf{d},\boldsymbol{\theta},\mathbf{y})$ and the joint distribution between the data and model parameters for design $\mathbf{d}$. Unless the prior $p(\boldsymbol{\theta})$ and data generating distribution $p(\mathbf{y} \mid \boldsymbol{\theta},\mathbf{d})$ are specifically chosen to allow for analytic calculation of this double integral, the expected utility, and hence the optimisation problem in Equation 2.6, does not have a closed-form solution. This means that we generally have to use numerical approximations to solve the above.

Because of the computational difficulties associated with the aforementioned maximisation and integration tasks, Bayesian experimental design methods have often been avoided. Besides having to compute a double integral over multi-dimensional variable spaces, we must also first evaluate the posterior distribution which is already difficult. Thus, in order to find an optimal design via Bayesian methodologies, we have to find computationally efficient, yet statistically accurate, ways of solving these maximisation and integration problems.

The optimal design $\mathbf{d}^*$ obtained through the optimisation problem posed in Equation 2.6 highly depends on the choice of utility function $U(\mathbf{d},\boldsymbol{\theta},\mathbf{y})$. The utility function reflects the aims of the experiment, e.g. some utilities are geared towards future predictions, while others are for parameter estimation or model selection. In this work, we are mostly interested in utility functions for parameter estimation; furthermore, we consider information-based utilities, as opposed to e.g. quadratic loss. Finding out which utilities are best in certain situations (see Ryan et al., 2016b) is a whole different research field and therefore we shall not go into detail here. A commonly-used, versatile information-based expected utility is the *mutual information*, which has been used for efficient parameters estimation, reducing prediction uncertainty and model discrimination (Ryan et al., 2016b).

For information-based utilities, we are essentially concerned about what design $\mathbf{d}$ would yield the most information gain about $\boldsymbol{\theta}$, given some future data $\mathbf{y}$. An efficient way of estimating this information gain may be via the Kullback-Leibler (KL) Divergence D_{KL} (Kullback and Leibler, 1951) between the posterior and the prior,

$$D_{KL}(p(\boldsymbol{\theta} \mid \mathbf{d},\mathbf{y}) \mid\mid p(\boldsymbol{\theta})) = \int p(\boldsymbol{\theta} \mid \mathbf{d},\mathbf{y}) \log\left(\frac{p(\boldsymbol{\theta} \mid \mathbf{d},\mathbf{y})}{p(\boldsymbol{\theta})}\right) \mathrm{d}\boldsymbol{\theta} \tag{2.10}$$

The KL-Divergence is a distance-measure, although mathematically not a proper distance, and in this case tells us how 'close' our newly-inferred posterior distribution is to the prior distribution. The KL-Divergence depends on the data $\mathbf{y}$; in order to obtain the mutual information, we then take an average over the data marginal, i.e.

$$\begin{aligned} U(\mathbf{d}) &= \mathbb{E}_{p(\mathbf{y}|\mathbf{d})}[D_{KL}(p(\boldsymbol{\theta} \mid \mathbf{d},\mathbf{y}) \mid\mid p(\boldsymbol{\theta}))] && (2.11)\\ &= \int\int \log\left(\frac{p(\boldsymbol{\theta} \mid \mathbf{d},\mathbf{y})}{p(\boldsymbol{\theta})}\right) p(\boldsymbol{\theta} \mid \mathbf{y},\mathbf{d}) p(\mathbf{y} \mid \mathbf{d}) \mathrm{d}\boldsymbol{\theta}\mathrm{d}\mathbf{y} && (2.12)\\ &= \int\int \log\left(\frac{p(\boldsymbol{\theta} \mid \mathbf{d},\mathbf{y})}{p(\boldsymbol{\theta})}\right) p(\mathbf{y} \mid \boldsymbol{\theta},\mathbf{d}) p(\boldsymbol{\theta}) \mathrm{d}\boldsymbol{\theta}\mathrm{d}\mathbf{y} && (2.13) \end{aligned}$$

Another commonly-used information-based utility is the 'Bayesian D-Posterior precision', which is based on the determinant of the covariance matrix of the posterior distribution (Ryan et al., 2016b). This utility works well for uni-modal distribution but breaks down for multi-modal distributions. Because we do not want make too many assumptions and preserve generality, we shall chose to use the more theoretically grounded mutual information as the expected utility.

In order to compute $U(\mathbf{d})$ in Equation 2.13 however, we would need to evaluate two integrals: one over the data space and one over the whole parameter space. Because these are generally high-dimensional spaces and the integrands may be complicated functions, evaluating these integrals exactly is not a straightforward task. Using a technique called *Monte-Carlo Integration*, we can approximate the expected utility as

$$\begin{aligned} U(\mathbf{d}) &= \int\int \log\left(\frac{p(\boldsymbol{\theta} \mid \mathbf{d},\mathbf{y})}{p(\boldsymbol{\theta})}\right) p(\mathbf{y} \mid \boldsymbol{\theta},\mathbf{d}) p(\boldsymbol{\theta}) \mathrm{d}\boldsymbol{\theta}\mathrm{d}\mathbf{y} && (2.14)\\ &\approx \frac{1}{N}\sum_{i=1}^{N} \log\left(\frac{p(\boldsymbol{\theta}^{(i)} \mid \mathbf{d},\mathbf{y}^{(i)})}{p(\boldsymbol{\theta}^{(i)})}\right), && (2.15) \end{aligned}$$

where $\mathbf{y}^{(i)} \sim p(\mathbf{y} \mid \mathbf{d},\boldsymbol{\theta}^{(i)})$ and $\boldsymbol{\theta}^{(i)} \sim p(\boldsymbol{\theta})$. In this form, the Monte-Carlo approximation requires us to sample from the prior distribution and from the data generating distribution, as

well as analytically evaluate their densities. As previously said in Chapter 1, for simulator-based statistical models we do not have analytical forms of the data generating distributions and hence neither of the posterior distribution; thus, the above Monte-Carlo approximation would not be applicable. In Section 2.3 we discuss how we address this issue via likelihood-free inference methods.

So far, all the aforementioned equations were *static*, i.e. there were no updates and real data observations. A more intuitive way of thinking about Bayesian experimental design however, would be via *sequential* design. In this case, a real observation is taken at the design point $\mathbf{d}^*$ that maximises the expected utility. This observed data point is then used to update the prior distribution, following which the expected utility at each design point changes and a new optimal design can be computed. This procedure is then repeated several times to obtain a set of sequentially designed experiments. This framework is closer to reality, where an experimenter has to firmly think about the experimental setup that is used to collect data next. Conversely, because the computation of the expected utility takes time, this is only a valid approach when there is sufficient time between the observations.

The observed data points $\mathbf{y}_t^*$ are assumed to be generated from the real-world data-generating process at the optimal design $\mathbf{d}_t^*$. The form of the expected utility for sequential designs does not change very much, except that the posterior distribution and prior distribution at iteration t depend on the previous observations $\mathbb{D}_{t-1} = \{\mathbf{d}_{1:t-1}^*, \mathbf{y}_{1:t-1}^*\}$. Analogous to Equation 2.13, the expected utility for sequential design is then

$$U_t(\mathbf{d}) = \int\int \log\left(\frac{p(\boldsymbol{\theta} \mid \mathbf{d}, \mathbf{y}, \mathbb{D}_{t-1})}{p(\boldsymbol{\theta} \mid \mathbb{D}_{t-1})}\right) p(\mathbf{y} \mid \boldsymbol{\theta}, \mathbf{d}) p(\boldsymbol{\theta} \mid \mathbb{D}_{t-1}) \mathrm{d}\boldsymbol{\theta}\mathrm{d}\mathbf{y} \tag{2.16}$$

$$\tag{2.17}$$

We here assume that the data $\mathbf{y}$ is independent of the previous data generating operations, i.e. $\mathbf{y} \perp\!\!\!\perp \mathbb{D}_{t-1}$. This may not always be the case in special situations where observing data actually changes the real-world data generating process.

The expected utility can again be approximated via Monte-Carlo integration,

$$\widehat{U}_t(\mathbf{d}) \approx \frac{1}{N} \sum_{i=1}^{N} \log\left(\frac{p(\boldsymbol{\theta}^{(i)} \mid \mathbf{d}, \mathbf{y}^{(i)}, \mathbb{D}_{t-1})}{p(\boldsymbol{\theta}^{(i)} \mid \mathbb{D}_{t-1})}\right) \tag{2.18}$$

Since the data generating distribution is invariant to the observed data, we have $\mathbf{y}^{(i)} \sim p(\mathbf{y} \mid \mathbf{d}, \boldsymbol{\theta}^{(i)})$, just like in Equation 2.15. The model parameter samples however, now come from the updated prior, the posterior of the previous iteration, i.e. $\boldsymbol{\theta}^{(i)} \sim p(\boldsymbol{\theta} \mid \mathbb{D}_{t-1})$. Thus,

Monte-Carlo integration in this form is only possible if we can sample from the posterior. Computing this expectation gets more complicated if that is not possible — we shall explain how to do this in Chapter 4.

The sequential Bayesian experimental design framework for discrete design space is summarised in Algorithm 1. In Section 2.4 we shall explain an alternative to computing the expected utilities on a design space grid.

Algorithm 1 Sequential Bayesian Exp. Design using Grid Search

Let $\mathbb{D}_0 = \varnothing$
for t=1 to t=T **do**
    for $\mathbf{d}$ in discrete design space $\mathscr{D}$ **do**
        Sample model parameters from prior: $\boldsymbol{\theta}^{(i)} \sim p(\boldsymbol{\theta} \mid \mathbb{D}_{t-1})$ for $i = 1, \ldots, N$
        Simulate data from the model: $\mathbf{y}^{(i)} \sim p(\mathbf{y} \mid \mathbf{d}, \boldsymbol{\theta}^{(i)})$ for $i = 1, \ldots, N$
        Compute the expected utility approximation $\widehat{U}_t(\mathbf{d})$ given in Equation 2.18
    end for
    Find the next optimal design point: $\mathbf{d}_t^* = \arg\min_{\mathbf{d}} \widehat{U}_t(\mathbf{d})$
    Perform an experiment to observe some data, i.e. $\mathbf{y}_t^* \sim p(\mathbf{y} \mid \mathbf{d}_t^*, \boldsymbol{\theta}_{\text{true}})$
    Update the prior distribution by updating the data set: $\mathbb{D}_t = \mathbb{D}_{t-1} \cup \{\mathbf{d}_t^*, \mathbf{y}_t^*\}$
end for

Let us now come back to the toy example outlined in Section 2.1. The aim of sequential Bayesian experimental design in context of this toy example is to find the next altitude z where we should take real-world observations of the pressure P. The log pressure $y = \log P$ varies linearly with altitude (see Equation 2.2) and the data generating distribution, prior distribution and posterior distribution are all Gaussians. The update to the posterior, given an altitude z and observations y, is given in Equations 2.3–2.5; we shall assume here that the measurement noise σ_y in the data is 1. Because all involved distributions are Gaussian, it is trivial to sample parameters from them.

Assuming that the true parameters are given by $\boldsymbol{\theta}_{true} = [5.00, -0.12]^\top$ (see Section 2.1), we can apply Algorithm 1 and sequentially find the optimal altitudes where we would have to take data next. We shall limit ourselves to the troposphere, i.e. altitudes up to 12 km, because above that the model starts to become non-linear (Braeunig, 2014). The design space is therefore discretised into 100 design points z between 0 km and 12 km. We also choose to sample 1000 model parameters from a zero-mean Gaussian with a standard deviation of 5 and thus simulate 1000 data points for each design point. The resulting expected utility functions $U_t(z)$, calculated according to Algorithm 1, for a total of four iterations (i.e. four real-world data observations), are shown in Figure 2.1.

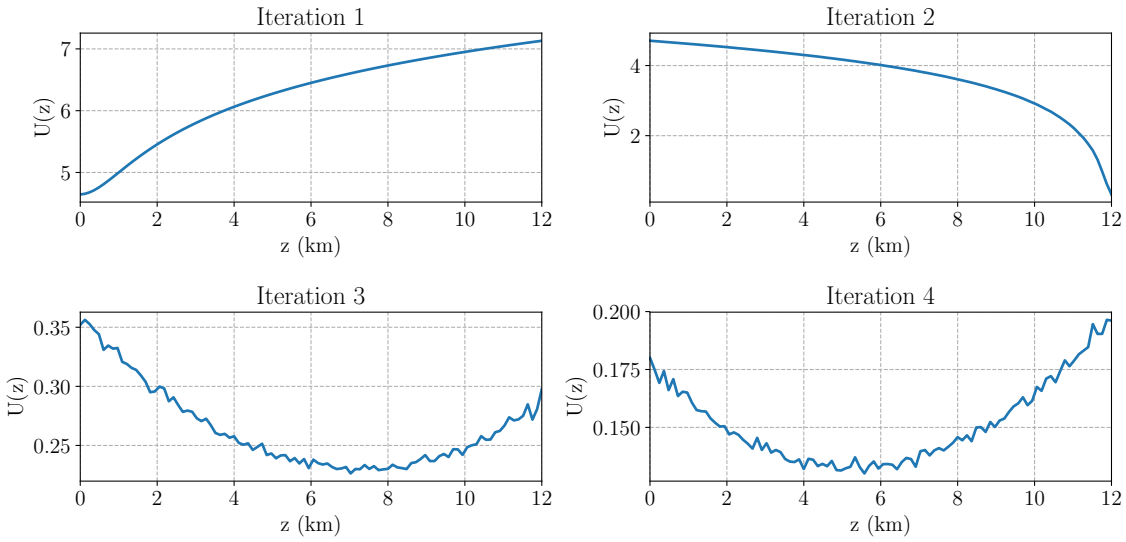

Fig. 2.1 Expected utility functions $U_t(z)$ for the linear toy example, for a total of four iterations, using the grid search method. For each design point, 1000 y values were simulated from the posterior to calculate the expectation; $\boldsymbol{\theta}_{true} = [5.00, -0.12]^\top$ was used to make the observations and update the prior distribution. Note the different ranges.

As can be seen from Figure 2.1, the expected utility function for the first iteration is low at $z = 0$ km and higher at $z = 12$ km. Presumably, this is because the relative influence of the noise $\sigma_y = 1$, i.e. the signal-to-noise ratio, in the observations and simulations is less at higher altitudes and higher y. Once an observation is then taken at $z = 12$ km and the prior distribution is updated, the next expected utility function is near zero at $z = 12$ km and high at $z = 0$ km, the other end of the design space. This makes sense, because taking an observation again at the exact same location as previously does not give us much information when we have only done one observation. However, after having taken the next observation at $z = 0$ km, the magnitude of the expected utility function quickly decreases (see Iteration 3 and 4 in Figure 2.1). This is intuitive, as two data points are mathematically already enough to fully describe a straight line. Most of the next optimal designs are all at the boundaries, i.e. at either $z = 0$ km or $z = 12$ km.

While the Bayesian experimental design procedure in Algorithm 1 is theoretically sound, it may not be totally clear how much of an advantage, if any at all, we are getting by designing our experiments. As opposed to using a grid to maximise the expected utility and selecting an optimal design point accordingly, we could also randomly select an optimal design point using a Uniform distribution. Using the linear toy model and for comparison purposes, we are considering the distance of the posterior mean at every iteration to the true model parameter

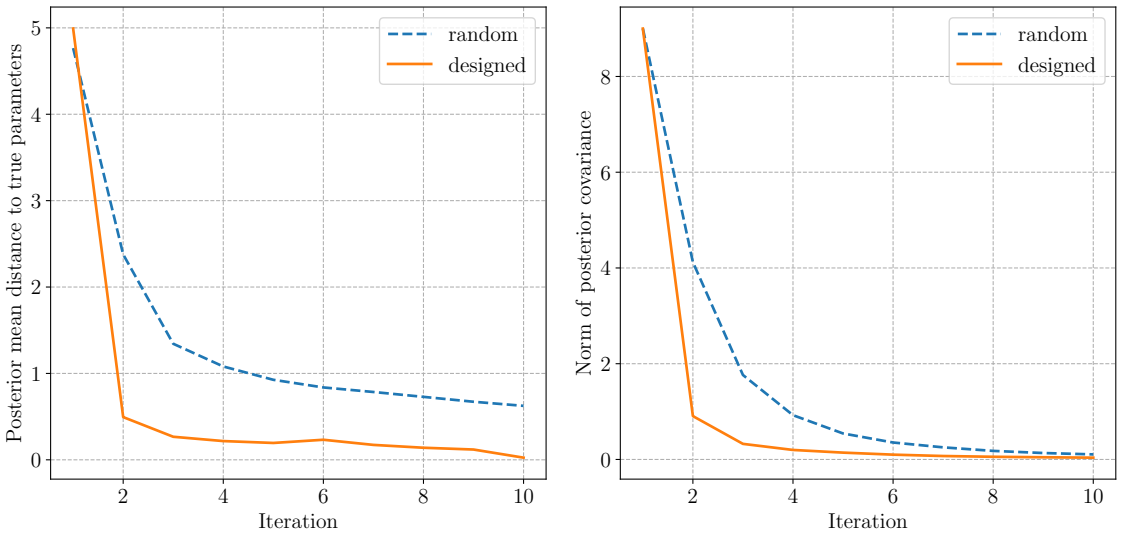

Fig. 2.2 Left: Distance of the posterior mean to the true model parameter values $\boldsymbol{\theta}_{true} = [5.00, -0.12]^\top$ for random (blue) and designed (orange) acquisitions, as a function of iteration for the toy model using grid search. Right: Norm of the posterior covariance for random (blue) and designed (orange) acquisitions, as a function of iteration for the toy model using grid search. Note that the quantities for the random experiments are the means of 100 repetitions.

values $\boldsymbol{\theta}_{true} = [5.00, -0.12]^\top$, as well as the norm of the posterior covariance. Having run the random experiment 100 timesw, we show the means of the aforementioned quantities in Figure 2.2, together with the quantities for a designed experiment.

While both methods perform similarly for the simple toy model, designing the experiment results in posterior means that are closer to the true value and posterior covariances that are narrower than for randomly chosen altitudes z^*. Sometimes the random acquisitions result in posterior distributions that are surprisingly close to that for designed acquisitions, due to optimal design points being selected closer to the bounds. We already see a difference between these methods for a simple example where randomly chosen designs are informative. It is reasonable to expect that the benefits increase for more complex scenarios.

2.3 Likelihood-Free Inference

As explained in Section 1.2, simulator-based models have intractable likelihoods and therefore traditional Bayesian inference becomes infeasible for them. In order to numerically approximate likelihoods and posterior distributions, we shall make use of the class of stochas-

tic methods called *likelihood-free inference* methods. A requirement for these techniques is that one needs to be able to sample from the likelihood function, given some parameter values, as is the case for simulator-based models.

Popular likelihood-free inference methods include Approximate Bayesian Computation (ABC) (Rubin, 1984) and Synthetic Likelihood (SL) (Wood, 2010). Part of the ABC class of methods is the prominent ABC rejection method that works by reducing observed and simulated data to summary statistics and measuring their distance; if this distance is small enough, the parameters that were used to simulate the data are accepted as samples from the posterior. Although this works well in theory, in practice making the approximation sufficiently accurate results in a massive increase of computational costs. The SL approach also requires computation of summary statistics from simulated data but assumes that the summary statistics follow a Gaussian distribution. While this method works well for a variety of problems, its Gaussianity assumption may not always hold.

A more recent framework proposed by Dutta et al. (2016), called Likelihood-Free Inference by Ratio Estimation (LFIRE), includes an automatic selection of summary statistic and does not assume any Gaussianity. The basic idea is to approximate the ratio of likelihood to marginal distribution by designing a classification problem between samples from the likelihood and samples from the marginal. Furthermore, Dutta et al. (2016) found their method to be equivalent to, or even better than, SL for the models tested. Thus, we decided that the main likelihood-free inference method that shall be used in this work is the LFIRE approach.

The basic idea of LFIRE is to evaluate the ratio $r(\mathbf{d}, \mathbf{y}, \boldsymbol{\theta})$ of the data generating distribution $p(\mathbf{y} \mid \mathbf{d}, \boldsymbol{\theta})$ and the marginal of the data $p(\mathbf{y} \mid \mathbf{d})$,

$$r(\mathbf{d}, \mathbf{y}, \boldsymbol{\theta}) = \frac{p(\mathbf{y} \mid \mathbf{d}, \boldsymbol{\theta})}{p(\mathbf{y} \mid \mathbf{d})}. \tag{2.19}$$

An estimate $\hat{r}(\mathbf{d}, \mathbf{y}, \boldsymbol{\theta})$ of the true ratio results in an estimate of the posterior, by Bayes' Rule:

$$\hat{p}(\boldsymbol{\theta} \mid \mathbf{d}, \mathbf{y}) = \hat{r}(\mathbf{d}, \mathbf{y}, \boldsymbol{\theta}) p(\boldsymbol{\theta}). \tag{2.20}$$

Dutta et al. (2016) approximate this ratio by framing the opimisation problem as a *Logistic Regression* problem. Because we are dealing with simulator-based statistical models, we can sample from the likelihood, or data generating distribution, $p(\mathbf{y} \mid \mathbf{d}, \boldsymbol{\theta})$ in the numerator of this ratio.

Let $\mathbf{Y}_{\boldsymbol{\theta}} = \{\mathbf{y}_{\boldsymbol{\theta}}^{(i)}\}_{i=1}^{n_{\boldsymbol{\theta}}}$ be a set of $n_{\boldsymbol{\theta}}$ independently-sampled data points from the data generating distribution, given a fixed value of $\boldsymbol{\theta}$. We can also generate a dataset $\mathbf{Y}_m = \{\mathbf{y}_m^{(i)}\}_{i=1}^{n_m}$ of n_m independently-sampled data points from the marginal $p(\mathbf{y} \mid \mathbf{d})$ in the denominator of the ratio. In order to sample from the marginal, one has to sample model parameters from the prior, i.e. $\boldsymbol{\theta}^{(i)} \sim p(\boldsymbol{\theta})$, substitute them into the data generating distribution and then sample from it, i.e. $\mathbf{y}_m^{(i)} \sim p(\mathbf{y} \mid \mathbf{d}, \boldsymbol{\theta}^{(i)})$.

The aim is then to determine whether some data $\mathbf{y}$ was generated from the data generating distribution $p(\mathbf{y} \mid \mathbf{d}, \boldsymbol{\theta})$ or the marginal $p(\mathbf{y} \mid \mathbf{d})$. In order to solve this classification problem via logistic regression, one can define the probability of $\mathbf{y}$ belonging to $\mathbf{Y}^{\boldsymbol{\theta}}$ as

$$\mathbb{P}(\mathbf{y} \in \mathbf{Y}_{\boldsymbol{\theta}}; h) = \frac{1}{1 + \nu \exp(-h(\mathbf{y}))}, \tag{2.21}$$

where $\nu = n_m / n_{\boldsymbol{\theta}}$ adjusts any class imbalance; in this work however, we shall consistently use $n_m = n_{\boldsymbol{\theta}}$ such that $\nu = 1$. The function h is unknown and has to be learned from the simulated data. Dutta et al. (2016) wrote this function as a linear combination of summary statistics $\boldsymbol{\psi}(\mathbf{y})$, i.e. $h(\mathbf{y}) = \boldsymbol{\beta}^\top \boldsymbol{\psi}(\mathbf{y})$, where the $\boldsymbol{\beta}$ is a set of coefficients that we would then have to learn. Note that this function could technically take different forms. The standard loss function $\mathcal{J}(\mathbf{d}, \boldsymbol{\beta}, \boldsymbol{\theta})$ that needs to be minimised for this classification problem is the negative binomial log-likelihood, i.e.

$$\mathcal{J}(\mathbf{d}, \boldsymbol{\beta}, \boldsymbol{\theta}) \propto \sum_{i=1}^{n_{\boldsymbol{\theta}}} \log[1 + \nu \exp(-\boldsymbol{\beta}^\top \boldsymbol{\psi}(\mathbf{y}_{\boldsymbol{\theta}}^{(i)})] + \sum_{i=1}^{n_m} \log[1 + \exp(\boldsymbol{\beta}^\top \boldsymbol{\psi}(\mathbf{y}_m^{(i)})/\nu] \tag{2.22}$$

The estimated coefficients for each model parameter $\boldsymbol{\theta}$ and design point $\mathbf{d}$ are then

$$\hat{\boldsymbol{\beta}}(\mathbf{d}, \boldsymbol{\theta}) = \arg\min_{\boldsymbol{\beta}} \mathcal{J}(\mathbf{d}, \boldsymbol{\beta}, \boldsymbol{\theta}). \tag{2.23}$$

It can be shown (Dutta et al., 2016) that in the limit of $n_{\boldsymbol{\theta}}$ and n_m going to infinity, the product of the minimising coefficients $\hat{\boldsymbol{\beta}}$ and the summary statistics $\boldsymbol{\psi}(\mathbf{y})$ is given by the log-ratio of the data generating distribution to the prior distribution, i.e. our desired ratio from Equation 2.19:

$$\hat{\boldsymbol{\beta}}(\mathbf{d}, \boldsymbol{\theta})^\top \boldsymbol{\psi}(\mathbf{y}) = \log\left(\frac{p(\mathbf{y} \mid \mathbf{d}, \boldsymbol{\theta})}{p(\mathbf{y} \mid \mathbf{d})}\right) \tag{2.24}$$

$$\text{so that} \quad \hat{r}(\mathbf{d}, \mathbf{y}, \boldsymbol{\theta}) = \exp\left(\hat{\boldsymbol{\beta}}(\mathbf{d}, \boldsymbol{\theta})^\top \boldsymbol{\psi}(\mathbf{y})\right) \tag{2.25}$$

This means that by solving the classification problem stated in Equation 2.23 for every $\boldsymbol{\theta}$ and $\mathbf{d}$, we are essentially estimating the ratio of likelihood to marginal and therefore the posterior $\hat{p}(\boldsymbol{\theta} \mid \mathbf{d}, \mathbf{y})$. Furthermore, the coefficients $\hat{\boldsymbol{\beta}}(\mathbf{d}, \boldsymbol{\theta})$ do not explicitly depend on any data $\mathbf{y}$, allowing us to quickly calculate the ratio $\hat{r}(\mathbf{d}, \mathbf{y}, \boldsymbol{\theta})$ for different data points, given a fixed $\boldsymbol{\theta}$ and $\mathbf{d}$.

Naturally, the choice of summary statistics $\psi(\mathbf{y})$ is a crucial one and essentially determines what we are learning through logistic regression. The summary statistics could be the means, standard deviations, etc. of the data or they could be some other kind of mapping. This generally depends on the specific problem and what kind of data one has at hand. In the SL approach, the underlying assumption is that the summary statistics are normally distributed but this does not have to be the case for the LFIRE approach.

Unlike for the ABC and SL approaches, we can allow for automatic selection of summary statistics by applying L1-regularisation to Equation 2.23. The minimisation problem then becomes

$$\hat{\boldsymbol{\beta}}_{reg}(\mathbf{d}, \boldsymbol{\theta}, \lambda) = \arg\min_{\boldsymbol{\beta}} \{ \mathcal{J}(\mathbf{d}, \boldsymbol{\beta}, \boldsymbol{\theta}) + \lambda \sum_i \mid \beta_i \mid \} \tag{2.26}$$

The idea is that sufficiently large values of the regularisation parameter λ put some of the coefficients exactly to zero. Therefore, if λ is chosen carefully, we can automatically select which of the summary statistics are important (in other words, if their coefficients are non-zero). The optimal λ can be chosen via a scoring method, such as prediction risk, and via cross-validation (see Dutta et al., 2016). The logistic regression, L1-regularisation and scoring methods have been wholly implemented via the GLMNET package in R.

The general procedure of the LFIRE approach, to estimate the posterior at design point $\mathbf{d}$ and model parameter $\boldsymbol{\theta}$, is summarised in Algorithm 2. Note that Lines 3 and 4 are executed by the GLMNET package in R.

Algorithm 2 Likelihood-Free Inference by Ratio Estimation

1: Simulate data from the marginal: $\mathbf{y}_m^{(i)} \sim p(\mathbf{y} \mid \mathbf{d})$ for $i = 1, \ldots, n_m$
2: Simulate data from the likelihood: $\mathbf{y}_{\boldsymbol{\theta}}^{(i)} \sim p(\mathbf{y} \mid \mathbf{d}, \boldsymbol{\theta})$ for $i = 1, \ldots, n_{\boldsymbol{\theta}}$
3: Estimate $\hat{\boldsymbol{\beta}}_{reg}(\mathbf{d}, \boldsymbol{\theta}, \lambda)$ by solving the optimisation problem posed in Equation 2.26 for a range of λ values
4: Find the λ_{min} that minimises the prediction risk (see Dutta et al., 2016) via ten-fold cross-validation and set $\hat{\boldsymbol{\beta}}(\mathbf{d}, \boldsymbol{\theta}) = \hat{\boldsymbol{\beta}}_{reg}(\mathbf{d}, \boldsymbol{\theta}, \lambda_{min})$
5: For data $\mathbf{y}$, compute the ratio estimate $\hat{r}(\mathbf{d}, \mathbf{y}, \boldsymbol{\theta})$ according to Equation 2.25

Let us illustrate the LFIRE method by coming back to the toy example outlined in Section 2.1. For this particular case, we have an analytical expression for the posterior

distribution, given in Equations 2.3–2.5. We would like to apply the LFIRE method to this problem and see how well we can approximate the true posterior distribution. The data generating distribution is Gaussian, i.e $p(y\,|\,\mathbf{z},\boldsymbol{\theta}) = \mathcal{N}(\mathbf{y};\boldsymbol{\theta}^\top\mathbf{z},\sigma_y^2)$, which makes it trivial to execute the sampling steps in Lines 1 and 2 of Algorithm 2. For this particular case, we use $n_m = n_{\boldsymbol{\theta}} = 1000$ and the summary statistics used were powers of altitudes up to an exponent of 4. We evaluate the estimated ratios at design point $z = 5$ km on a grid of parameter values, where $\theta_0 \in [-6,6]$ and $\theta_1 \in [-1,3]$, for a total of 400 parameter configurations. Firstly, the coefficients $\boldsymbol{\beta}(z=5,\theta)$ are estimated according to Lines 3 and 4 of Algorithm 2 for each θ in the parameter grid. Lastly, we make a real-world observation $\mathbf{y}^* \sim p(\mathbf{y} \mid z=5,\boldsymbol{\theta}_{true})$ and use this to estimate all ratios according to Line 5 of Algorithm 2. Finally, given a ratio estimate, the posterior density at each θ can be estimated via Equation 2.20 since we have an analytical expression of the prior distribution.

The posterior distribution approximated via the LFIRE method is shown in the contour plot in Figure 2.3, together with the analytical posterior distribution given the same real-world observation. Also shown are projections onto the θ_0 and θ_1 axis, to better visualise the accuracy of the approximation.

It becomes apparent that LFIRE approximates the shape of the posterior distribution well. Although the approximation does not match the analytical distribution perfectly in the peak region, the tails are captured accurately. If we were to increase the number of parameters used in the LFIRE approach then the approximation becomes increasingly close to the analytical posterior (Dutta et al., 2016). Unfortunately, this would also increase computation costs and we thus shall leave $n_m = n_{\boldsymbol{\theta}} = 1000$ throughout this work.

2.4 Bayesian Optimisation

In Section 2.2 we previously explained how to compute the expected utility $U(\mathbf{d})$. In order to solve the optimisation problem posed in Equation 2.6, we discretised the design space $\mathscr{D}$ and evaluated the expected utility on a grid of design points. This is manageable if the design space is one-dimensional, as is the case with our toy model. As we go up in dimensions and when evaluating the utility is expensive however, grid search quickly becomes unfeasible. We could optimise the utility by random search. While this is better than grid evaluations (Bergstra and Bengio, 2012), it is still expensive to guarantee a good coverage when the dimensions of the design space are high. Instead, we can use the popular method of *Bayesian Optimisation* (Shahriari et al., 2016) to optimise the expected utility

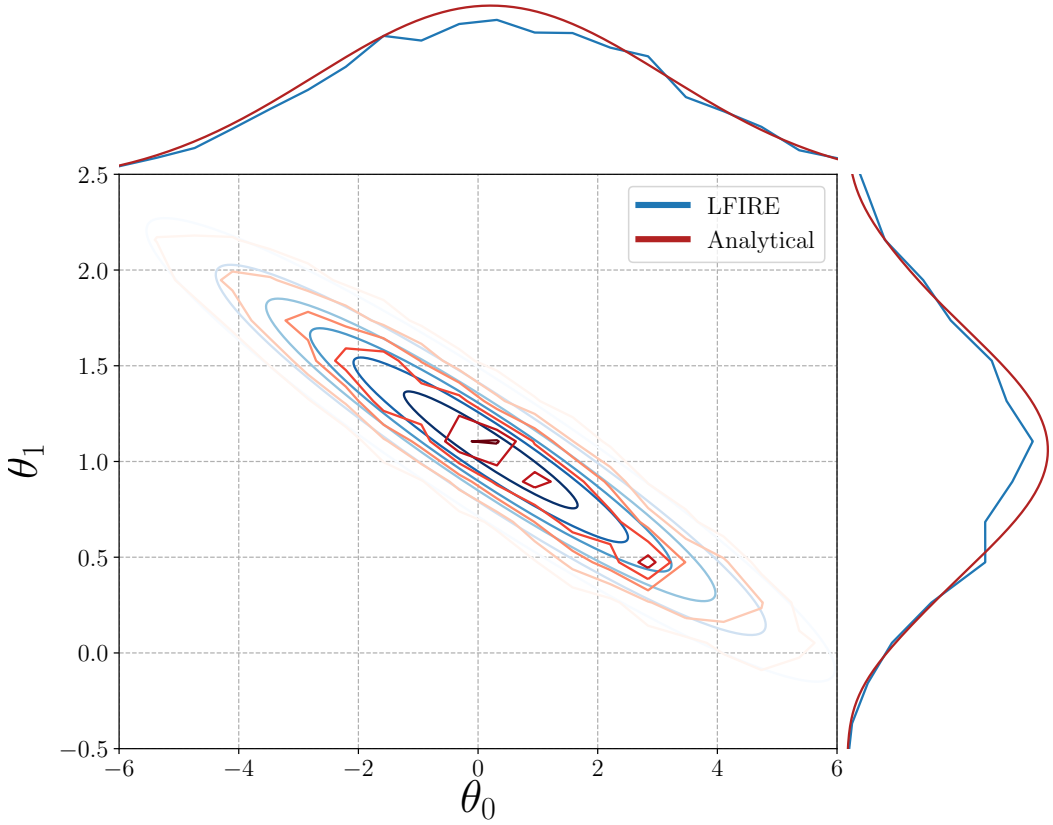

Fig. 2.3 Contour plot of the LFIRE posterior (shown in red) and the analytical posterior (shown in blue) for the toy example, evaluated on a parameter grid. The true parameters of the model were $\boldsymbol{\theta}_{true} = [5.00, -0.12]^\top$ and the observed data was $(z, y) = (5.00, 4.37)$. The top and right plots are projections of the distributions onto the θ_0 and θ_1 axes, respectively.

$U(\mathbf{d})$. This is an optimisation scheme for objective functions that we can evaluate but whose form and gradients are unknown.

The general idea of Bayesian Optimisation is to build a probabilistic model of an objective function and to use the model in order to decide where to evaluate the objective function next. There are thus two essential parts to a Bayesian Optimisation approach: (1) we need to define a *prior probability measure* on the objective function that, given the evaluation, will be updated to a posterior and (2) we need to define a posterior-dependent *acquisition function* that tells us where to evaluate the function next. The optimal choice for both of these parts is a mature research field and we shall not go into much detail here. For us, the objective function is the expected utility $U(\mathbf{d})$. Every time we obtain an evaluation U_i, the prior distribution over the expected utility function is updated and the acquisition function

is optimised again. This procedure is very similar to the sequential Bayesian experimental design outlined in Algorithm 1, as both are sequential design strategies, but geared towards different purposes.

A popular choice for the prior over the $U(\mathbf{d})$ function is a Gaussian Process (GP) prior (Rasmussen and Williams, 2005), due to its flexibility and tractability. A GP is a distribution over functions, specified by its prior mean and covariance function $\mu(\mathbf{d})$ and $k(\mathbf{d},\mathbf{d}')$, respectively. Our expected utility function is assumed to be a sample from it, i.e.

$$U(\mathbf{d}) \sim \mathbb{GP}(\mu(\mathbf{d}), k(\mathbf{d},\mathbf{d}')) \tag{2.27}$$

As often done, we assume that the prior mean $\mu(\mathbf{d}) = 0$ (Rasmussen and Williams, 2005). The covariance function, sometimes called GP kernel, indicates how smooth the expected utility function is; it is required to be symmetric and positive-definite. A popular choice of the kernel is the squared exponential function,

$$k(\mathbf{d},\mathbf{d}') = \sigma^2 \exp\left(-\frac{\|\mathbf{d}-\mathbf{d}'\|^2}{2l^2}\right), \tag{2.28}$$

where l^2 and σ^2 are positive parameters. Using a set of expected utility evaluations $\{\mathbf{d}_i, U_i\}$, we can then obtain a GP posterior and posterior predictive distribution over $U(\mathbf{d})$. The posterior predictive distribution of $U(\mathbf{d})$ at a new, potential design point $\mathbf{d}_{new}$ is given by

$$U(\mathbf{d}_{new}) \mid \{\mathbf{d}_i, U_i\} \sim \mathcal{N}(m(\mathbf{d}_{new}), \rho^2(\mathbf{d}_{new})) \tag{2.29}$$

The posterior predictive mean $m(\mathbf{d}_{new})$ and variance $\rho^2(\mathbf{d}_{new})$ have closed form solutions that can be found in Rasmussen and Williams (2005) (we shall omit the mathematical details here as they will not be explicitly needed).

Acquisition functions are generally defined such that potentially high values of the objective function in question result in high acquisitions. The next design point at which we want to evaluate our function is given by maximising the acquisition function. A popular choice for the acquisition function $\alpha(\mathbf{d}_{new})$ is the *Expected Improvement* (EI) (Mockus et al., 1978), given by

$$\alpha_{EI}(\mathbf{d}_{new}) = \rho(\mathbf{d}_{new})\left[\gamma(\mathbf{d}_{new})\Phi(\gamma(\mathbf{d}_{new})) + \mathcal{N}(\gamma(\mathbf{d}_{new}); 0, 1)\right] \tag{2.30}$$

where

$$\gamma(\mathbf{d}_{new}) = \frac{U(\mathbf{d}_{best}) - m(\mathbf{d}_{new}) + \psi}{\rho(\mathbf{d}_{new})}, \tag{2.31}$$

with the parameters $m(\mathbf{d}_{new})$ and $\rho(\mathbf{d}_{new})$ being the GP posterior predictive distribution mean and standard deviation. Φ is the cumulative function of the standard Normal distribution and ψ is a tunable parameter that governs how much we would like to explore the design space. We have also denoted the best current design point as $\mathbf{d}_{best} = \arg\max_{\mathbf{d}_i} U(\mathbf{d}_i)$.

We shall use Gaussian Process Bayesian Optimisation with the EI acquisition function as implemented in the GPyOpt package in Python (GPyOpt, 2016) to optimise the expected utility $U(\mathbf{d})$. This package only allows for minimisation of the objective function; therefore, instead of maximising the expected utility, we shall minimise the negative expected utility, which ought to yield the same result. Since GPyOpt also allows for batch parallelisation (González et al., 2015), we can make use of multiple CPU cores to perform the optimisation process.

By substituting the grid search in Algorithm 1 with Bayesian Optimisation, we obtain a new sequential Bayesian experimental design procedure, as shown in Algorithm 3.

Algorithm 3 Sequential Bayesian Experimental Design with Bayesian Optimisation

Let $\mathbb{D}_0 = \varnothing$
for t=1 to t=T **do**
    Sample model parameters from prior: $\boldsymbol{\theta}^{(i)} \sim p(\boldsymbol{\theta} \mid \mathbb{D}_{t-1})$ for $i = 1, \ldots, N$
    Start with a randomly chosen design $\mathbf{d}_{j=0}$
    while $\mathbf{d}_{best}$ not converged **do**
        Simulate data from the model: $\mathbf{y}^{(i)} \sim p(\mathbf{y} \mid \mathbf{d}_j, \boldsymbol{\theta}^{(i)})$ for $i = 1, \ldots, N$
        Compute $\widehat{U}_t(\mathbf{d}_j)$ given in Equation 2.18
        if $\widehat{U}_t(\mathbf{d}_j) < \widehat{U}_t(\mathbf{d}_{best})$ **then**
            Let $\mathbf{d}_{best} = \mathbf{d}_j$
        end if
        Compute $\alpha_{EI}(\mathbf{d}_{new})$ in Equation 2.30 for a range of potential designs
        Let $\mathbf{d}_{j+1} = \arg\max_{\mathbf{d}_{new}} \alpha_{EI}(\mathbf{d}_{new})$
    end while
    Let $\mathbf{d}_t^* = \mathbf{d}_{best}$
    Perform an experiment at $\mathbf{d}_t^*$ to observe some real data $\mathbf{y}_t^*$
    Update the prior distribution by updating the data set: $\mathbb{D}_t = \mathbb{D}_{t-1} \cup \{\mathbf{d}_t^*, \mathbf{y}_t^*\}$
end for

Consider the linear toy model that we previously used to illustrate Bayesian experimental design in Section 2.2. In that example, we optimised the expected utility by evaluating it on a grid of 100 design points $z \in [0, 12]$ km for every iteration t. We here utilise Algorithm 3 instead, i.e. use Bayesian Optimisation to decide where we should evaluate the expected utility next. While we need 100 points to completely explore the design space using the grid

evaluations, we show next that using Bayesian Optimisation results in a lot less evaluations, while still retaining enough exploration.

For illustration purposes, we have optimised the expected utility of the toy model for the first iteration only — this is shown in Figure 2.4, together with the grid search expected utility as well.

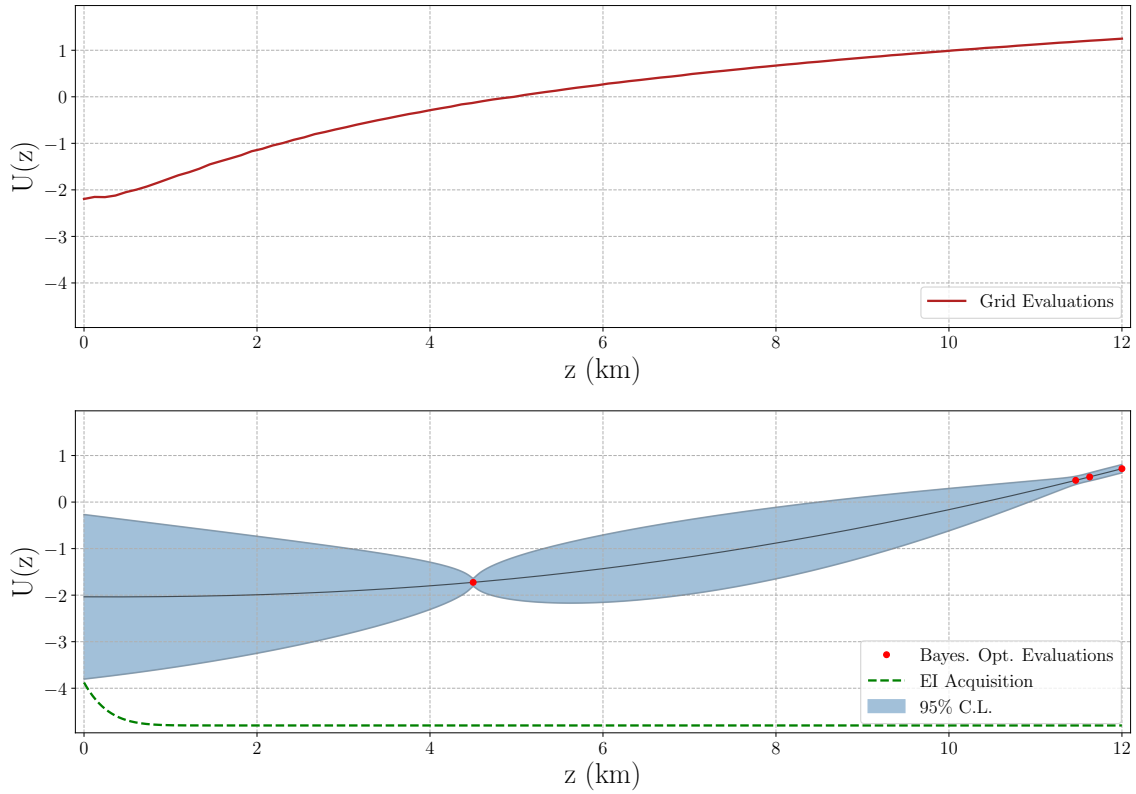

Fig. 2.4 Expected utility maximisation of the linear toy model using the grid method (top figure) and the Bayesian optimisation method (bottom plot). Shown are the expected utilities for the first iteration of the sequential Bayesian experimental design process outlined in Algorithm 1. For the bottom plot, the mean and 95% confidence interval of the posterior predictive distribution are shown in blue, while the EI acquisition function is shown in green.

As can be seen, we only need 4 evaluations of the expected utility to solve the optimisation problem in Equation 2.6, as opposed to 100 for the grid method. In the figure, we show the mean and a 95% confidence interval of the posterior predictive distribution, as well as the EI acquisition function. The confidence interval quickly becomes miniscule as we do more expected utility evaluations.

2.5 Example Models

In Section 2.1 we introduced a linear toy model that we used to illustrate several methods in this chapter. We will test the methodology developed in this work on some further models that are more realistic and have been used in the literature before. Due to the sequential Bayesian experimental literature not being particularly extensive, the choice of suitable models was limited. We decided to focus on the *Death Model* (Cook et al., 2008) and the *SIR Model* (Allen, 2008) from epidomology. For both models, the design variable **d** is time. In other words, the aim is to find out at what time τ we would have to make our measurements in order to most accurately estimate the model parameters.

2.5.1 Death Model

The Death Model is a stochastic process that describes the decline of a population due to some infection. Suppose that we have an initial population N, all of which are in the susceptible state S, i.e. they can be infected. In this particular model, we describe the change from a susceptible state S to an infected state I as a continuous-time Markov process with constant infection rate b (Cook et al., 2008). At time τ, the number of susceptibles and infected are given by $S(\tau)$ and $I(\tau)$, respectively. The probability that the number of infected increases by one within the next infinitesimal time frame $\delta\tau$ is thus modelled by $\Pr[I(\tau+\delta\tau) = I(\tau)+1] = bS(\tau)\delta\tau + \mathcal{O}(\delta\tau)$.

This means that each individual has a chance $p(\tau) = 1-\exp(-b\tau)$ of getting infected by time τ (Cook et al., 2008). The number of infected, modelled by a Binomial distribution, at time point τ_i is given by

$$I(\tau_i) \sim \text{Bin}(N, p(\tau_i)) \tag{2.32}$$

If we have made successive observations, we can condition on the previous observations $I(\tau_{i-1})$ to find that

$$I(\tau_i) - I(\tau_{i-1}) \sim \text{Bin}(N - I(\tau_{i-1}), p(\tau_i - \tau_{i-1})) \tag{2.33}$$

For illustration, assume that the true infection rate for this model is $b_{true} = 1.5$ and that we start out with $\tau_0 = 0$, $I(\tau_0) = 0$ and $S(\tau_0) = 50$. The distribution of the number of infected as a function of time for 1000 Death processes is shown in Figure 2.5.

The number of infected rises exponentially with time and converges to $I(\tau \to \infty) = 50$. The highest uncertainty in the model occurs in the region of $0.5 < \tau < 1.5$ and decreases for low and high τ.

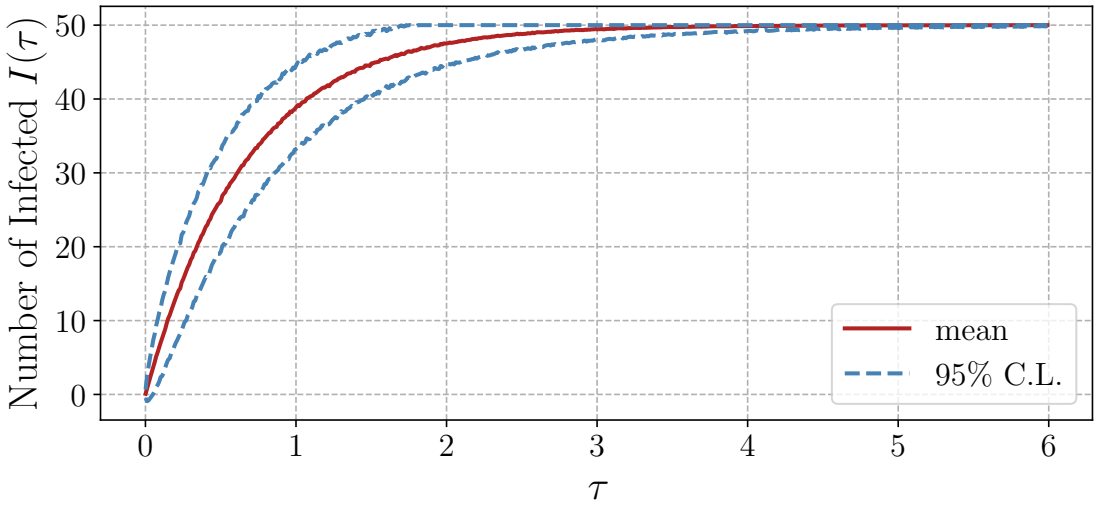

Fig. 2.5 Number of infected $I(\tau)$ as a function of time τ for the Death Model with a true infection rate of $b_{true} = 1.5$. In addition to the mean number of infected (shown in red), there are also 95% confidence limits (shown in blue).

2.5.2 SIR Model

The SIR Model (Allen, 2008) can be understood as a more complex version of the Death Model. In addition to the number of susceptibles $S(\tau)$ and infected $I(\tau)$, we also have $R(\tau)$ recovered individuals. Those that have recovered from their infection cannot be infected again, e.g. they have developed antibodies against the disease that we are trying to model. We shall ignore vital dynamics such as birth and death rates here, i.e. the total population stays the same: $N = S(\tau) + I(\tau) + R(\tau)$.

At time τ_i, we define that the probability of a susceptible getting infected is $p_{\text{inf}}(\tau_i) = \beta I(\tau_i)/N$, where $\beta \in [0,1]$. Unlike for the Death Model, the probability of infection here depends on the number of infected as well, meaning that individuals that are already infected can also further infect others. Similarly, we define the probability of an infected recovering from the disease as $p_{\text{rec}}(\tau_i) = \gamma$, where $\gamma \in [0,1]$. At a particular time, let the number of suspectibles that move to the infected state be given by ΔI and the number of infected that move to the recovered state given by ΔR. These two population changes are computed by sampling from a Binomial distribution, i.e.

$$\Delta I \sim \text{Bin}(S(\tau_i), p_{\text{inf}}(\tau_i)) \tag{2.34}$$

$$\Delta R \sim \text{Bin}(I(\tau_i), p_{\text{rec}}(\tau_i)) \tag{2.35}$$

Using a discrete timestep $\tau_{i+1} = \tau_i + \Delta\tau_i$ we then update the different state populations:

$$S(\tau_{i+1}) = S(\tau_i) - \Delta I \tag{2.36}$$

$$I(\tau_{i+1}) = I(\tau_i) + \Delta I - \Delta R \tag{2.37}$$

$$R(\tau_{i+1}) = R(\tau_i) + \Delta R \tag{2.38}$$

We shall start this time series with $N-1$ susceptibles, 1 infected and zero recovered.

The infection rate β and the recovery rate γ dictate how fast the state transitions happen. The dynamics of the number of infected can be described by the so-called basic reproduction number $R_0 = \beta/\gamma$. If this number is less than one, the disease is said to be 'stable', i.e. it will eventually be eliminated (Dietz, 1993). If $R_0 > 1$, then the disease is 'unstable' and will eventually spread; diseases spread faster for higher R_0. For instance, Ebola has $2 < R_0 < 3$, Smallpox has $R_0 \approx 6$ and Malaria has $R_0 \approx 16$ (McCandless, 2014).

Assume for now that we wanted to model a disease such as Ebola; we choose that $\beta_{true} = 0.15$ and $\gamma_{true} = 0.05$, such that $R_0 = 3$. We shall simulate the SIR Model 1000 times and discretise the time design space into 300 steps between 0 and 3, i.e. $\Delta\tau = 0.01$. Starting with $S(\tau_0) = 49$, $I(\tau_0) = 1$ and $R(\tau_0) = 0$, the mean state populations are shown in Figure 2.6

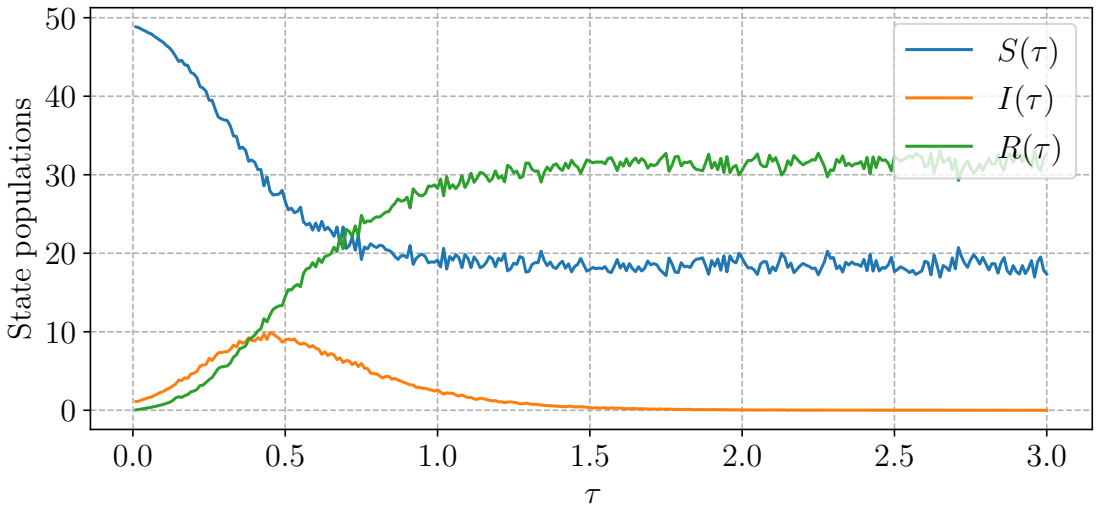

Fig. 2.6 SIR Model state populations for the number of susceptibles $S(\tau)$, the number of infected $I(\tau)$ and the number of recovered $R(\tau)$. The true model parameters used where $\beta_{true} = 0.15$ and $\gamma_{true} = 0.05$, resulting in a basic reproduction number of $R_0 = 3$. The state populations were computed for 1000 SIR Model processes, where the time design space was discretised into 300 steps between 0 and 3.

Chapter 3

Research Objectives

3.1 Previous Work

While the Bayesian experimental design literature is extensive for models with tractable likelihoods (see the review paper by Ryan et al., 2016b), there is little work about simulator-based statistical models. This is a relatively recent and growing research area that leaves ample room for exploration.

One of the earliest efforts into Bayesian experimental design for simulator models, where the likelihood is intractable, was done by Drovandi and Pettitt (2013), who focussed on designing experiments in epidemiology. They used the optimisation algorithm of Müller (1999), which is a sampling-based approach that converts the optimisation problem into a problem of sampling from a distribution for which the optimal design point is the mode. They found, however, that this algorithm is slow to converge. In order to obtain samples from the posterior distribution, they used the popular method of Approximate Bayesian Computation (ABC) rejection sampling (Beaumont et al., 2002). The utility function they used depended on the precision of the posterior variances, as opposed to the Kulback-Leibler (KL) divergence (Kullback and Leibler, 1951) that was discussed in Section 2.2. Overall, while they found their method to be working well, ABC rejection sampling for likelihood-free inference is computationally expensive, preventing them to consider higher dimensions and expensive simulators. In addition, their methodology required matching directly on previous real-world observations, which may not always be available. Drovandi and Pettitt (2013) also considered a non-myopic design approach, i.e. where they aim to find several optimal design points, not just one, at once. Other authors (e.g. Dehideniya et al., 2018; Price et al., 2016) have also used ABC rejection sampling to solve optimal Bayesian design problems,

specifically for the field of epidemiology. Their methods also suffer from the issue that high-dimensional design spaces, e.g. with more than four dimensions, are not feasible.

Instead of focussing on time-dependent models, Hainy et al. (2016b) have used ABC rejection sampling to perform Bayesian experimental design on a spatial weather model. They suffered from the same drawbacks as Drovandi and Pettitt (2013), i.e. that direct comparison between simulations and observations are needed for the ABC algorithm; in addition, they found that choosing suitable summary statistics for their model was difficult.

Other than ABC rejection sampling, there has also been some work by Ryan et al. (2016a) and Overstall and McGree (2018) using auxiliary modelling to solve Bayesian experimental design problems with intractable likelihoods. The former used sampling-based approaches and therefore their method was restricted to small dimensions. While the latter's methodology is efficient, it only works on likelihood-based utility functions, which may not always represent the experiment accurately.

All aforementioned work has only focussed on static design and there has been little to no effort into optimal sequential design for models with intractable likelihoods. Hainy et al. (2016a) have previously touched on that topic by considering a likelihood-free extension of a sequential Monte-Carlo (SMC) algorithm. They considered ABC rejection sampling and Synthetic Likelihood (SL) Wood (2010) for obtaining samples from the posterior distribution. However, they only applied their method to a simple toy model that was low-dimensional.

3.2 Goals

It becomes apparent that there needs to be more effort into sequential Bayesian experimental design for simulator-based models. In addition, there has been no work on using more advanced likelihood-free inference methods such as the LFIRE approach outlined in Section 2.3 for Bayesian experimental design. Dutta et al. (2016) found that LFIRE performed better than ABC or SL for the models tested. Theoretically, LFIRE does not have any Gaussianity assumption for the summary statistics and also does not require any previous real-world observations. Furthermore, except swhere the algorithm of Müller (1999) was used, all work has used grid evaluations to minimise the expected utility function — as we have seen in Section 2.4, a Bayesian Optimisation approach might be more efficient, especially for high-dimensional design spaces and when evaluating the utility is expensive. Also, only Drovandi and Pettitt (2013) have considered non-myopic optimal designs, i.e. where we aim to choose more than one optimal design point, and no work has been done on non-myopic sequential

Bayesian experimental design. This may be advantageous in situations where we know exactly how many data points we are able to take, e.g. due to cost or time restrictions.

We want to address these research gaps in this dissertation. Thus, our research goals are:

I. Constructing a sequential Bayesian experimental design framework that uses the modern and more robust LFIRE approach to likelihood-free inference.

II. Solving the optimal design problem using Bayesian optimisation, instead of discretising the design space into a grid,

III. Considering the application of the sequential design framework to non-myopic situations.

After addressing points I–III, we hope to have constructed an efficient sequential Bayesian experimental design framework for simulator-based statistical models. We aim to devise a framework that is robust to different simulator models, design dimensions and parameter dimensions. Because the aforementioned research gaps have never been addressed before, however, we are not planning on beating any kind of benchmark but focus on establishing the foundations and providing a proof of concept.

Chapter 4

Sequential Utility Computation

In order to build an efficient sequential Bayesian experimental design framework for models with intractable likelihoods, we have to ultimately combine Algorithm 1 and Algorithm 2, introduced in Chapter 2. In other words, we have to construct a sequential design algorithm that utilises likelihood-free inference methods for the posterior computations. This chapter in particular focusses on the computation of the sequential utility on a grid of design points. We consider myopic design, where the aim is to find the single, next best design point.

In Section 4.1 we show how to combine static Bayesian experimental design and LFIRE to find the optimal design for the first iteration using grid search. For subsequent iterations, we need to know how to sample from the updated prior distribution which we explain in Section 4.2. In Section 4.3 we devise a resampling strategy for situations when the effective sampling size of our approximations gets too low. We show how to slightly alter LFIRE for sequential design in Section 4.4 and then go on to describe a fully sequential Bayesian experimental design scheme in Section 4.5. Most of the sections hitherto were illustrated by means of a linear toy model; we test the devised framework on two epidemiological models in Section 4.6.

4.1 Static Design Utility

Let us recall the mutual information expected utility function for *static experimental design*,

$$U(\mathbf{d}) = \int\int \log\left(\frac{p(\boldsymbol{\theta} \mid \mathbf{d}, \mathbf{y})}{p(\boldsymbol{\theta})}\right) p(\mathbf{y} \mid \boldsymbol{\theta}, \mathbf{d}) p(\boldsymbol{\theta}) \mathrm{d}\boldsymbol{\theta} \mathrm{d}\mathbf{y}. \tag{4.1}$$

Looking at the definition of the LFIRE ratio in Equation 2.20, we can substitute the ratio $r(\mathbf{d}, \mathbf{y}, \boldsymbol{\theta})$ into the logarithm in Equation 4.1 to obtain

$$U(\mathbf{d}) = \int \int \log\left[r(\mathbf{d}, \mathbf{y}, \boldsymbol{\theta})\right] p(\mathbf{y} \mid \boldsymbol{\theta}, \mathbf{d}) p(\boldsymbol{\theta}) \mathrm{d}\boldsymbol{\theta} \mathrm{d}\mathbf{y}. \tag{4.2}$$

As done previously, we can approximate this double integral via Monte-Carlo integration,

$$\widehat{U}(\mathbf{d}) \approx \frac{1}{N} \sum_{i=1}^{N} \log\left[r(\mathbf{d}, \mathbf{y}^{(i)}, \boldsymbol{\theta}^{(i)})\right], \tag{4.3}$$

where $\mathbf{y}^{(i)} \sim p(\mathbf{y} \mid \mathbf{d}, \boldsymbol{\theta}^{(i)})$ and $\boldsymbol{\theta}^{(i)} \sim p(\boldsymbol{\theta})$. We have already shown in Algorithm 2 how to compute the LFIRE ratio for a given model parameter $\boldsymbol{\theta}$ and design point $\mathbf{d}$.

Consider again the linear toy model from Section 2.1. In Section 2.3 we calculated LFIRE ratios for this toy model on a 2-d grid of model parameters $\boldsymbol{\theta}$ at design point $z = 5$ km. In order to compute the expected utility given in Equation 4.3, we first obtain prior samples $\boldsymbol{\theta}^{(i)} \sim \mathcal{N}(\boldsymbol{\theta}; 0, 5^2)$. We then compute the LFIRE ratio for each of the prior samples in the set $\{\boldsymbol{\theta}^{(i)}\}_{i=1}^{1000}$. When simulating data from the likelihood for a given prior sample, we can use the rest of the samples in $\{\boldsymbol{\theta}^{(i)}\}_{i=1}^{1000}$ to simulate data from the marginal (see Algorithm 2). As done in Section 2.2, we discretise the design space between $z = 0$ km and $z = 12$ km into 100 points. For each design point z on our grid we then compute the expected utility according to Equation 4.3. The resulting expected utility function is shown in Figure 4.1, together with the analytical static expected utility (see Section 2.2).

For low z, the static expected utility computed using LFIRE ratios roughly follows the same behaviour as the analytical expected utility. However, as the altitude z increases, the difference between the approximate and analytical method grows. In fact, while the analytical expected utility keeps on growing, the expected utility computed using LFIRE ratios seems to plateau after $z \approx 6$. While the optimal design lies at $z_1^* = 12$ km for the analytical method, we obtain $z_1^* = 8.35$ km for the approximation using LFIRE. Because the expected utility computed using LFIRE plateaus however, the posterior distribution at $z_1^* = 12$ km and $z_1^* = 8.35$ km would presumably not be much different. Perhaps for the approximation using LFIRE ratios, the measurement noise in the data generating distribution does not have much of an impact at all for design points beyond $z \approx 6$, whereas the analytical version is still affected. We make an observation via the real-world data generating process: using $\boldsymbol{\theta}_{true} = [5.00, -0.12]^\top$ and $z_1^* = 8.35$ km, we observe data $y_1^* = 4.14$ by sampling from a Gaussian of variance 1 (see Section 2.1).

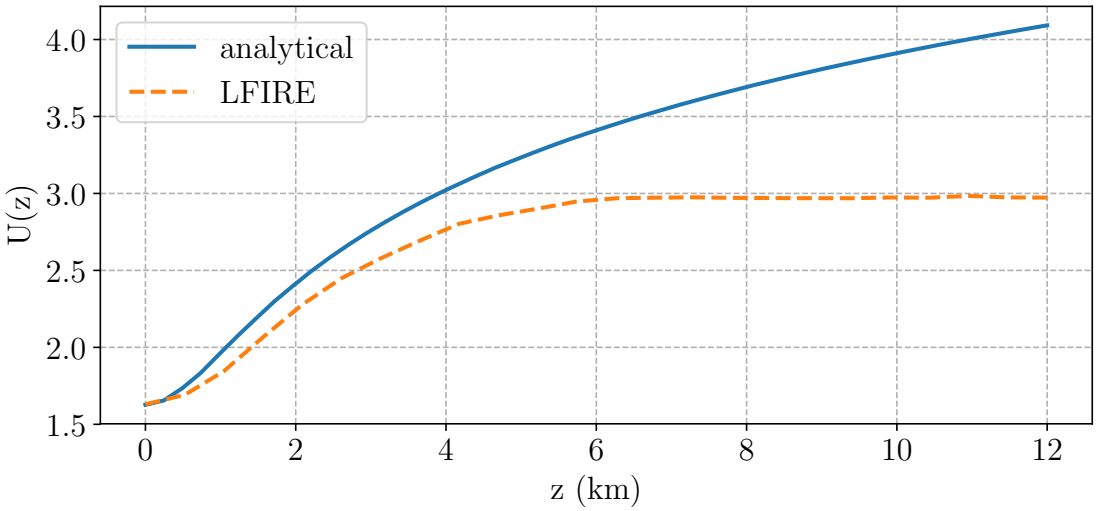

Fig. 4.1 Static expected utility function $U_t(z)$ for the linear toy example computed using LFIRE ratios on a grid of design points. 1000 prior samples were obtained by sampling from $\mathcal{N}(\boldsymbol{\theta}; 0, 5^2)$ and the design space was discretised into 100 steps between $z = 0$ km and $z = 12$ km. As a comparison, the analytical static expected utility calculated in Section 2.2 is also shown.

4.2 Obtaining Updated Prior Samples

The expected utility for static design in Equation 2.15 only requires samples from the initial prior distribution $p(\boldsymbol{\theta})$ which, presumably, we can easily sample from. In order to compute the expected utility for sequential design in Equation 2.18, we instead require samples from the updated prior distribution $p(\boldsymbol{\theta} \mid \mathbb{D}_{t-1})$ — generally, this is not trivial as we do not have its analytical form.

First of all, let us define what it means to update the prior distribution. Consider the static expected utility in Section 4.1 and the first observations $(\mathbf{d}_1^*, \mathbf{y}_1^*)$ that we obtained by maximising it. By definition, the LFIRE ratio is the ratio of the posterior distribution to the prior distribution (see Section 2.3) and therefore we can write the posterior as

$$p(\boldsymbol{\theta} \mid \mathbb{D}_1) = r_1(\mathbf{d}_1^*, \mathbf{y}_1^*, \boldsymbol{\theta}) p(\boldsymbol{\theta}). \tag{4.4}$$

In other words, we update the initial prior distribution $p(\boldsymbol{\theta})$ by multiplying it with the LFIRE ratio evaluated at the observed data points. In order to obtain samples from the updated prior $p(\boldsymbol{\theta} \mid \mathbb{D}_1)$, we shall construct weights $w_2^{(i)} = r_1(\mathbf{d}_1^*, \mathbf{y}_1^*, \boldsymbol{\theta}^{(i)})$, where $\boldsymbol{\theta}^{(i)} \sim p(\boldsymbol{\theta})$. Each of

these weights $w_2^{(i)}$ essentially tells us how likely it is that a particular parameter sample $\boldsymbol{\theta}^{(i)}$ from the prior is also a sample from the updated prior distribution.

Recall that the LFIRE ratio depends on the dot product of some coefficients $\boldsymbol{\beta}(\mathbf{d}, \boldsymbol{\theta})$ and summary statistics $\boldsymbol{\psi}(\mathbf{y})$ of the data (See Section 2.3). Thus, we do not need to solve the logistic regression problem in Algorithm 2 again but instead we can simply store the coefficients for every design point and initial prior sample. Once we have selected an optimal design point $\mathbf{d}^*$ and observed some data $\mathbf{y}^*$, we take the stored coefficients for that design point and use $\boldsymbol{\beta}(\mathbf{d}^*, \boldsymbol{\theta}^{(i)})$ and $\boldsymbol{\psi}(\mathbf{y}^*)$ to compute $w_2^{(i)} = r_1(\mathbf{d}_1^*, \mathbf{y}_1^*, \boldsymbol{\theta}^{(i)})$. This allows us to save computation time, as we do not need to approximate all the ratios at the optimal design point again.

The question remains how we can use Equation 4.4 to obtain parameter samples from $p(\boldsymbol{\theta} \mid \mathbb{D}_1)$. Using the observations $(\mathbf{d}_1^*, \mathbf{y}_1^*)$ that we obtained by performing the experiment and and the initial samples from a Gaussian prior, we can construct a set of pairs $\{w_2^{(i)}, \boldsymbol{\theta}^{(i)}\}_{i=1}^{i=N}$. Afterwards, we normalise the weights: $W_2^{(i)} = w_2^{(i)} / \Sigma_{i=1}^N w_2^{(i)}$. Followingly, we sample an index from a categorical distribution, i.e $I \sim \text{cat}(\{W_2^{(i)}\})$, and choose the initial prior sample $\boldsymbol{\theta}^{(I)}$. Doing this for a certain number of times results in a set of parameter samples that follows the updated prior distribution $p(\boldsymbol{\theta} \mid \mathbb{D}_1)$. Many samples from the updated prior distribution will be identical, as those with high weights will be selected more often.

We can again use the linear toy model to illustrate this method. Using the previously computed LFIRE ratio coefficients at $z_1^* = 8.35$ km, we can obtain weights $w_2^{(i)} = r_1(z_1^*, y_1^*, \boldsymbol{\theta}^{(i)})$ for the observed data $y_1^* = 4.14$. The value of the weight $w_2^{(i)}$ for each initial prior sample $\boldsymbol{\theta}^{(i)}$ is shown in the left plot in Figure 4.2. The updated prior samples obtained by sampling from a categorical distribution, using the method described previously, is shown in the right plot in Figure 4.2. Note how there are several identical updated prior samples, discernible by a darker shade.

The left plot in Figure 4.2 indicates that the initial prior samples with high weights $w_2^{(i)}$ all lie on a straight line and are not focussed around one particular point. This could be explained by the fact that the linear toy model is not fully identifiable from a single data point. Therefore we would have to obtain more observations to get a more concentrated distribution of parameters with high weights. The updated prior samples in the right plot in Figure 4.2 all correspond to initial prior samples with high weights, as intended. Some of the weights in the left plot have extremely low values, e.g. $\approx 10^{-30}$, and hence we do not see any updated prior samples from initial prior samples with low weights.

We can generalise the aforementioned procedure to obtaining updated prior samples at any iteration t. Consider the updated prior distribution $p(\boldsymbol{\theta} \mid \mathbb{D}_{t-1})$; we rewrite this

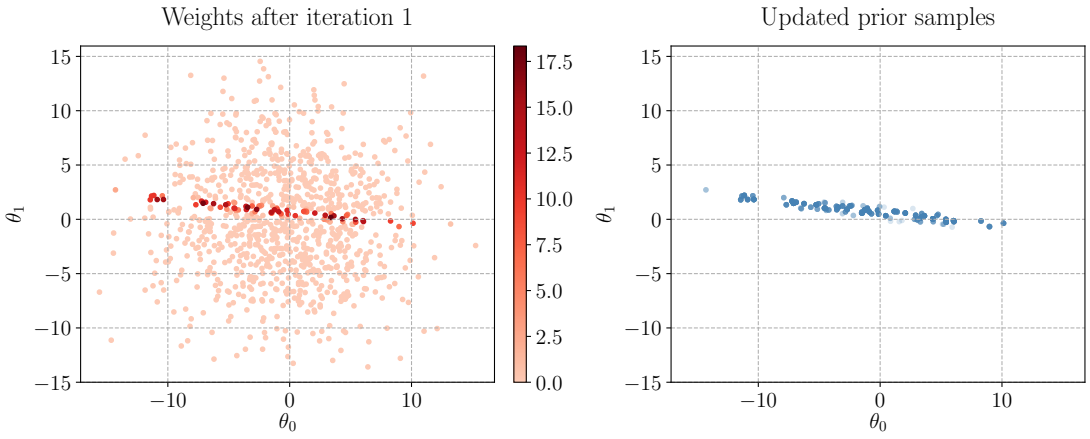

Fig. 4.2 Left: Weight values $w_2^{(i)}$ for each initial prior sample $\boldsymbol{\theta}^{(i)}$ for the toy model. Right: Updated prior samples obtained by sampling from a categorical distribution using normalised weights $W_2^{(i)}$. A darker shade implies a higher frequency of identical parameters.

distribution as a function of the initial prior $p(\boldsymbol{\theta})$,

$$p(\boldsymbol{\theta} \mid \mathbb{D}_{t-1}) = \frac{p(\boldsymbol{\theta} \mid \mathbb{D}_{t-1})}{p(\boldsymbol{\theta} \mid \mathbb{D}_{t-2})} \frac{p(\boldsymbol{\theta} \mid \mathbb{D}_{t-2})}{p(\boldsymbol{\theta} \mid \mathbb{D}_{t-3})} \dots \frac{p(\boldsymbol{\theta} \mid \mathbb{D}_1)}{p(\boldsymbol{\theta})} p(\boldsymbol{\theta}). \tag{4.5}$$

As done in Equation 4.4, once we have made observations $(\mathbf{d}_t^*, \mathbf{y}_t^*)$ we update the prior distribution by evaluating the estimated ratios at the observed data points, i.e.

$$p(\boldsymbol{\theta} \mid \mathbb{D}_t) = r_t(\mathbf{d}_t^*, \mathbf{y}_t^*, \boldsymbol{\theta}, \mathbb{D}_{t-1}) p(\boldsymbol{\theta} \mid \mathbb{D}_{t-1}), \tag{4.6}$$

Using this result, we can rewrite the ratios of updated prior distributions in Equation 4.5 as

$$p(\boldsymbol{\theta} \mid \mathbb{D}_{t-1}) = r_{t-1}(\mathbf{d}_{t-1}^*, \mathbf{y}_{t-1}^*, \boldsymbol{\theta}, \mathbb{D}_{t-2}) r_{t-2}(\mathbf{d}_{t-2}^*, \mathbf{y}_{t-2}^*, \boldsymbol{\theta}, \mathbb{D}_{t-3}) \dots r_1(\mathbf{d}_1^*, \mathbf{y}_1^*, \boldsymbol{\theta}) p(\boldsymbol{\theta}) \tag{4.7}$$

$$= w_t(\boldsymbol{\theta}, \mathbb{D}_{t-1}) p(\boldsymbol{\theta}), \tag{4.8}$$

where we have defined the weight w_t to be

$$w_t(\boldsymbol{\theta}, \mathbb{D}_{t-1}) = \prod_{s=1}^{t-1} r_s(\mathbf{d}_s^*, \mathbf{y}_s^*, \boldsymbol{\theta}, \mathbb{D}_{s-1}), \tag{4.9}$$

with $r_1(\mathbf{d}_1^*, \mathbf{y}_1^*, \boldsymbol{\theta}, \mathbb{D}_0) = r_1(\mathbf{d}_1^*, \mathbf{y}_1^*, \boldsymbol{\theta})$.

For each initial prior sample $\boldsymbol{\theta}^{(i)} \sim p(\boldsymbol{\theta})$ we then have a corresponding weight $w_t^{(i)} = w_t(\boldsymbol{\theta}^{(i)}, \mathbb{D}_{t-1})$ at iteration t. Essentially, in order to compute the weight for each inital prior sample we need to multiply the ratios, evaluated at the observed data points, of all previous iterations. Because we can simply store these ratios after every iteration however, we do not incur any extra computation costs. Note that at the first iteration, we initialise all weights to be one, i.e. $w_1^{(i)} = 1$ for $i = 1, \ldots, N$.

Thus, assuming that at iteration t we start out with a set of weight and initial prior sample pairs $\{w_t^{(i)}, \boldsymbol{\theta}^{(i)}\}_{i=1}^{i=N}$, we use them to obtain samples that conform to the updated prior distribution: Firstly, we normalise the weights: $W_t^{(i)} = w_t^{(i)} / \Sigma_{i=1}^{N} w_t^{(i)}$. Followingly, we sample an index from a categorical distribution, i.e. $I \sim \text{cat}(\{W_t^{(i)}\})$, and choose the initial prior sample $\boldsymbol{\theta}^{(I)}$. Doing this for a certain number of times results in a set of parameter samples that follows the updated prior distribution $p(\boldsymbol{\theta} \mid \mathbb{D}_{t-1})$.

Algorithm 4 summarises the aforementioned, generalised procedure for obtaining samples from the updated prior distribution $p(\boldsymbol{\theta} \mid \mathbb{D}_{t-1})$ at iteration t, given some weights and initial prior samples.

Algorithm 4 Obtaining Updated Prior Samples

1: At iteration t, start with weight and parameter pairs $\{w_t^{(i)}, \boldsymbol{\theta}^{(i)}\}_{i=1}^{i=N}$
2: Normalise the weights: $W_t^{(i)} = w_t^{(i)} / \Sigma_{i=1}^{N} w_t^{(i)}$ for $i = 1, \ldots, N$
3: **for** i=1 to i=N **do**
4: Sample from a categorical distribution: $I \sim \text{cat}(\{W_t^{(i)}\})$
5: Choose $\boldsymbol{\theta}^{(I)}$ as a sample from the updated prior distribution
6: **end for**

4.3 Resampling

Equation 4.9 says that a sample weight $w_t^{(i)}$ is computed by the product of all previous LFIRE ratios at initial prior sample $\boldsymbol{\theta}^{(i)}$, evaluated at the observed data points. This also means that less significant prior samples, i.e. ones that have low ratios, have weights that quickly decay to zero. Eventually, we are left with only a small number of prior samples that are realistically chosen through the categorical sampling approach outlined in Algorithm 4; we can quantify this number via the *effective sample size* η (Kish, 1965), defined by

$$\eta = \frac{\sum_{i=1}^{N} \left(w_t^{(i)} \right)^2}{\left(\sum_{i=1}^{N} w_t^{(i)} \right)^2}. \tag{4.10}$$

If η is small, i.e. $\eta \ll N$, then we do not cover much relevant parameter space and our approximations may become poor. Thus, if the effective sample size becomes smaller than a minimum sample size $\eta_{\min}$ we would like to resample our initial prior samples; this allows us to have a set of new prior samples that loosely conforms to the most recent updated prior distribution. Throughout this work we shall use a $\eta_{\min}$ that is 5% of N, e.g. 50 if $N = 1000$.

Consider again the set of weight and initial parameter sample pairs $\{w_t^{(i)}, \boldsymbol{\theta}^{(i)}\}_{i=1}^{i=N}$, obtained through the procedure explained in Section 4.2. We are going to model the posterior distribution $p(\boldsymbol{\theta} \mid \mathbb{D}_t)$ of the current iteration t as a spherical Mixture of Gaussians model, i.e.

$$p(\boldsymbol{\theta} \mid \mathbb{D}_t) \approx \sum_{i=1}^{N} W_t^{(i)} \mathcal{N}(\boldsymbol{\theta}; \boldsymbol{\theta}^{(i)}, \mathbb{I}\sigma^2), \tag{4.11}$$

where $\mathbb{I}$ is the identity matrix, σ^2 is the variance of the Gaussians and $W_t^{(i)}$ are the normalised weights. Note that we have one Gaussian for every initial prior sample; each Gaussian is centered at that parameter sample $\boldsymbol{\theta}^{(i)}$ and has the same variance σ^2.

We compute the Gaussian variance by first using a k-dimensional (KD) tree (Bentley, 1975) to find the nearest neighbour $\mathrm{NN}(\boldsymbol{\theta}^{(i)})$ of each initial prior sample. We then calculate the squared median of all the distances of a prior sample to its nearest neighbour and multiply that by two (to allow for greater coverage) in order to obtain the variance, i.e.

$$\sigma^2 = 2 \cdot \mathrm{median}\left(|\boldsymbol{\theta}^{(i)} - \mathrm{NN}(\boldsymbol{\theta}^{(i)})|\right)^2 \tag{4.12}$$

Firstly, we normalise the weights: $W_t^{(i)} = w_t^{(i)} / \Sigma_{i=1}^{N} w_t^{(i)}$. Then, in order to get a new sample from the updated prior distribution, we sample an index from a categorical distribution, i.e. $I \sim \mathrm{cat}(\{W_t^{(i)}\})$, and then obtain a parameter sample from the corresponding Gaussian $\mathcal{N}(\boldsymbol{\theta}; \boldsymbol{\theta}^{(I)}, \mathbb{I}\sigma^2)$. Doing this a number of times should yield a set of new prior samples that are in the vicinity of initial prior samples that have high weight; they should thus loosely conform to the updated prior distribution. Afterwards, we set the weights of each of the new prior samples to one, just like we did for the first iteration. The aforementioned procedure of resampling parameters is outlined in Algorithm 5.

As done previously, we shall use the linear toy model to illustrate the resampling procedure in Algorithm 5. We have already calculated the required weights $w_2^{(i)} = r_1(z_1^*, y_1^*, \boldsymbol{\theta}^{(i)})$ for all initial prior samples in Section 4.2 (see the left plot in Figure 4.2). Figure 4.3 shows the categorically-sampled parameters (see the right plot in Figure 4.2), as well as the resampled model parameters that ought to follow the updated prior distribution.

Algorithm 5 Resampling via a Mixture of Gaussian model

1: At iteration t, start with weight and parameter pairs $\{w_t^{(i)}, \boldsymbol{\theta}^{(i)}\}_{i=1}^{i=N}$
2: Normalise the weights: $W_t^{(i)} = w_t^{(i)} / \Sigma_{i=1}^{N} w_t^{(i)}$ for $i = 1, \ldots, N$
3: Find the nearest neighbour of each initial prior sample
4: Compute the variance for the Mixture of Gaussian model, according to Equation 4.12
5: **for** i=1 to i=N **do**
6: Sample from a categorical distribution: $I \sim \text{cat}(\{W_t^{(i)}\})$
7: Sample $\boldsymbol{\theta}_{new}^{(i)}$ from the Gaussian $\mathcal{N}(\boldsymbol{\theta}; \boldsymbol{\theta}^{(I)}, \mathbb{I}\sigma^2)$
8: **end for**

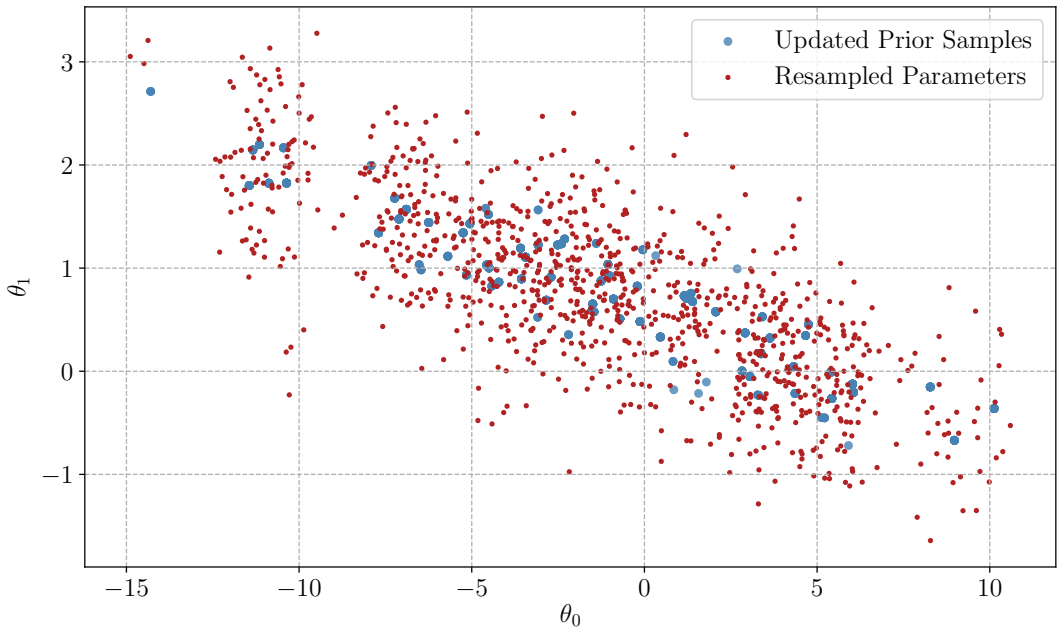

Fig. 4.3 Resampled model parameters (shown in red) by applying Algorithm 5 to the toy model, using the updated prior parameters (shown in blue) computed in Section 4.2.

By modelling the updated prior distribution as a Mixture of Gaussians, the resampled model parameters ought to be close to initial prior samples that have high weights which, looking at Figure 4.3, appears to be correct. Conceptually, these resampled parameters can be understood as unweighted samples from the posterior.

4.4 LFIRE for Sequential Design

For static design, the computation of the LFIRE ratio stays as described in Algorithm 2; LFIRE is slightly more complicated for sequential design however. Given a certain model parameter $\boldsymbol{\theta}$, sampling a data set $\mathbf{Y}_{\boldsymbol{\theta}} = \{\mathbf{y}_{\boldsymbol{\theta}}^{(i)}\}_{i=1}^{n_{\boldsymbol{\theta}}}$ from the data generating distribution $p(\mathbf{y} \mid \mathbf{d}, \boldsymbol{\theta})$ stays the same. The marginal distribution $p(\mathbf{y} \mid \mathbf{d})$ however, is now given by an integral that depends on the updated prior distribution: $p(\mathbf{y} \mid \mathbf{d}) = \int p(\mathbf{y} \mid \mathbf{d}, \boldsymbol{\theta}) p(\boldsymbol{\theta} \mid \mathbb{D}_{t-1}) \mathrm{d}\boldsymbol{\theta}$. Therefore, to generate a data set $\mathbf{Y}_m = \{\mathbf{y}_m^{(i)}\}_{i=1}^{n_m}$ from the marginal distribution, as was done for the traditional LFIRE approach in Algorithm 2, we need to be able to sample from the updated prior distribution $p(\boldsymbol{\theta} \mid \mathbb{D}_{t-1})$. To do this, we utilise the approach outlined in Algorithm 4: obtain a set of weights according to Equation 4.9 and generate a set of updated prior samples by sampling indices from a categorical distribution. We use this set of model parameters that conforms to the updated prior distribution and then generate the data as before.

To calculate an estimate of the ratio $r_t(\mathbf{d}, \mathbf{y}^{(i)}, \boldsymbol{\theta}^{(i)}, \mathbb{D}_{t-1})$ for a particular value of $\boldsymbol{\theta}$ and $\mathbf{d}$, the updated LFIRE algorithm for sequential Bayesian experimental design at iteration t is shown in Algorithm 6.

Algorithm 6 LFIRE for Sequential Design

1: At iteration t, start with weight and parameter pairs $\{w_t^{(i)}, \boldsymbol{\theta}^{(i)}\}_{i=1}^{i=N}$
2: Obtain parameter samples from the updated prior distribution by applying Algorithm 4
3: Using these samples, simulate data from the marginal: $\mathbf{y}_m^{(i)} \sim p(\mathbf{y} \mid \mathbf{d})$ for $i = 1, \ldots, N$
4: Use a parameter $\boldsymbol{\theta}$ to simulate data from the likelihood: $\mathbf{y}_{\boldsymbol{\theta}}^{(i)} \sim p(\mathbf{y} \mid \mathbf{d}, \boldsymbol{\theta})$ for $i = 1, \ldots, N$
5: Estimate $\hat{\boldsymbol{\beta}}_{reg}(\mathbf{d}, \boldsymbol{\theta}, \lambda)$ by solving the optimisation problem posed in Equation 2.26 for a range of λ values
6: Find the λ_{min} that minimises the prediction risk (see Dutta et al., 2016) via ten-fold cross-validation and set $\hat{\boldsymbol{\beta}}(\mathbf{d}, \boldsymbol{\theta}) = \hat{\boldsymbol{\beta}}_{reg}(\mathbf{d}, \boldsymbol{\theta}, \lambda_{min})$
7: For some data $\mathbf{y}$, compute the ratio estimate $\hat{r}_t(\mathbf{d}, \mathbf{y}, \boldsymbol{\theta}, \mathbb{D}_{t-1})$ according to Equation 2.25

As can be seen from a comparison of Algorithm 2 and Algorithm 6, the only thing that has changed for sequential design is the process of sampling data from the marginal distribution.

4.5 Sequential Design Utility

Consider the sequential expected utility that was shown previously in Section 2.2,

$$U_t(\mathbf{d}) = \int\int \log\left(\frac{p(\boldsymbol{\theta} \mid \mathbf{d}, \mathbf{y}, \mathbb{D}_{t-1})}{p(\boldsymbol{\theta} \mid \mathbb{D}_{t-1})}\right) p(\mathbf{y} \mid \boldsymbol{\theta}, \mathbf{d}) p(\boldsymbol{\theta} \mid \mathbb{D}_{t-1}) \mathrm{d}\boldsymbol{\theta}\mathrm{d}\mathbf{y} \tag{4.13}$$

The posterior and prior distributions depend on the observed data points $\mathbb{D}_{t-1} = \{\mathbf{d}^*_{1:t-1}, \mathbf{y}^*_{1:t-1}\}$ and therefore the same applies to the LFIRE ratio, i.e. $r_t = r_t(\mathbf{d}, \mathbf{y}, \boldsymbol{\theta}, \mathbb{D}_{t-1})$. We can write the LFIRE ratio r_t for sequential design as the ratio between posterior and prior distribution,

$$r_t(\mathbf{d}, \mathbf{y}, \boldsymbol{\theta}, \mathbb{D}_{t-1}) = \frac{p(\boldsymbol{\theta} \mid \mathbf{d}, \mathbf{y}, \mathbb{D}_{t-1})}{p(\boldsymbol{\theta} \mid \mathbb{D}_{t-1})} \tag{4.14}$$

Analogous to the static design expected utility, we can substitute this ratio into the logarithm in Equation 4.13:

$$U_t(\mathbf{d}) = \int\int \log\left[r_t(\mathbf{d}, \mathbf{y}, \boldsymbol{\theta}, \mathbb{D}_{t-1})\right] p(\mathbf{y} \mid \boldsymbol{\theta}, \mathbf{d}) p(\boldsymbol{\theta} \mid \mathbb{D}_{t-1}) \mathrm{d}\boldsymbol{\theta}\mathrm{d}\mathbf{y} \tag{4.15}$$

Finally, we again approximate the double integral in Equation 4.15 via Monte-Carlo integration,

$$\widehat{U}_t(\mathbf{d}) \approx \frac{1}{N} \sum_{i=1}^{N} \log\left[r_t(\mathbf{d}, \mathbf{y}^{(i)}, \boldsymbol{\theta}^{(i)}, \mathbb{D}_{t-1})\right], \tag{4.16}$$

where $\mathbf{y}^{(i)} \sim p(\mathbf{y} \mid \mathbf{d}, \boldsymbol{\theta}^{(i)})$ and $\boldsymbol{\theta}^{(i)} \sim p(\boldsymbol{\theta} \mid \mathbb{D}_{t-1})$.

In Algorithm 4 we have outlined how to obtain samples from the updated prior distribution, i.e. how to get $\boldsymbol{\theta}^{(i)} \sim p(\boldsymbol{\theta} \mid \mathbb{D}_{t-1})$. Additionally, in Algorithm 6 we have described how to compute the LFIRE ratio $r_t(\mathbf{d}, \mathbf{y}^{(i)}, \boldsymbol{\theta}^{(i)}, \mathbb{D}_{t-1})$ for sequential design, given some weight and prior sample pairs. With this we have all the tools we need to compute the sequential expected utility given in Equation 4.16. Furthermore, in Section 4.3 we have also explained how we can resample initial prior samples when the effective sampling size gets too low. We can assemble all of this together to build a framework for sequential Bayesian experimental design via LFIRE using grid search, as shown in Algorithm 7.

Using Algorithm 7, we can now sequentially design the linear toy model experiment. As done in Section 4.1, we discretise the design space into 100 points between $z = 0$ km and $z = 12$ km and use 1000 initial prior samples $\boldsymbol{\theta}^{(i)} \sim \mathcal{N}(\boldsymbol{\theta}; 0, 5^2)$. We use the ground

Algorithm 7 Sequential Bayesian Exp. Design via LFIRE using Grid Search

Let $\mathbb{D}_0 = \varnothing$
Sample initial model parameters from prior: $\boldsymbol{\theta}^{(i)} \sim p(\boldsymbol{\theta})$ for $i = 1, \ldots, N$
Initialise weights: $w_1^{(i)} = 1$ for $i = 1, \ldots, N$
for t=1 to t=T **do**
    Normalise the weights: $W_t^{(i)} = w_t^{(i)} / \Sigma_{i=1}^{N} w_t^{(i)}$
    Obtain samples $\{\boldsymbol{\theta}_{\text{up}}^{(i)}\}_{i=1}^{N}$ from the updated prior by applying Algorithm 4
    for $\mathbf{d}$ in discrete design space $\mathscr{D}$ **do**
        for i=1 to i=N **do**
            Sample data from the model: $\mathbf{y}^{(i)} \sim p(\mathbf{y} \mid \mathbf{d}, \boldsymbol{\theta}_{\text{up}}^{(i)})$
            Compute the ratio $r_t(\mathbf{d}, \mathbf{y}^{(i)}, \boldsymbol{\theta}_{\text{up}}^{(i)}, \mathbb{D}_{t-1})$ according to Algorithm 6
        end for
        Compute the expected utility approximation $\widehat{U}_t(\mathbf{d})$ given in Equation 4.16
    end for
    On the design space grid, find the next optimal design point: $\mathbf{d}_t^* = \arg\min_{\mathbf{d}} \widehat{U}_t(\mathbf{d})$
    Perform an experiment at $\mathbf{d}_t^*$ to observe some real data $\mathbf{y}_t^*$
    Update the prior distribution by updating the data set: $\mathbb{D}_t = \mathbb{D}_{t-1} \cup \{\mathbf{d}_t^*, \mathbf{y}_t^*\}$
    For all prior samples $\boldsymbol{\theta}^{(i)}$, compute new weights $w_{t+1}^{(i)}$ according to Equation 4.9
    Calculate the effective sampling size η using Equation 4.10
    if $\eta < \eta_{\min}$ **then**
        Obtain new prior samples $\boldsymbol{\theta}^{(i)}$ by resampling according to Algorithm 5
        Set the weights for the iteration to one, i.e. $w_{t+1}^{(i)} = 1$ for $i = 1, \ldots, N$
    end if
end for

truth $\boldsymbol{\theta}_{true} = [5.00, -0.12]^\top$ to generate real-world data y_t^* at optimal design points z_t^*. The resulting toy model sequential expected utilities for four iterations are shown in Figure 4.4.

As found previously in Section 4.1, the expected utility $U_1(z)$ for the first iteration converges at around $z \approx 6$ km. This means that we do not see much of an information gain in performing the next observation anywhere between 6 km and 12 km; in fact, the maximum expected utility occurs at $z_1^* = 8.35$ km. Consequently, for the second iteration, $U_2(z)$ has a minimum at around $z \approx 8$ km. This is to be expected, as we do not gain much information when observing data at exactly the same location when we have only made one observation so far. The maximum for the second iteration lies at $z_2^* = 0$ km, as far away from the first optimal altitude as possible. After the second iteration we perform some resampling (see Section 4.3) because the effective sampling size is below 50. The expected utility for the third iteration behaves similar to that of the first one, although it does not plateau, and peaks at $z_3^* = 12$ km. The expected utility $U_4(z)$ for the fourth iteration looks similar to that of the third iteration, albeit slightly lower overall; its maximum is also at $z_4^* = 12$ km. This

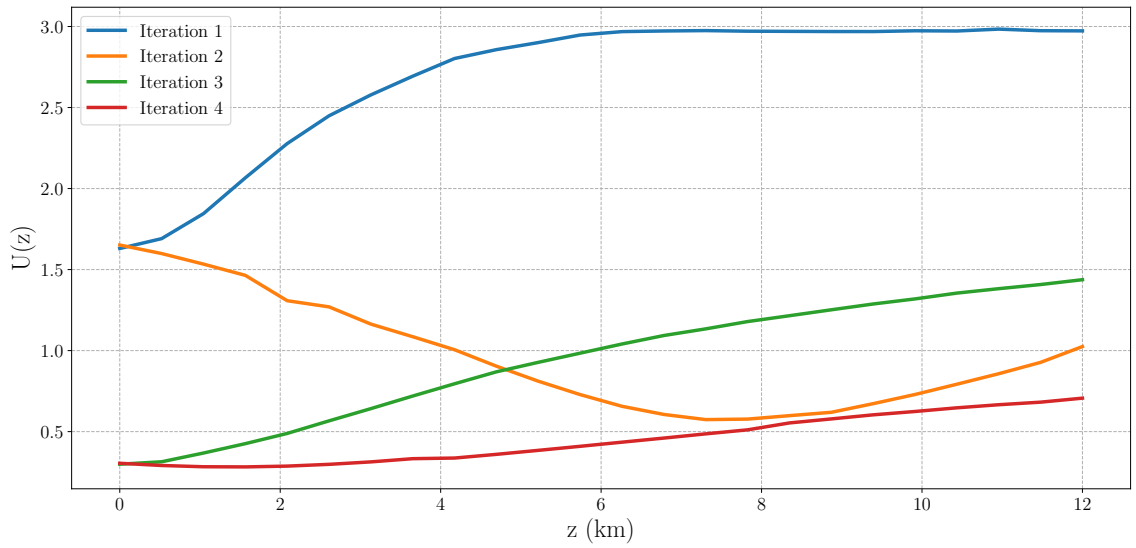

Fig. 4.4 Sequential expected utilities for the linear toy model, for a total of four iterations, using the grid search method.

may be attributed to the fact that two points are already enough to describe a linear model and following points do not make as much difference (see Section 2.2). This can also be connected to the fact that the signal-to-noise-ratio (SNR) is largest at $z = 12$ km.

It is also useful to look at the weights $w_{t+1}^{(i)}$ for each prior sample as a function of iteration t. Shown in Figure 4.5 are the weights obtained after each iteration. Note that the weights used in iteration t are obtained by using Equation 4.9 and the observations from the previous iteration.

For the first iteration, as found in Section 4.2, the initial prior samples with high weights all seem to lie on a straight line. Once we obtain a second observation, the prior samples with high weights become more concentrated. Although the individual weight values heavily increase, the number of prior samples with high weights drastically decreases. Thus, we applied resampling as outlined in Algorithm 5 in order to obtain new updated prior samples $\{\boldsymbol{\theta}_{new}^{(i)}\}_{i=1}^{1000}$. We then set the weight values to one and use the weight, prior sample pairs for computing the expected utility in the third iteration. After the third iteration, the updated prior samples with high weights lie on a straight line again. This does not change for the weights after the fourth iteration either. This may be attributed to the fact that we found the optimal altitudes to be $z_3^* = z_4^* = 12$ km for the third and fourth iteration.

We can use the weights shown in Figure 4.5 to obtain posterior samples for each iteration by utilising Algorithm 4. Because we sample from a categorical distribution in that algorithm,

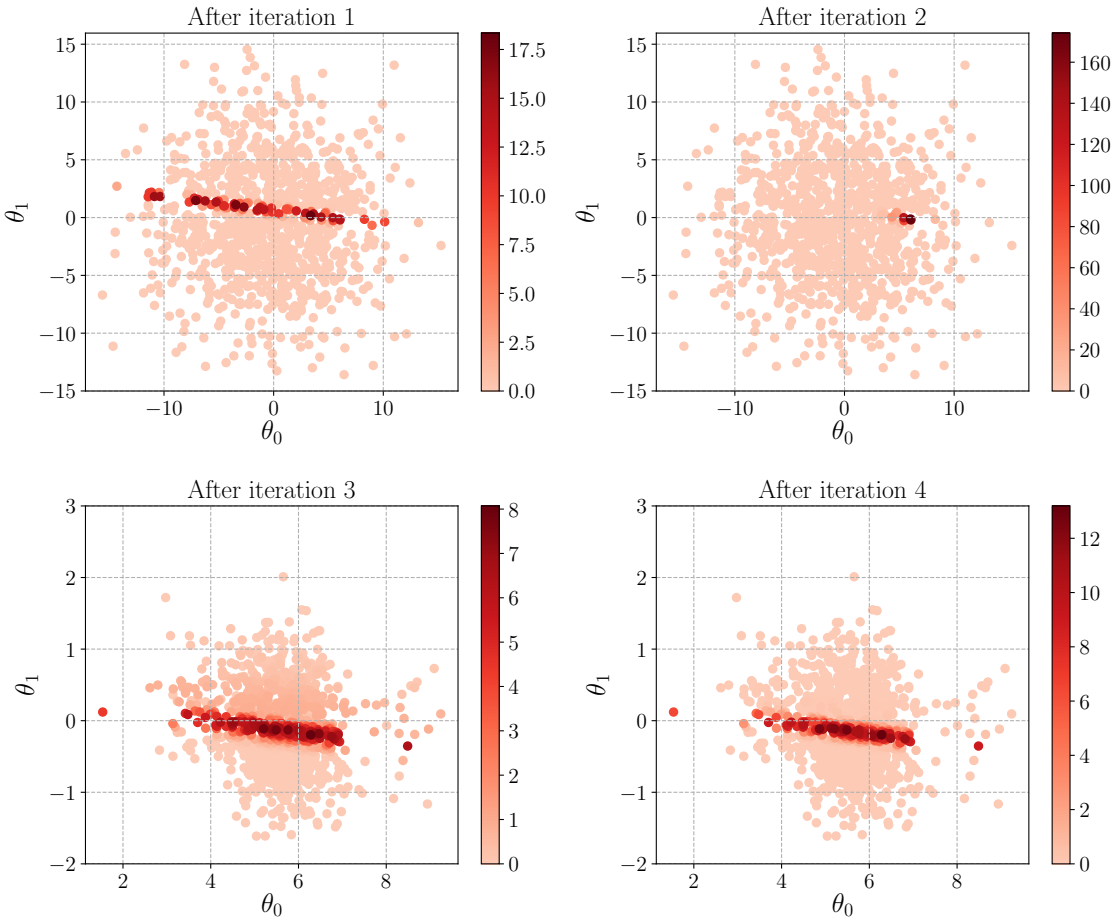

Fig. 4.5 Weights $w_{t+1}^{(i)}$ for each prior sample as a function of iteration t, using the grid search method; the weights are computed after each iteration using Equation 4.9. Note the different ranges and scales.

we would often obtain several identical samples due to their high weights; this results in some of the discrete posterior distributions not looking particularly smooth. Histograms of the posterior samples for each iteration are shown in Figure 4.6.

If we use the weights $w_5^{(i)}$ obtained after the fourth iteration to obtain 10,000 samples from the posterior distribution, as outlined in Algorithm 4, we can obtain estimates $\widehat{\theta}_0$ and $\widehat{\theta}_1$ of the model parameters. This results in median estimates $\widehat{\theta}_0 = 5.55^{+1.35}_{-1.85}$ and $\widehat{\theta}_1 = -0.15^{+0.23}_{-0.16}$, where the errors define a 95% confidence interval. Thus, our estimated model parameters capture the true parameters $\boldsymbol{\theta}_{true} = [5.00, -0.12]^\top$ with 95% confidence.

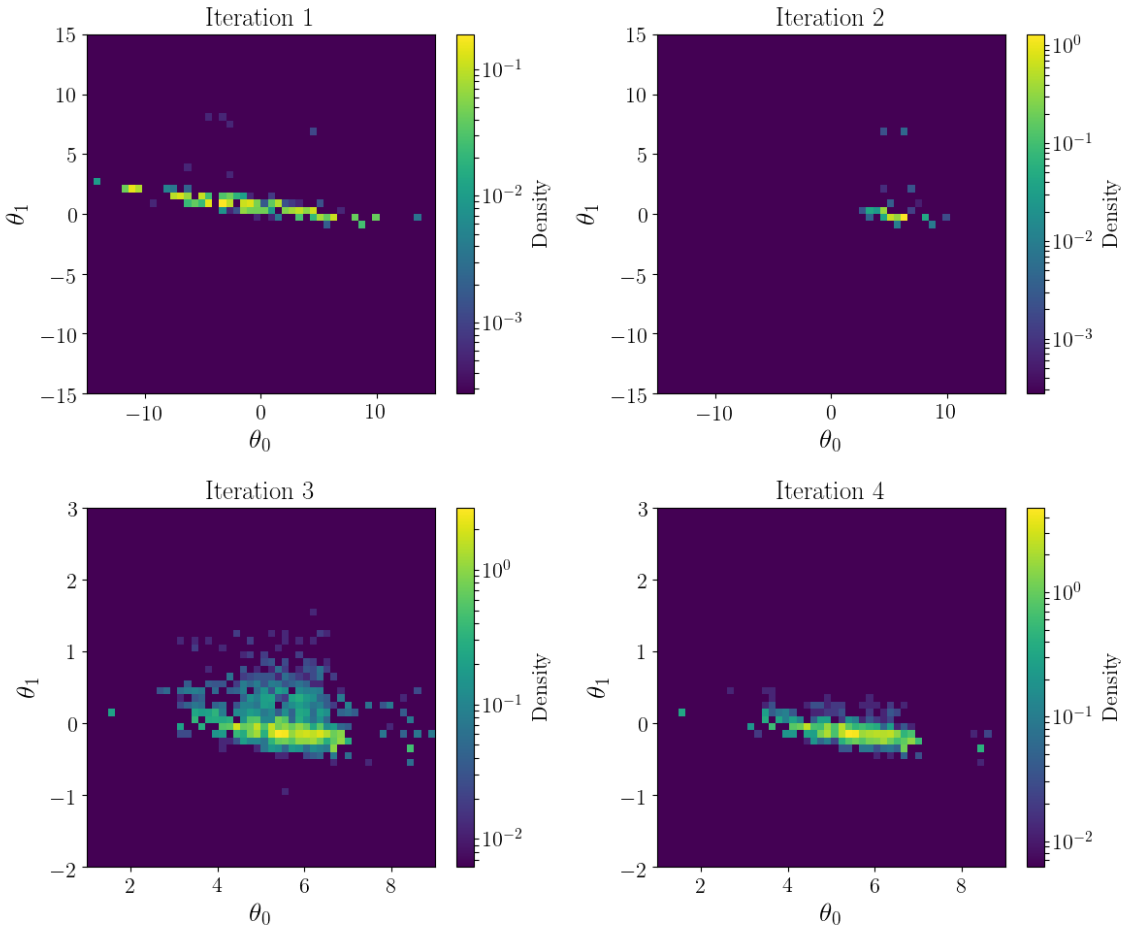

Fig. 4.6 Two-dimensional histograms showing discrete posterior distributions of the model parameters after each iteration, using the grid search method. The 10,000 posterior samples were obtained by using the weights $w_{t+1}^{(i)}$ and prior samples $\boldsymbol{\theta}^{(i)}$ shown in Figure 4.5, as well as Algorithm 4. Note the different ranges after iteration 2, due to resampling.

4.6 Application to Epidemiological Models

We have previously shown how Algorithm 7 works for the linear toy model. We would now like to test our sequential design algorithm using grid search for the epidemiology models introduced in Section 2.5.

4.6.1 Death Model

The Death Model only has one model parameter, the infection rate b, which determines how many susceptibles have been infected at time τ (see Section 2.5). We shall put a relatively

wide prior distribution over this parameter, i.e. a Gaussian of mean 1 and standard deviation 1, truncated such that $b > 0$; we sample 1000 parameters from this initial prior. Cook et al. (2008) and Drovandi and Pettitt (2013) have used a much narrower log-normal prior distribution for this model, where most prior samples were around 1. We found that their narrow prior distribution did not work for our algorithm because, we presume, the data in the logistic regression problem of the LFIRE approach becomes too similar.

Since neither Cook et al. (2008) or Drovandi and Pettitt (2013) used any summary statistics, we similarly only used the number of infected $I(\tau)$ as the data in the LFIRE logistic regression problem. The design variable for this model is time τ, i.e. the time at which we measure the state population of infected or suspectibles. In this particular case, we discretise the design time space into 40 points between $\tau = 0$ and $\tau = 4$. For the sequential design approach, it would make sense to change the time domain after an observation was made at τ^*, i.e. to have a window that starts at the previous optimal design time. However, Cook et al. (2008) and Drovandi and Pettitt (2013) have not done this and thus, because we would like to stay close to literature, we shall assume that our design space starts at $\tau = 0$ for every iteration. Conceptually, this would mean that at every iteration, we have a new population, subject to identical conditions as the first, and that the population from the previous iteration is discarded. This might be realistic for situations where actually looking at the population interferes in the model process (Cook et al., 2008).

We can use Equation 2.32 to generate samples from the Death Model, given a prior sample $b^{(i)}$ and a design time τ; the real-world observations are generated using $b_{true} = 1.5$. We can then use Algorithm 7 to sequentially design experiments for the Death Model. Figure 4.7 shows the expected utilities $U_t(\tau)$ that we obtain when doing this for four iterations.

As can be seen from Figure 4.7, the expected utilities for all four iterations peak in the same region and have the same shape. They only significantly differ in the fact that the expected utilities at later iterations seem to be shifted downwards, i.e. they are generally lower. This seems to suggest that, for the Death Model, the expected utility does not change significantly if we have observed some data. In fact, the optimal observed times were all close to each other: $\tau_1^* = 1.1$ km, $\tau_2^* = 0.9$ km, $\tau_3^* = 1.3$ km and $\tau_4^* = 0.8$ km. Cook et al. (2008) found their static optimal design time to be around 1.7; although they used a different utility function, their expected utility takes a similar form to the ones seen in Figure 4.7.

It is not entirely clear why the expected utilities do not change much. Looking back at Figure 2.5, the highest uncertainty and the greatest mean change in the Death Model occurs in the region of $0.5 < \tau < 1.5$; this could explain why all optimal design times lie in that region. For low τ, there are not many infected yet and therefore we do not see much change

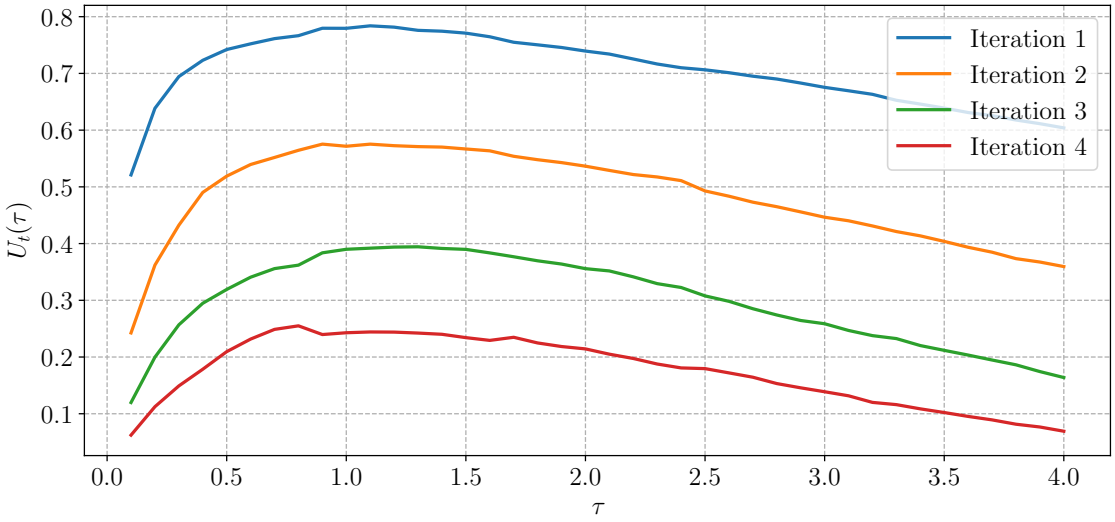

Fig. 4.7 Sequential expected utilities $U_t(\tau)$ for the Death Model, for four iterations.

between different parameter values. Similarly, at high times τ, all susceptibles have been infected and the number of infected does not change at all. Because there is not much change in the data, no matter the parameter values, making observations at these extreme time points does not provide much additional information, resulting in low expected utility values.

Similarly to the toy model, we can look at the weights $w_{t+1}^{(i)}$ obtained after every iteration t. The values of these weights for each initial prior sample $b^{(i)}$ are shown in Figure 4.8.

The weights form a well-defined, bell-like curve for every iteration. The weights after the first iteration form a wide curve with a low peak. This curve becomes progressively sharper, i.e. the width decreases and the maximum increases, for each subsequent iteration. In addition, the peak of each curve moves closer to the true model parameter $b_{true} = 1.5$ that was used in the real-world data generation process. This means that for every iteration we become increasingly confident and accurate in our belief about the true, underlying model parameter. These curves have barely any outliers, with only the weights $w_5^{(i)}$ after the fourth iteration having some large variance around the peak.

Using Algorithm 4, the set of initial prior samples and the weights shown in Figure 4.8, we can obtain samples from the posterior for every iteration. We sample from the posterior 10,000 times, resulting in the distributions shown in Figure 4.9. For every iteration, the posterior distribution becomes sharper, i.e. we become more confident in our parameter estimation. In addition, the mode of the distribution also moves progressively closer to the true parameter value of $b_{true} = 1$.

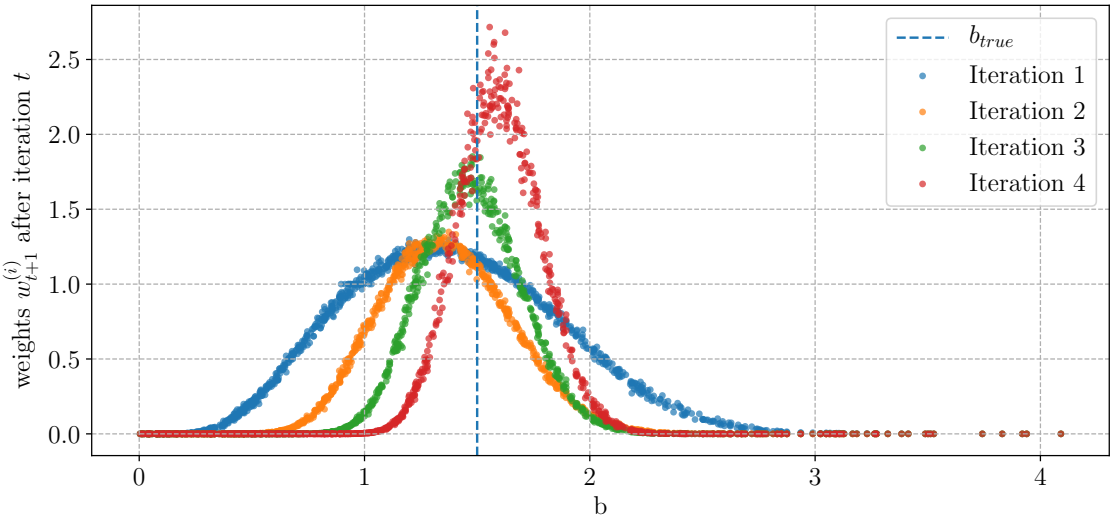

Fig. 4.8 Death Model weights $w_{t+1}^{(i)}$ for each prior sample as a function of iteration t, using the grid search method; the weights are computed after each iteration using Equation 4.9.

Using the posterior samples from the fourth iteration, shown in Figure 4.9, we can obtain an estimate $\widehat{b}$ of the infection rate; this yields a median estimate $\widehat{b} = 1.59^{+0.35}_{-0.36}$, where the errors define a 95% confidence interval. Because the true parameter is well within this uncertainty range, we can say with 95% confidence that the final posterior captures the underlying truth of the model.

4.6.2 SIR Model

Unlike the Death Model, the SIR Model has two parameters, the rate of infection β and the rate of recovery γ, both of which are bound between 0 and 1 (see Section 2.5). We shall put a standard uniform prior $U(0,0.5)$ on both model parameters and use it to obtain 1000 initial prior samples $\{\beta^{(i)}, \gamma^{(i)}\}_{i=1}^{1000}$. We also tried a wider prior of $U(0,1)$ but found that was not working well, as we did not obtain enough relevant model parameters and most weights were near-zero. Like we did with the Death Model, we are not using any summary statistics for the SIR Model. This means that the data used in the LFIRE logistic regression is the number of suspectibles $S(\tau)$, infected $I(\tau)$ and recovered $R(\tau)$. Just like for the Death Model, the design variable for this model is time τ. We obtain a time grid by discretising the space between $\tau = 0$ and $\tau = 3$ into 30 points. In addition, for each iteration we shall assume that

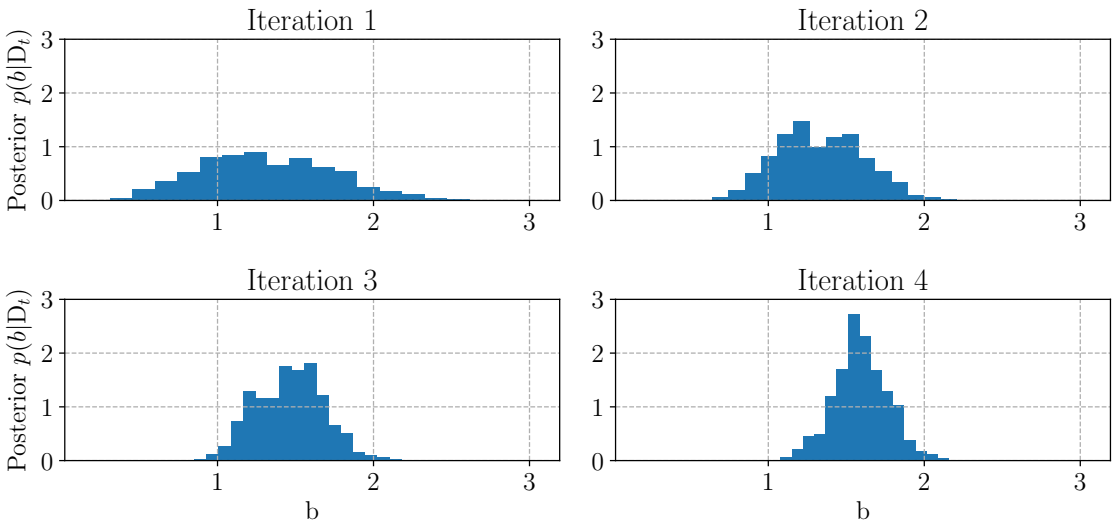

Fig. 4.9 Death Model posterior distributions, for four iterations, for the grid search method. Posterior samples have been computed using the weights $w_{t+1}^{(i)}$ for each initial prior sample and Algorithm 4

the time space starts at $\tau = 0$ again, instead of having a sliding time window, similar to what we did for the Death Model.

Given prior samples $(\beta^{(i)}, \gamma^{(i)})$ and design time τ, we can use the procedure outlined in Section 2.5 to generate data for the SIR Model. We use $\beta_{true} = 0.15$ and $\mu_{true} = 0.05$ to generate real-world data for the observations; this is equivalent to a basic reproduction number of $R_0 = 3$. The sequential expected utilities $U_t(\tau)$ for the SIR Model, computed using Algorithm 7, are shown in Figure 4.10 for four iterations.

All sequential expected utilities in Figure 4.10 show similar behaviour; $U_t(\tau)$ peaks around $\tau \approx 0.5$ and is low for low and high τ. Just like for the Death Model, the overall behaviour does not change much for each iteration, although $U_t(\tau)$ is lower for every subsequent iteration. The expected utility for the first iteration is a bit flatter than subsequent utilities, which may perhaps be because we use an uninformative, uniform prior. In the peak region, the expected utility for the second iteration is actually higher than the one for iteration 1, although it is lower every else. The expected utilities for iteration 3 and iteration 4 are very close to each other; this could be because we had to apply resampling after iteration 3, as the effective sampling size got below 50. In addition, the expected utility curves have significant fluctuations, as opposed to the smooth curves we have seen for the toy model in Figure 4.4 and the Death Model in Figure 4.7. This could mean that, for the SIR Model, computing the

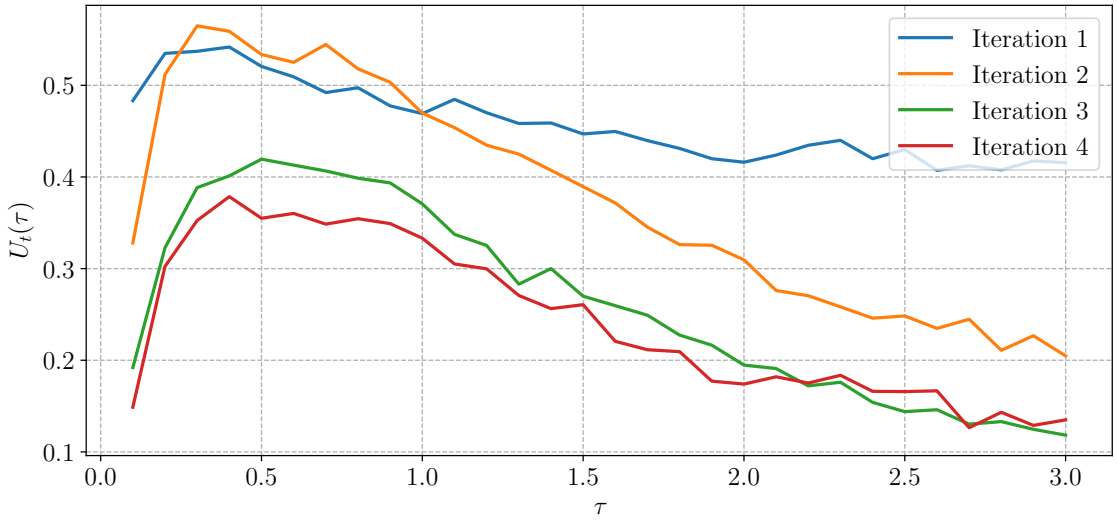

Fig. 4.10 Sequential expected utilities $U_t(\tau)$ for the SIR Model, for four iterations, using the grid search method.

LFIRE ratio with 1000 initial prior samples may not capture the posterior accurately enough or that our prior distribution is too uninformative; in fact, we noticed that the fluctuations increase if we increased the upper bound of the uniform prior. The optimal design times for each iteration are $\tau_1^* = 0.4$ km, $\tau_2^* = 0.3$ km, $\tau_3^* = 0.5$ km and $\tau_4^* = 0.4$ km. Overall, the expected utilities for the SIR Model show similar behaviour to those of the Death Model in Figure 4.7. For the SIR Model however, the optimal design times occur earlier and the peaks are more sharply skewed towards early τ.

For the SIR Model, as done previously, we can look at the weights $w_{t+1}^{(i)}$ obtained after each iteration t. For each pair of initial prior samples $(\beta^{(i)}, \gamma^{(i)})$, its weight is computed according to Equation 4.9; the weights for the SIR Model are shown in Figure 4.11.

For the first iteration, the distribution of initial prior samples with high weights is concentrated around low γ and a rather large range of β. This variance reduces for the second iteration, where high weights already seem to form a peak close to the true parameter values. This distribution does not change much for iteration 3, except becoming slightly sharper. As said previously, after iteration 3 we had to apply resampling to the model parameters. The new, resampled parameters and the weights obtained after iteration 4 are shown in the bottom right plot in Figure 4.11. The resampling was not entirely succesfull however, as there seem to be many important parameters missing in the bottom right part of the resampling region.

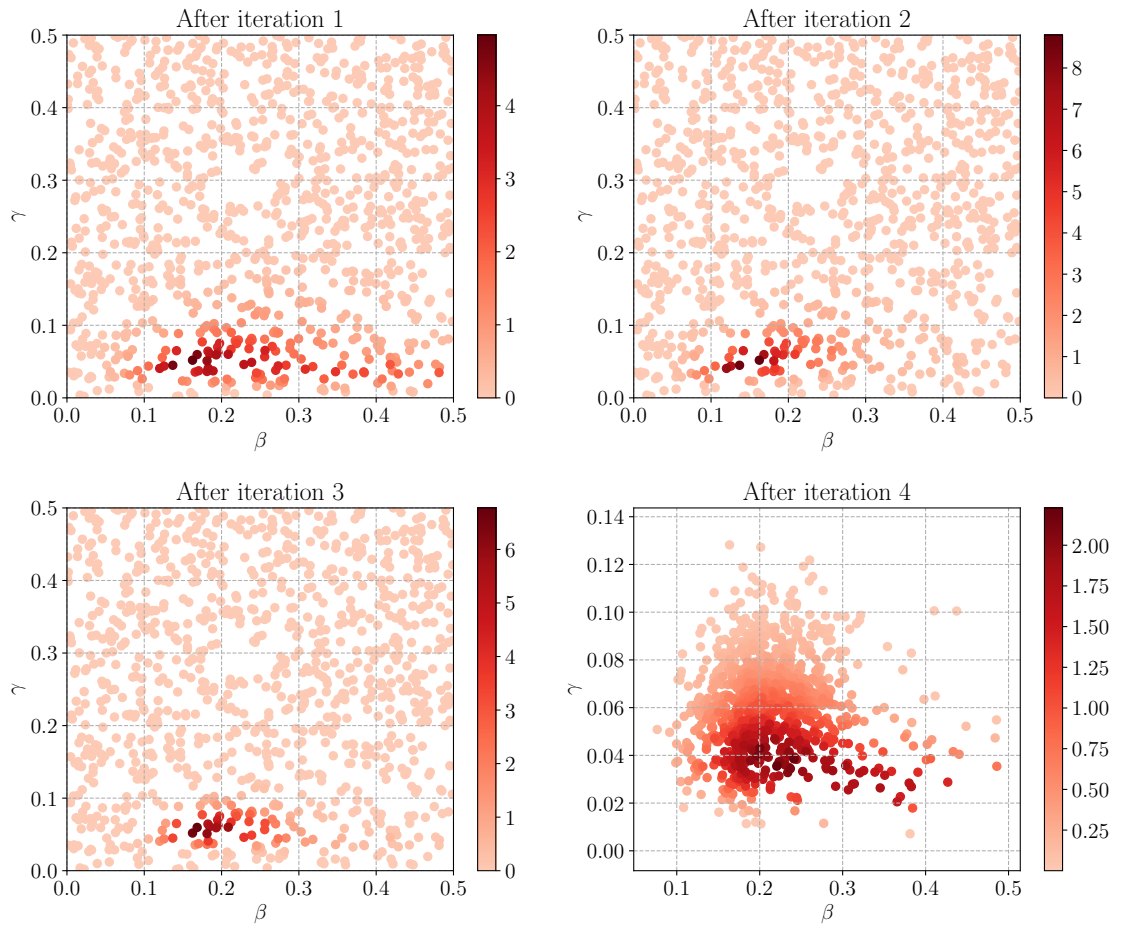

Fig. 4.11 SIR Model weights $w_{t+1}^{(i)}$ for each prior sample as a function of iteration t; the weights are computed after each iteration using Equation 4.9. Note the different ranges and scales.

As done previously, we can use Algorithm 4, the initial prior samples and the weights shown in Figure 4.11 to obtain posterior samples for every iteration. By sampling from the posterior distribution 10,000 times, we obtain the discrete densities shown in Figure 4.12. Because we only had 1000 initial prior samples, we did not completely cover all regions equally, resulting in some empty bins in the two-dimensional histograms.

By using the posterior samples shown in Figure 4.12, we can again obtain estimates $\widehat{\beta}$ and $\widehat{\gamma}$ of the model parameters. This results in median estimates $\widehat{\beta} = 0.20^{+0.15}_{-0.06}$ and $\widehat{\gamma} = 0.05^{+0.03}_{-0.02}$ for the infection rate and recovery rate, respectively, where the errors define the 95% confidence interval. The infection rate estimate $\widehat{\beta}$ only barely agree with the true value of $\beta_{true} = 0.15$, at least within a 95% confidence limit. However, the recovery rate estimate $\widehat{\gamma}$

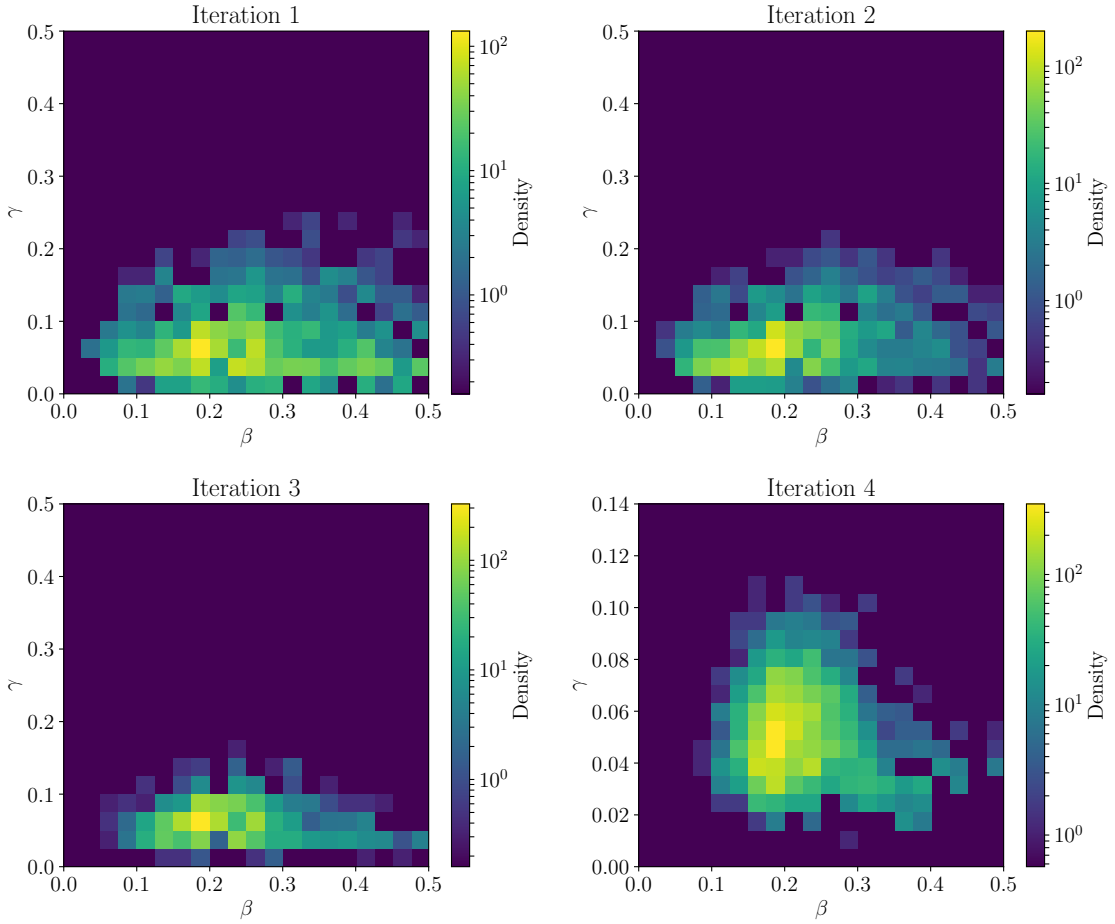

Fig. 4.12 SIR Model posterior distributions, for four iterations, for the grid search method. Posterior samples have been computed using the weights $w_{t+1}^{(i)}$ for each initial prior sample and Algorithm 4. The empty bins are a result of not having enough initial prior samples; note that the colourbar is logarithmic.

agrees extremely well with the true value $\mu_{true} = 0.05$. Using the same posterior samples, the basic reproduction number estimate is $\widehat{R}_0 = 4.01^{+7.17}_{-1.72}$ (recall that R_0 is the ratio of β to γ). This estimate barely agrees with the true value of $R_0 = 3$, within the 95% confidence limit. Although the median basic reproduction number is higher than the true value, we correctly identify the disease as being unstable and spreading.

Overall, applying sequential Bayesian experimental design using grid search to the SIR Model yields satisfactory results. However, the expected utilities are noisy and sensitive to the choice of the prior hyper-parameters. This may be a consequence of not using a detailed enough grid and/or not having enough prior samples for the LFIRE computation. Although

only the estimated infection rate does not quite agree with the true value, all other quantities are estimated well.

Chapter 5

Efficient Utility Optimisation and Non-Myopic Design

In Chapter 4 we have previously shown how to perform sequential Bayesian experimental design for simulator-based statistical models by evaluating the expected utility on a grid of design points. In Section 2.4 we have also shown that we can reduce the number of expected utility evaluations by using Bayesian Optimisation instead. In this chapter we describe how to use Bayesian Optimisation in the myopic design setting for simulator models. Towards the end, we briefly touch on a non-myopic design extension as well, i.e. where we aim to find the next set of optimal design points.

5.1 Implementation of Bayesian Optimisation

By evaluating the sequential expected utility $U_t(\mathbf{d})$ on a grid of design points, we potentially make computations that do not significantly aid us in optimising $U_t(\mathbf{d})$. Using Bayesian Optimisation, which we explained in Section 2.4, we can, instead of evaluating $U_t(\mathbf{d})$ on a grid, intelligently decide where to evaluate $U_t(\mathbf{d})$ next in order to find its optimum; this drastically decreases computation times. To build such a design framework, we have to substitute the grid search in Algorithm 7 by Bayesian Optimisation; in Section 2.4 we already did this for models with tractable likelihoods (see Algorithm 3).

There are only few things that we have to change in the sequential design procedure using Bayesian Optimisation, outlined in Algorithm 3, when we consider simulator models. We now have weights that we use to obtain samples from the updated prior distribution by means of Algorithm 4. Because we do not have analytical forms of the data generating distributions, we approximate the ratio of posterior to prior with LFIRE ratios, computed according to

Algorithm 6. We use these ratios for the Monte-Carlo approximation in Equation 4.16 to compute an estimate of the sequential expected utility $\widehat{U}_t(\mathbf{d})$ at iteration t. If the effective sample size of this approximation becomes too low, we have to resample our initial model parameters according to Algorithm 5. Conceptually, we understand this procedure as the combination of Algorithm 3 and Algorithm 7.

This method of sequential Bayesian experimental design with Bayesian Optimisation is summarised in Algorithm 8.

Algorithm 8 Sequential Bayesian Exp. Design via LFIRE using Bayesian Optimisation

1: Let $\mathbb{D}_0 = \varnothing$
2: Sample initial model parameters from prior: $\boldsymbol{\theta}^{(i)} \sim p(\boldsymbol{\theta})$ for $i = 1, \ldots, N$
3: Initialise weights: $w_1^{(i)} = 1$ for $i = 1, \ldots, N$
4: **for** t=1 to t=T **do**
5: Normalise the weights: $W_t^{(i)} = w_t^{(i)} / \Sigma_{i=1}^N w_t^{(i)}$
6: Obtain samples $\{\boldsymbol{\theta}_{up}^{(i)}\}_{i=1}^N$ from the updated prior by applying Algorithm 4
7: Start with j=0 and initialise $\mathbf{d}_j$ with a randomly chose design
8: Let $\mathbf{d}_{best}$ be the design point resulting in the best expected utility estimate
9: **while** $\mathbf{d}_{best}$ not converged **do**
10: **for** i=1 to i=N **do**
11: Sample data from the model: $\mathbf{y}^{(i)} \sim p(\mathbf{y} \mid \mathbf{d}_j, \boldsymbol{\theta}_{up}^{(i)})$
12: Compute the ratio $r_t(\mathbf{d}_j, \mathbf{y}^{(i)}, \boldsymbol{\theta}_{up}^{(i)}, \mathbb{D}_{t-1})$ according to Algorithm 6
13: **end for**
14: Compute $\widehat{U}_t(\mathbf{d}_j)$ given in Equation 4.16
15: Using Bayesian Optimisation, outlined in Section 2.4, find the design point $\mathbf{d}_{j+1}$ at which to evaluate the expected utility next
16: **end while**
17: Let $\mathbf{d}_t^* = \mathbf{d}_{best}$
18: Perform an experiment at $\mathbf{d}_t^*$ to observe some real data $\mathbf{y}_t^*$
19: Update the parameter prior distribution by updating the data set: $\mathbb{D}_t = \mathbb{D}_{t-1} \cup \{\mathbf{d}_t^*, \mathbf{y}_t^*\}$
20: For all prior samples $\boldsymbol{\theta}^{(i)}$, compute new weights $w_{t+1}^{(i)}$ according to Equation 4.9
21: Calculate the effective sampling size η using Equation 4.10
22: **if** $\eta < \eta_{\min}$ **then**
23: Obtain new prior samples $\boldsymbol{\theta}^{(i)}$ by resampling according to Algorithm 5
24: Set the weights for the iteration to one, i.e. $w_{t+1}^{(i)} = 1$ for $i = 1, \ldots, N$
25: **end if**
26: **end for**

As before, the Bayesian optimisation steps in Lines 7–16 of Algorithm 8 are implemented with the GPyOpt package in Python. We also make use of its option to parallelise the optimisation process over several CPU cores.

5.2 Application to Epidemiological Models

We have used the toy model to illustrate Bayesian Optimisation in Section 2.4 and we applied the grid search sequential design algorithm to it in Section 4.5. We have also applied the latter algorithm to two epidemiological models, the Death Model (Cook et al., 2008) and the SIR Model (Allen, 2008), in Section 4.6. Here we apply the sequential design algorithm using Bayesian optimisation to these two epidemiological models.

5.2.1 Death Model

As before, we are trying to estimate the infection rate b that determines how many susceptibles $S(\tau)$ have been infected at time τ (see Section 2.5). We shall use the same prior distribution as before, i.e. a Gaussian of mean one and variance one that is truncated such that $b > 0$. No summary statistics are used, i.e. the simulated data used in the LFIRE approximation is again just the number of infected $I(\tau)$. Instead of defining a grid, we only need to specify the boundaries of the continuous design variable time τ; we here choose that $0 \leq \tau \leq 4$. To generate data, the Death Model equations in Section 2.5 are used; the real-world data is generated using the true parameter of $b_{true} = 1.5$. In order to find the optimal design times τ_t^* for every iteration, we then apply Algorithm 8 to the Death Model.

We show the sequential expected utilities in Figure 5.1. The red points display the $U(\tau_j)$ evaluations, where the τ_j were found using Bayesian Optimisation. The filled-in, blue region defines the 95% confidence limit of the Gaussian Process posterior predictive distribution, while the dotted, green line represents the expected improvement acquisition function after the last evaluation.

The expected utilities in Figure 5.1 behave in the same way as the ones for the grid search in Figure 4.7; again, the behaviour of the expected utility does not change much as a function of iteration. It is unfortunately hard to compare their relative sizes, because the GPyOpt package applies some normalisation to $U_t(\tau)$ (GPyOpt, 2016). Therefore, it cannot be concluded that expected utilities at later iterations are lower, as was the case for the grid search expected utilities in Figure 4.7. The $U_t(\tau)$ evaluations are clustered around the same region which appears to be the maximum of the mean expected utilities. As intended, there are few evaluations where the expected utility is low; because the expected improvement equations in Section 2.4 imply an exploitation-exploration trade-off (Mockus et al., 1978), we still sometimes have evaluations in regions where we do not expect a maximum. For the grid search method on the other hand, we had a low resolution around the maximum and several pointless evaluations in regions of low expected utility (see Figure 4.7). The optimal

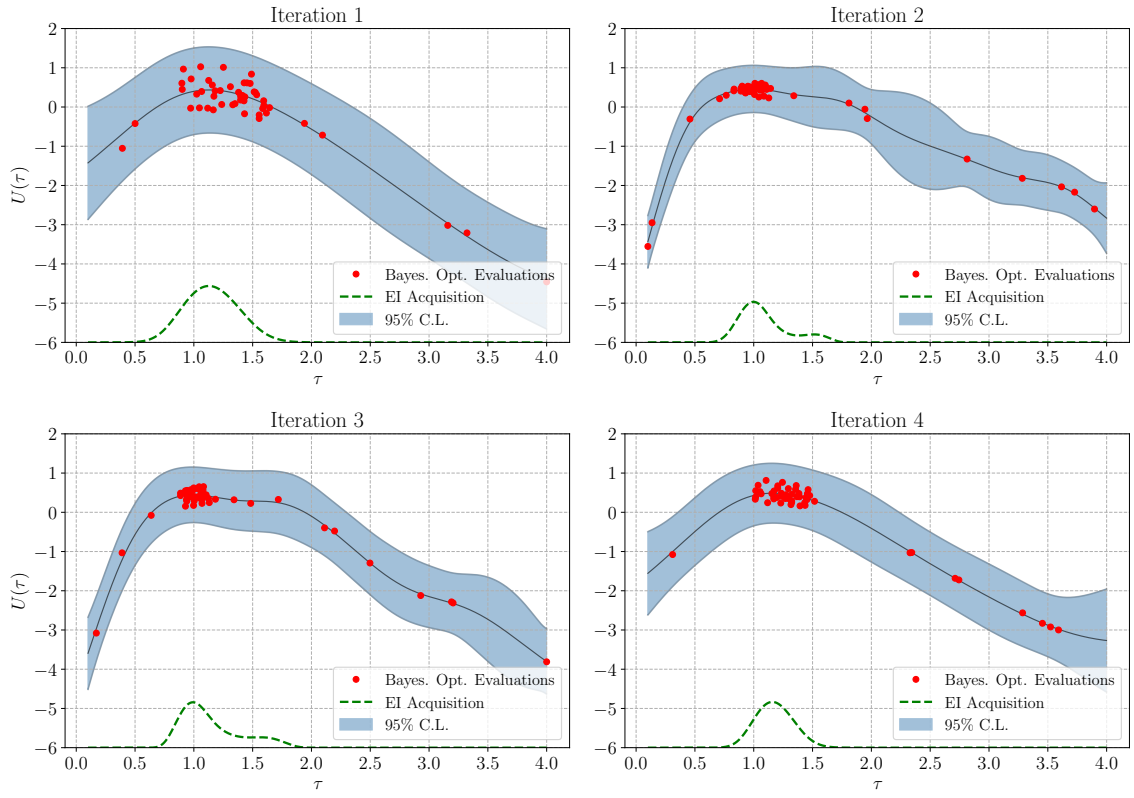

Fig. 5.1 Sequential expected utilities $U_t(\tau)$ for the Death Model, for four iterations, using Bayesian Optimisation. Shown are the expected utility evaluations (red points), the 95% confidence limit of the Gaussian Prior posterior predictive distribution (shaded, blue region) and the expected improvement acquisition function (dotted, green line) after the last iteration.

design times in Figure 5.1 are $\tau_1^* = 1.06$, $\tau_2^* = 1.06$, $\tau_3^* = 1.08$ and $\tau_4^* = 1.11$; these optimal times are much closer together than for the grid search expected utilities.

Instead of showing the weights after each iteration as well, we shall only be showing the posterior distributions here. Using Algorithm 4 we can obtain samples from the posterior distribution, given a set of weights and initial prior samples. Here we use the weights $w_{t+1}^{(i)}$ obtained after each iteration t, computed using Equation 4.9, and the initial samples from the truncated Gaussian prior distribution. For each iteration, we sample 10,000 times from the posterior distributions, yielding the discrete probability densities shown in Figure 5.2.

The posterior distribution for the first iteration is relatively wide and has a low peak; since we have only made one observation so far, our beliefs about the parameter are not that confident yet. For every subsequent iteration, the posterior distribution becomes sharper, i.e. we become more confident about our beliefs. The peak of every posterior is always

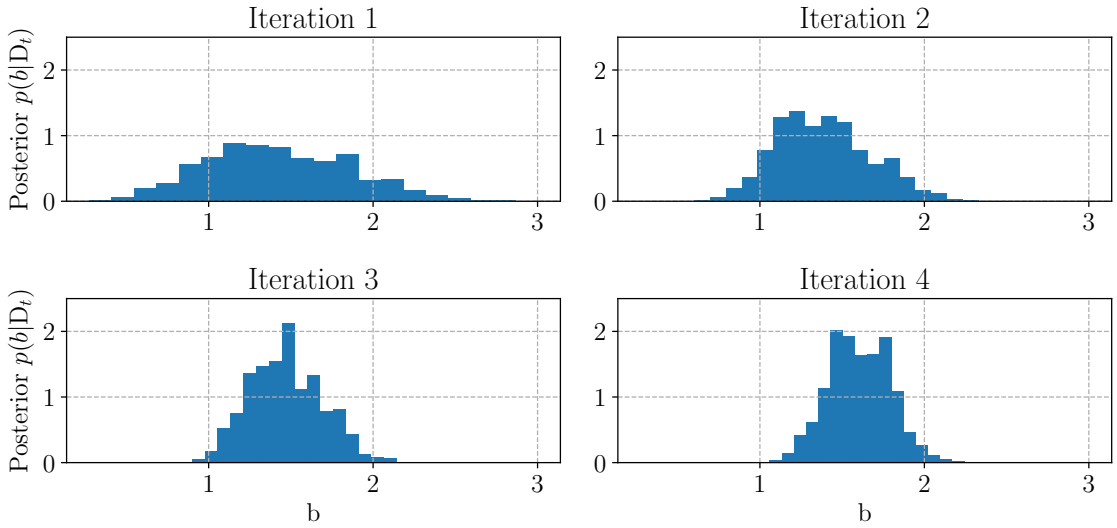

Fig. 5.2 Death Model posterior distributions, for four iterations, for the Bayesian Optimisation method. Posterior samples have been computed using the weights $w_{t+1}^{(i)}$ for each initial prior sample and Algorithm 4

in the region $1 < b < 2$ but the median does shift slightly towards the true value of b_{true} for every iteration. Using the discrete probability densities shown in Figure 5.2, we can obtain estimates $\widehat{b}$ of the model parameter. After the final, fourth iteration we find that the median estimate is $\widehat{b} = 1.62^{+0.37}_{-0.38}$, where the errors indicate a 95% confidence limit. This estimate is very close to what we obtained using the grid search method in Section 4.6 and it actually has larger confidence intervals. Thus, we do not see much of an improvement in estimation accuracy when using Bayesian Optimisation over grid search. Nonetheless, we know that by using Bayesian Optimisation we tend to converge to the optimum faster than using grid search (see Section 2.4). When using Bayesian Optimisation we only need around 20 evaluations, for every iteration, to optimise the expected utility, whereas in Section 4.6 we used a grid of 40 design points to do the same, albeit at a much lower resolution around the optimum.

5.2.2 SIR Model

For the SIR Model, we are trying to estimate the rate of infection β and the rate of recovery γ (see Section 2.5). As done previously, we put a uniform prior $U(0, 0.5)$ on both model parameters in order to obtain 1000 initial prior samples $\{\beta^{(i)}, \gamma^{(i)}\}_{i=1}^{1000}$. We do not use any summary statistics and therefore the data used in the LFIRE ratio computation is the number

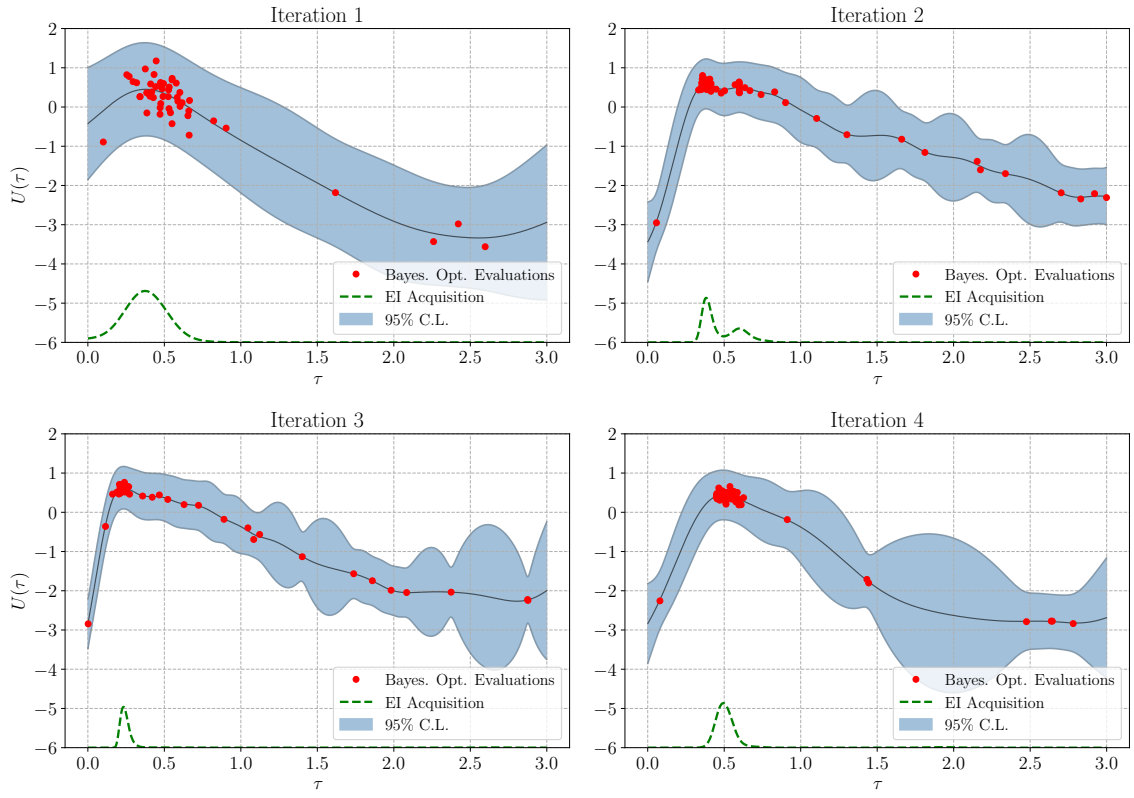

Fig. 5.3 Sequential expected utilities $U_t(\tau)$ for the SIR Model, for four iterations, using Bayesian Optimisation. Shown are the expected utility evaluations (red points), the 95% confidence interval of the Gaussian Prior posterior predictive distribution (shaded, blue region) and the expected improvement acquisition function (dotted, green line) after the last iteration.

of susceptibles $S(\tau)$, the number of infected $I(\tau)$ and the number of recovered $R(\tau)$. To simulate this data, given a design time τ and a set of model parameters, we use the procedure outlined in Section 2.5; real-world data is generated using $\beta_{true} = 0.15$ and $\gamma_{true} = 0.05$. We choose $0 \leq \tau \leq 3$ as the domain for the continuous design time τ. We then apply Algorithm 8 to the SIR Model for a total of four iterations.

The sequential expected utilities for each iteration are shown in Figure 5.3. The red points indicate the expected utility evaluations, while the filled-in, blue region defines the 95% confidence limit of the Gaussian Process posterior predictive distribution. The dotted, green line indicates the expected improvement acquisition function after the last evaluation.

All expected utilities in Figure 5.3 show similar behaviour to the ones shown for the grid search method in Figure 4.10. As explained previously, GPyOpt does some internal

normalising to the expected utility evaluations (GPyOpt, 2016) and therefore we cannot compare their relative sizes. This means that we cannot test whether or not the expected utility generally decreases for later iterations, as was the case for the grid search method. Most of the $U_t(\tau)$ evaluations cluster around the maximum of the mean expected utilities, similar to what we found for the Death Model in Figure 5.1. For the grid search method, we found that the expected utility of the first iteration was flatter than the ones for subsequent iterations but we do not see the same result for the Bayesian Optimisation method. This may be because with Bayesian Optimisation we focus on the region of the optimum and elsewhere the approximation can become coarse. Using the expected utilities in Figure 5.3, we obtain optimal design times of $\tau_1^* = 0.45$, $\tau_2^* = 0.36$, $\tau_3^* = 0.24$ and $\tau_4^* = 0.54$. These are very close to the optimal design times obtain using the grid search in Section 4.6. Compared to the grid search method however, we can converge much faster to the optimum of $U_t(\tau)$ when using Bayesian Optimisation. Note that we do not see any resampling happening, as opposed to what we saw with the grid search method, because the effective sampling size does not go below 50 here.

Given a set of weights and initial prior samples, we use Algorithm 4 to obtain 10,000 posterior samples. We use the weights $w_{t+1}^{(i)}$ obtained after each iteration t, computed using Equation 4.9, and the initial samples from the uniform prior distribution. This yields the posterior probability densities shown in Figure 5.2.

The posterior distributions for all iterations show similar behaviour, peaking at low parameter values with a wide spread in β and a relatively low spread in γ. For the first iteration, this distribution is relatively wide and becomes sharper for subsequent iterations. We observed the same behaviour for the grid search method posteriors in Figure 4.11, except for no resampling happening when using the Bayesian Optimisation method. Using the discrete probability densities shown in Figure 5.4 we can obtain estimates of the model parameters. We find that the median estimates were $\widehat{\beta} = 0.19^{+0.13}_{-0.07}$ and $\widehat{\gamma} = 0.07^{+0.06}_{-0.03}$, where the errors define a 95% confidence interval. Just like for the grid search method, both parameter estimations capture the true parameter values, within the errors. The basic reproduction number estimate is $\widehat{R} = 2.81^{+2.29}_{-1.41}$; this median estimate is much closer to the true value of $R_0 = 3$ than the estimate from the grid search method (see Section 4.6). Thus, it appears that for the SIR Model, we obtain slightly better parameter estimations when using Bayesian Optimisation instead of grid search. In addition, we find that, similar to the Death model, we only need around 20 expected utility evaluations to solve the opimisation problem for every iteration when using Bayesian Optimisation. On the other hand, in Section 4.6 we had to use a grid of 30 design points to optimise the expected utiility and did so at a

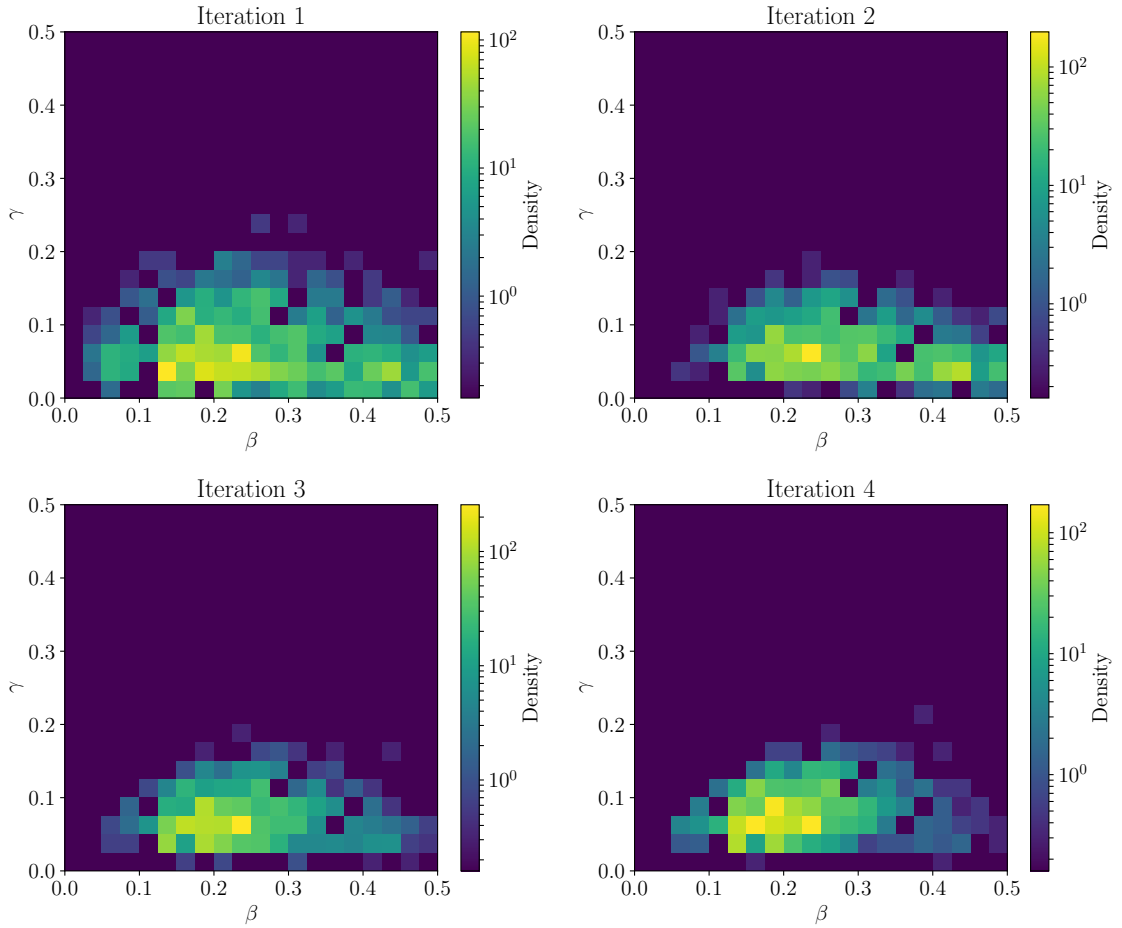

Fig. 5.4 SIR Model posterior distributions, for four iterations, for the Bayesian Optimisation method. Posterior samples have been computed using the weights $w_{t+1}^{(i)}$ for each initial prior sample and Algorithm 4. The white squares show empty bins and the colourbar is logarithmic.

much lower resolution around the optimum. Thus, for the SIR Model, we also find that we converge to the optimum faster when using Bayesian Optimisation instead of grid search. We presume this difference increases dramatically when one consideres multi-dimensional design variables.

5.3 Non-Myopic Design

So far we have only considered myopic design, i.e. design problems where we want to find the next, single optimal design point at which we ought to make observations. This is

generally a useful strategy, especially in situations when, for instance, we do not know how many experiments we can make in total or when there is ample time between the experiments. If we however know how many experiments we can make, given a certain budget, then the premise of the design problem changes and we may find different optimal designs. In non-myopic design we are searching for a next set of optimal design points at which to make observations. This is perhaps an even more realistic premise, as researchers generally have budgets for their experiments and can project how many observations they can make.

With regards to implementation, the general idea of non-myopic design is to increase the dimensions of the design variable according to the number of observations we can make. Assume for now that the myopic design variable is a scalar, i.e. $\mathbf{d}_{myopic} = d$. If we, for instance, know that can make four observations, the resulting non-myopic design variable would be a 4-dimensional vector $\mathbf{d}_{non-myopic} = [d^1, d^2, d^3, d^4]^\top$, where the d^i generally have the same domain, although there might be some constraints between them. The method of simulating data using this non-myopic design variable generally stays the same. Assuming that the myopic, simulated data is also a scalar, we would obtain 4-dimensional data $\mathbf{y} = [y^1, y^2, y^3, y^4]^\top$, where $y^i \sim p(y \mid d^i, \boldsymbol{\theta}, \{y^j\}_{j \neq i})$; each observation y^i might also depend on a set of other observations $\{y^j\}_{j \neq i}$ and therefore we cannot say that they are independent.

Increasing the dimensions of the design variable results in the expected utility function being a surface in multi-dimensional space. Therefore grid search becomes too expensive and we shall thus use Bayesian Optimisation instead. Once we have optimised the expected utility surface of the first iteration and obtained an optimal design point $\mathbf{d}^*_{1,non-myopic} = [d_1^{1,*}, d_1^{2,*}, d_1^{3,*}, d_1^{4,*}]^\top$, we need to make an observation. We here assume that we cannot make several real-world observations simultaneously, although that may not always be the case; instead, we make an observation $y_1^{i,*}$ at one of the myopic optimal design points $d_1^{i,*}$. Which of the design points that make up $\mathbf{d}^*_{1,non-myopic}$ are chosen may depend on the domain constraints; if there are no constraints, the point may be chosen randomly. Using $(d_1^{i,*}, y_1^{i,*})$ we can update our prior distributions as done previously in the myopic design setting (see Algorithm 8). Since we already made one observations, we can only make three more observations with the current example. Thus, for the second iteration, the non-myopic design variable is three-dimensional, $\mathbf{d}^*_{2,non-myopic} = [d_2^{1,*}, d_2^{2,*}, d_2^{3,*}]^\top$. Similarly, the non-myopic design variable is two-dimensional for the third iteration and one-dimensional for the fourth iteration. Note that we still only use a single optimal design point $d_t^{i,*}$ to update the prior distribution at iteration t.

5.3.1 Death Model

We shall explain and test how well Algorithm 8 works in the non-myopic design setting by means of the Death Model (Cook et al., 2008).

Assume that we are able to make a total of four observations. The myopic design variable of the Death Model is time τ and therefore the non-myopic design variable is $\boldsymbol{\tau} = [\tau^1, \tau^2, \tau^3, \tau^4]^\top$. Because we cannot go backwards in time however, we need to put some constraints on these dimensions, e.g.

$$\tau^1 < \tau^2 < \tau^3 < \tau^4 \tag{5.1}$$

Essentially we are saying that τ^1 is the first time at which we could make an observation and τ^4 is the last. This helps during the Bayesian Optimisation process as the constraints significantly reduces the total design space that we have to explore.

In order to simulate the number of infected $I(\tau)$, recall the process outlined in Section 2.5:

$$I(\tau_i) - I(\tau_{i-1}) \sim \text{Bin}(N - I(\tau_{i-1}), p(\tau_i - \tau_{i-1})), \tag{5.2}$$

where N is the total population size and $p(\tau) = 1 - \exp(-b\tau)$ (Cook et al., 2008); we assume that $\tau_0 = 0$ and $I(\tau_0) = 0$. Using Equation 5.2 we can therefore sequentially compute the data $I(\tau^i)$ for $i = 1, \ldots, 4$ and use that as our summary statistics in the LFIRE ratio computation. The rest of the sequential design procedure is as described in Algorithm 8, except for the fact that the dimension of the non-myopic design variable gets reduced by one at every iteration. We choose to make an observation at the first optimal time $\tau^{1,*}$ for every iteration. Just like in Section 5.2, we treat every iteration as a new, independent experiment and therefore the boundaries of the time domain do not change, i.e. we start at $\tau = 0$ again at every iteration. This might seem slightly counter-intuitive but may actually be more realistic in situations where looking at the population interferes in the model process (Cook et al., 2008).

As done in Section 4.6.1, we shall use a truncated Gaussian prior of mean 1 and variance 1 such that the model parameter $b > 0$; we start out with an initial population size of $N = 50$. We again use the domain boundaries $0 \leq \tau^i \leq 4$ for all time variables, as well as enforcing the constraint in Equation 5.1. In order to find the optimal non-myopic design times $\boldsymbol{\tau}_t^*$ for every iteration, we then apply Algorithm 8 to the Death Model.

Since it is difficult to visualise the expected utility surface for the multi-dimensional non-myopic design points. We shall therefore only state the optimal design points for every iteration in Table 5.1.

Iteration	$\tau^{1,*}$	$\tau^{2,*}$	$\tau^{3,*}$	$\tau^{4,*}$
1	2.23	3.40	3.72	3.99
2	0.75	2.18	3.17	
3	0.98	2.57		
4	1.22			

Table 5.1 Non-Myopic Optimal Designs for the Death Model.

For the first iteration, we find that the first optimal time is at about the mid-point of the domain, while the rest of the optimal times are clustered towards the end. For the second and third iteration, we find that the optimal times are more equally spread in the domain. Finally, for the last iteration, the optimal time is in the same region as what we found for myopic design in Section 4.6.1. The real-world observations were made at times 2.23, 0.75, 0.98 and 1.22 for the first to fourth iteration, respectively. Recall that we ran each iteration as if it was a new experiment with identical settings, except reduced dimensions of the design variable, and then chose the first optimal time of the design variable to make our observation at. These optimal times are significantly different than what we obtained for myopic design. Thus, great take care has to be taken when defining the premise, as we obtain different optimal designs for myopic and non-myopic designs.

We can compare the overall 'value' of these non-myopic optimal designs to that of the myopic optimal design points obtained in Section 5.2 by evaluating the non-myopic expected utility for both. For instance, let us use the same truncated Gaussian prior distribution as above. We then sort the optimal design points, obtained above and in Section 5.2, and define a non-myopic optimal design vector $\boldsymbol{\tau}^*_{non-myopic} = [0.75, 0.98, 1.22, 2.23]^\top$ and a myopic optimal design vector $\boldsymbol{\tau}^*_{myopic} = [1.06, 1.06, 1.08, 1.11]^\top$. When evaluating the expected utility for the first iteration for both of these vectors, we obtain $U_1(\boldsymbol{\tau}^*_{non-myopic}) = 0.93$ and $U_1(\boldsymbol{\tau}^*_{myopic}) = 0.80$. The expected utility for the non-myopic optimal designs is higher than that for myopic design points; this means that, in the case of the Death model, it is more beneficial to perform non-myopic design if we know that we can only do a certain number, e.g. 4, of experiments.

As done before, we can look at the parameter posterior distributions as a function of iterations. By means of Algorithm 4, we obtain 10,000 posterior samples using the weights computed after each iteration and the initial prior samples. The resulting posterior distributions are shown in Figure 5.5.

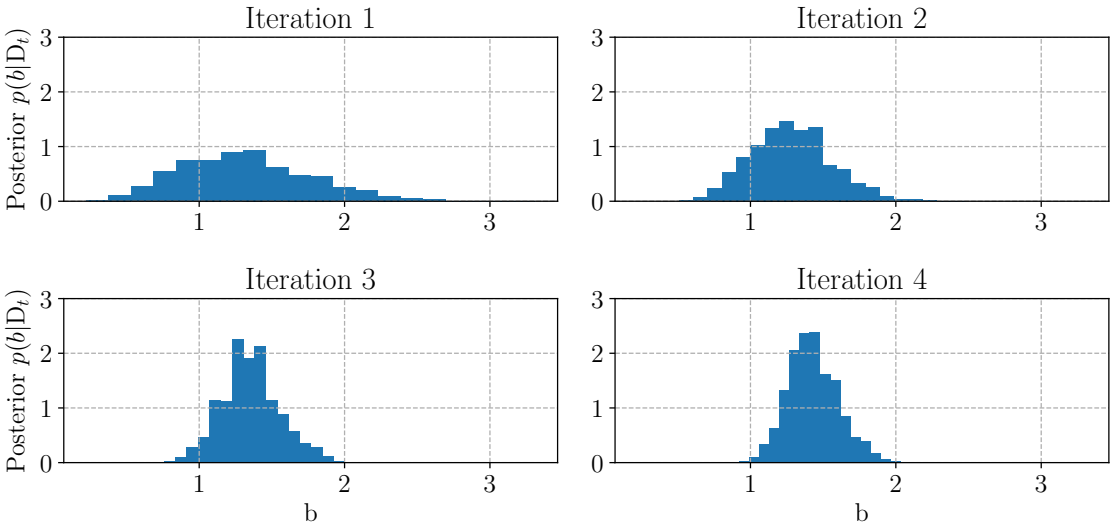

Fig. 5.5 Death Model posterior distributions, for four iterations, using the Bayesian Optimisation method and for non-myopic design. Posterior samples have been computed using the weights $w_{t+1}^{(i)}$ for each initial prior sample and Algorithm 4.

We find that the posterior distribution gets progressively narrower and shifts towards the true parameter value of $b_{true} = 1.5$; we observed the same for the myopic design case in Figure 5.2. Using the posterior samples from the fourth iteration, we can obtain estimates of the true parameter value. We find that, for non-myopic design, $\widehat{b} = 1.41^{+0.39}_{-0.30}$, where the errors define a 95% confidence interval. This is, in fact, a more accurate and more precise estimate than what we obtained in Section 5.2 for the myopic design setting, reflecting the result that we obtain a higher expected utility with the non-myopic optimal designs.

Chapter 6

Conclusions

6.1 Contributions

In this work we presented a novel sequential Bayesian experimental design framework for simulator-based statistical models. We devised a sequential design scheme that uses grid search in order to maximise the expected utility function. In addition to a linear toy model, we tested this algorithm on two epidemiological models, the Death Model (Cook et al., 2008) and the SIR Model (Allen, 2008). We found that the parameters of all models were estimated well within the error bounds. While the algorithm worked well, we noticed that there were many expected utility evaluations that were unnecessary, as they were far from the optimum.

In order to solve this, we described a sequential design framework that uses Bayesian Optimisation instead of grid search. We tested this algorithm on the two epidemiological models and found that we needed less expected utility evaluations to solve the optimisation problem, while still getting accurate parameter estimation and a higher resolution around the optimum. Overall, Bayesian Optimisation allowed for a higher resolution around the optimum, while still exploring enough of the design space.

Lastly, we considered non-myopic design problems, i.e. situations where the aim is to find a set of optimal design points and not just a single one. We found the design scheme to be working well for the Death Model; we obtained optimal design points that were different than for the myopic viewpoint, resulting in higher expected utilities and more accurate parameter estimation than for the myopic design setting.

All of these findings successfully answer the research questions that we laid out in Chapter 1 and Chapter 3.

6.2 Limitations

While our sequential Bayesian experimental design framework for simulator models worked well, we can identify a few things that could be problematic. First of all, performance is highly dependent on the choice of prior and the initial parameter samples. If the prior is too wide, we may find that the expected utility approximations are not accurate, as only few of the samples have sufficiently high weights. With enough iterations we can rectify this by means of the resampling procedure. If the prior is too narrow however, we may not capture all modes of the posterior distribution and hence get meaningless expected utilities. We observed this for both epidemiological models. Although we did not use any summary statistics for both example models, for more complex models the choice of summary statistics is important. LFIRE inherently does automatic summary statistic selection but that does not mean that they are sufficient. In addition, the aforementioned design scheme requires computation times of several hours; in situations where immediate action is required, other decision processes have to be used.

6.3 Future Work

Since this particular research area is largely unexplored, there is plenty of possible future work that can be done. First of all, we are only aware of one work (Hainy et al., 2016a) that performs sequential Bayesian experimental design for simulator models. The authors used Approximate Bayesian Computation (ABC) and Synthetic Likelihood (SL) to obtain samples from the posterior distribution, while we used LFIRE which generalises SL (Dutta et al., 2016). However, they only applied it to a simple toy model and used a simplistic utility function and optimisation method. One could test ABC and SL in sequential design for more difficult models and also use them in approximating the mutual information expected utility. On the same note, one could test all of these likelihood-free inference methods for different utility functions in the sequential design setting. While there has been some work on doing this for static experimental design (e.g. Dehideniya et al., 2018), we are not aware of any work doing this for sequential experimental design. In order to test our design algorithms we have used epiodemiological models where the design variable is time. It would be useful to test how well this works for other design variables, such as space, especially in the non-myopic setting. Bayesian experimental design is known to scale badly with the number of parameter and design dimensions. It would thus be interesting to test out the performance limits of the design schemes devised in this work and find ways to make them as efficient as possible.

References

Agostinelli, S., Allison, J., Amako, K., Apostolakis, J., M AraÃºjo, H., Arce, P., Asai, M., A Axen, D., Banerjee, S., Barrand, G., Behner, F., Bellagamba, L., Boudreau, J., Broglia, L., Brunengo, A., Chauvie, S., Chuma, J., Chytracek, R., Cooperman, G., and Zschiesche, D. (2003). Geant4-a simulation toolkit. 506:250.

Allen, L. J. S. (2008). *An Introduction to Stochastic Epidemic Models*, pages 81–130. Springer Berlin Heidelberg, Berlin, Heidelberg.

Alsing, J., Wandelt, B., and Feeney, S. (2018). Massive optimal data compression and density estimation for scalable, likelihood-free inference in cosmology. *MNRAS*, 477:2874–2885.

Atkinson, A. C. and Donev, A. N. (1992). *Optimum Experimental Designs*.

Beaumont, M., Zhang, W., and Balding, D. (2002). Approximate bayesian computation in population genetics. *Genetics*, 162(4):2025–2035.

Bentley, J. L. (1975). Multidimensional binary search trees used for associative searching. *Commun. ACM*, 18(9):509–517.

Bergstra, J. and Bengio, Y. (2012). Random search for hyper-parameter optimization. *J. Mach. Learn. Res.*, 13:281–305.

Braeunig, R. A. (2014). Atmospheric models. http://www.braeunig.us/space/atmmodel.htm. Accessed: 12-07-2018.

Cook, A. R., Gibson, G. J., and Gilligan, C. A. (2008). Optimal observation times in experimental epidemic processes. *Biometrics*, 64(3):860–868.

Dehideniya, M. B., Drovandi, C. C., and McGree, J. M. (2018). Optimal bayesian design for discriminating between models with intractable likelihoods in epidemiology. *Computational Statistics & Data Analysis*, 124:277 – 297.

Dietz, K. (1993). The estimation of the basic reproduction number for infectious diseases. *Statistical Methods in Medical Research*, 2(1):23–41. PMID: 8261248.

Drovandi, C. C. and Pettitt, A. N. (2013). Bayesian experimental design for models with intractable likelihoods. *Biometrics*, 69(4):937–948.

Dutta, R., Corander, J., Kaski, S., and Gutmann, M. U. (2016). Likelihood-free inference by ratio estimation. *ArXiv e-prints*.

Fedorov, V. (1972). *Theory of Optimal Experiments Designs*.

González, J., Dai, Z., Hennig, P., and Lawrence, N. D. (2015). Batch Bayesian Optimization via Local Penalization. *ArXiv e-prints*.

GPyOpt (2016). GPyOpt: A bayesian optimization framework in python. http://github.com/SheffieldML/GPyOpt.

Hainy, M., Drovandi, C. C., and McGree, J. (2016a). Likelihood-free extensions for bayesian sequentially designed experiments. In Kunert, J., Muller, C. H., and Atkinson, A. C., editors, *11th International Workshop in Model-Oriented Design and Analysis (mODa 2016)*, pages 153–161, Hamminkeln, Germany. Springer.

Hainy, M., Müller, W. G., and Wagner, H. (2016b). Likelihood-free simulation-based optimal design with an application to spatial extremes. *Stochastic Environmental Research and Risk Assessment*, 30(2):481–492.

Kish, L. (1965). *Survey sampling*. Chichester : Wiley New York.

Kullback, S. and Leibler, R. A. (1951). On information and sufficiency. *Ann. Math. Statistics*, 22:79–86.

M. Schafer, C. and Freeman, P. (2012). Likelihood-free inference in cosmology: Potential for the estimation of luminosity functions. 209:3–19.

McCandless, D. (2014). Microbe-scope. http://bit.ly/KIB_Microbescope.

Mockus, J., Tiesis, V., and Zilinskas, A. (1978). *The application of Bayesian methods for seeking the extremum*, volume 2.

Müller, P. (1999). Simulation-based optimal design.

Murphy, K. P. (2012). *Machine Learning: A Probabilistic Perspective*. The MIT Press.

Numminen, E., Cheng, L., Gyllenberg, M., and Corander, J. (2013). Estimating the transmission dynamics of streptococcus pneumoniae from strain prevalence data. *Biometrics*, 69(3):748–757.

Overstall, A. M. and McGree, J. M. (2018). Bayesian design of experiments for intractable likelihood models using coupled auxiliary models and multivariate emulation. *ArXiv e-prints*.

Price, D. J., Bean, N. G., Ross, J. V., and Tuke, J. (2016). On the efficient determination of optimal bayesian experimental designs using abc: A case study in optimal observation of epidemics. *Journal of Statistical Planning and Inference*, 172:1 – 15.

Rasmussen, C. E. and Williams, C. K. I. (2005). *Gaussian Processes for Machine Learning (Adaptive Computation and Machine Learning)*. The MIT Press.

Ricker, W. E. (1954). Stock and recruitment. *Journal of the Fisheries Research Board of Canada*, 11(5):559–623.

Rubin, D. B. (1984). Bayesianly justifiable and relevant frequency calculations for the applied statistician. *Ann. Statist.*, 12(4):1151–1172.

Ryan, C. M., Drovandi, C. C., and Pettitt, A. N. (2016a). Optimal bayesian experimental design for models with intractable likelihoods using indirect inference applied to biological process models. *Bayesian Analysis*, 11(3):857–883.

Ryan, E. G., Drovandi, C. C., McGree, J. M., and Pettitt, A. N. (2016b). A review of modern computational algorithms for bayesian optimal design. *International Statistical Review*, 84(1):128–154.

Shahriari, B., Swersky, K., Wang, Z., Adams, R. P., and de Freitas, N. (2016). Taking the human out of the loop: A review of bayesian optimization. *Proceedings of the IEEE*, 104(1):148–175.

Sjöstrand, T., Mrenna, S., and Skands, P. (2008). A brief introduction to PYTHIA 8.1. *Computer Physics Communications*, 178:852–867.

Williams, D. R. (2017). Earth fact sheet. https://nssdc.gsfc.nasa.gov/planetary/factsheet/earthfact.html. Accessed: 09-07-2018.

Wood, S. N. (2010). Statistical inference for noisy nonlinear ecological dynamic systems. *Nature*, 466:1102.